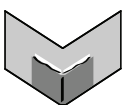
The Journal of Machine Learning Research
Volume 12
Print-Archive Edition

Pages 2335–3466



Microtome Publishing
Brookline, Massachusetts
www.mtome.com

The Journal of Machine Learning Research
Volume 12
Print-Archive Edition

The Journal of Machine Learning Research (JMLR) is an open access journal. All articles published in JMLR are freely available via electronic distribution. This Print-Archive Edition is published annually as a means of archiving the contents of the journal in perpetuity. The contents of this volume are articles published electronically in JMLR in 2011.

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

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

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

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

JMLR Editorial Board

Journal of Machine Learning Research

Volume 12, 2011

Journal of Machine Learning Research 12 (2011) 2335-2382 Submitted 1/10; Revised 8/10; Published 7/11

Producing Power-Law Distributions and Damping Word Frequencies with Two-Stage Language Models

Sharon Goldwater SGWATER@INF.ED.AC.UK
School of Informatics
10 Crichton Street
Edinburgh, EH8 9AB
United Kingdom

Thomas L. Griffiths TOM_GRIFFITHS@BERKELEY.EDU
Department of Psychology
University of California, Berkeley
3210 Tolman Hall, MC 1650
Berkeley, CA 94720-1650
USA

Mark Johnson MJOHNSON@SCIENCE.MQ.EDU.AU
Department of Computing
Macquarie University
Sydney, NSW 2109
Australia

Editor: Fernando Pereira

Abstract

Standard statistical models of language fail to capture one of the most striking properties of natural languages: the power-law distribution in the frequencies of word tokens. We present a framework for developing statistical models that can generically produce power laws, breaking generative models into two stages. The first stage, the generator, can be any standard probabilistic model, while the second stage, the adaptor, transforms the word frequencies of this model to provide a closer match to natural language. We show that two commonly used Bayesian models, the Dirichlet-multinomial model and the Dirichlet process, can be viewed as special cases of our framework. We discuss two stochastic processes—the Chinese restaurant process and its two-parameter generalization based on the Pitman-Yor process—that can be used as adaptors in our framework to produce power-law distributions over word frequencies. We show that these adaptors justify common estimation procedures based on logarithmic or inverse-power transformations of empirical frequencies. In addition, taking the Pitman-Yor Chinese restaurant process as an adaptor justifies the appearance of type frequencies in formal analyses of natural language and improves the performance of a model for unsupervised learning of morphology.

Keywords: nonparametric Bayes, Pitman-Yor process, language model, unsupervised

1. Introduction

It is important for models used in unsupervised learning to be able to describe the gross statistical properties of the data they are intended to learn from, otherwise these properties may distort inferences about the parameters of the model. One of the most striking statistical properties of nat-

ural languages is that the distribution of word frequencies is closely approximated by a power law. That is, the probability that a word w will occur with frequency n_w in a sufficiently large corpus is proportional to n_w^{-g}. This observation—usually attributed to Zipf (1932), though it enjoys a long and detailed history (Mitzenmacher, 2004)—stimulated intense research in the 1950s (e.g., Simon, 1955) but has largely been ignored in modern machine learning and computational linguistics.

By developing models that can generically exhibit power laws, it may be possible to improve methods for identifying structure in linguistic data. In particular, postulating a separate mechanism within the model that accounts for the skewed distribution of word frequencies takes the burden of explaining this distribution off the other components of the model, effectively reducing the frequencies of those words. Such "damping" of word frequencies can often be desirable. It is commonly observed in applications of statistical natural language processing that reducing the counts of word tokens, typically by taking their logarithms or inverse powers, can improve performance (Salton and Buckley, 1988).

An extreme version of damping frequencies forms part of a tension exhibited by formal approaches to natural language: whether explanations should be based upon the distinct *types* of words that languages exhibit, or the frequencies with which *tokens* (instances) of those words occur. One place where this tension manifests is in accounts of morphology (the substructure of words), where formal linguists develop accounts of why particular words appear in the lexicon (e.g., Pierrehumbert, 2003), while computational linguists focus on statistical models of the frequencies of tokens of those words (e.g., Hakkani-Tür et al., 2002). The same tension arises in various areas of statistical natural language processing and related fields. For example, one of the most successful forms of smoothing used in statistical language models, Kneser-Ney smoothing, explicitly interpolates between type and token frequencies (Ney et al., 1994; Kneser and Ney, 1995; Chen and Goodman, 1998). Information retrieval systems can also differ in whether they use binary vectors indicating the presence or absence of words in a document or a full vector of word frequencies (Baeza-Yates and Ribeiro-Neto, 1999), and the same distinction appears in machine learning methods applied to text (e.g., Blei et al., 2003; Thibaux and Jordan, 2007).

In this paper, we present a framework for developing generative models for language that produce power-law distributions. Our framework is based upon the idea of specifying these models in terms of two components: a *generator*, an underlying generative model for words which need not (and usually does not) produce a power-law distribution, and an *adaptor*, which transforms the stream of words produced by the generator into one whose frequencies obey a power-law distribution. This framework is extremely general: any generative model for language can be used as a generator, with the power-law distribution being produced as the result of making an appropriate choice for the adaptor.

Adopting this two-stage framework divides responsibility for the appearance of the tokens in the corpus between the generator and the adaptor, with only a subset of the tokens being produced by the generator. The parameters of the generator will be estimated based only on the tokens for which the generator is considered responsible, rather than on the full set of tokens in the corpus. By explaining away the presence of some of the tokens, the adaptor effectively damps the word counts used to estimate the parameters of the generator. Estimation of these parameters will thus be affected by assumptions about the form of the adaptor. We consider several adaptor-generator pairs, focusing especially on the Chinese restaurant process (Aldous, 1985) and its two-parameter generalization, derived from the Pitman-Yor process (Pitman, 1995; Pitman and Yor, 1997; Ishwaran and James, 2003), as adaptors. We show that using these stochastic processes as adaptors can

produce appropriate power-law distributions while implementing different forms of damping. We also show that the Pitman-Yor generalization of the Chinese restaurant process can be used to justify parameter estimation based purely on type frequencies, and demonstrate that using this adaptor improves the performance of a simple two-stage model applied to learning morphology.

Our work contributes to a growing body of research on Bayesian approaches to modeling and learning language. This paper is not the first to propose the use of the Chinese restaurant process or Pitman-Yor process for modeling language, and some of the models we discuss have been used in previous work by ourselves and others (Goldwater et al., 2006a; Teh, 2006b). However, considering these models in greater depth allows us to make several novel contributions. First, we show how the two-stage framework makes it possible to unify a variety of Bayesian models of language. This unified picture offers us a way to concisely summarize existing Bayesian language models, and to identify the mathematical relationships between these models. Second, we provide a quantitative argument that these models are a good fit for language by virtue of the power-law distributions they produce, detailing the differences between the distributions produced by different adaptors, and discussing the use of different approximations. Third, we present new empirical studies that provide insight into the practical effects of different approximations and parameter choices. Finally, we expand on the idea, introduced by Goldwater et al. (2006a), that these models provide a way to understand and model the relationship between linguistic types and tokens, and a mathematical justification for commonly used smoothing and damping techniques.

In addition to considering the general properties of models developed in our two-stage framework, we provide a detailed case study of applying this approach to an unsupervised learning problem: morphological segmentation. In this problem, the goal is to identify the meaningful components from which words are comprised. This problem is challenging because natural languages possess both regular and irregular morphology, with only a subset of words following regular morphological rules. Linguists have long noted a strong relationship between frequency and regularity in language, with irregular forms often being among the most frequent (Greenberg, 1966; Bybee, 1985). Without accounting for this fact, an unsupervised learning system is likely to be misled by the very frequent irregular forms, and fail to appropriately model the regular patterns that are needed to account for infrequent forms, which will comprise most unseen data. We show that the two-stage framework proposed here can explain the relationship between frequency and regularity and thus leads to better learning of regular patterns.

The morphological segmentation task is a good example of a situation where appropriately modeling word frequencies can significantly affect the outcome of unsupervised learning. While we explore this case in detail, the goal of this paper is not to develop state-of-the-art models for any particular application. Rather, we hope to strengthen intuitions and insights into how nonparametric Bayesian models of language behave in general, in order to give other researchers a better sense of when these tools may be helpful and how to use them. We consider other promising applications of this approach, and ways in which it can be extended, in Section 9.

The plan of the paper is as follows. Section 2 summarizes related work. Section 3 discusses stochastic processes that can produce power-law distributions and introduces the generic two-stage modeling framework. Section 4 presents models based on the Chinese restaurant process and Pitman-Yor Chinese restaurant process, stochastic processes from nonparametric Bayesian statistics that produce power-law distributions. Section 5 shows how some other Bayesian language models can be viewed as special cases of our two-stage framework. Section 6 examines some of the consequences of using the adaptors introduced in Section 4: Section 6.1 discusses the implications

of using these models for estimation of the parameters of the generator, Section 6.2 shows that estimation based on type and token frequencies are special cases of a two-stage language model, and Section 6.3 uses these results to provide a novel justification for the use of Kneser-Ney smoothing. Section 7 describes a two-stage model for unsupervised learning of the morphological structure of words, and Section 8 presents the results of some experiments with this model demonstrating that its performance improves as we move from estimation based upon tokens to types. Section 9 discusses additional applications and extensions of our approach, and Section 10 concludes.

2. Related Work

Our two-stage approach fits within a more general trend of using Bayesian models for linguistic data. Previous work has used Bayesian models in two ways: to understand and justify approaches to smoothing, or as a method of unsupervised structure discovery and learning. Since we will touch upon both of these topics in this paper, we now present a brief review of related work in each area.

Smoothing methods are schemes for regularizing empirical estimates of the probabilities of words, with the goal of improving the predictive performance of language models. The simplest kind of smoothing involves adding a small constant to the empirical frequencies of words prior to normalizing those frequencies (Chen and Goodman, 1998). This approach can be shown to be equivalent to Bayesian estimation of a multinomial distribution using a Dirichlet prior (MacKay and Peto, 1994), a method that has more recently evolved into the use of compound Dirichlet-multinomial models for text (Elkan, 2006; Madsen et al., 2005). The observation of a correspondence between smoothing methods and Bayesian inference has been used to define more complex smoothing schemes based on hierarchical Bayesian models (MacKay and Peto, 1994). The connection between Pitman-Yor processes and Kneser-Ney smoothing is one instance of this broader correspondence, and was independently pointed out by Teh (2006a,b) following our own work on this topic (Goldwater et al., 2006a). More recently, Wood and Teh (2008, 2009) have developed more sophisticated cross-domain smoothing models by combining multiple hierarchical Pitman-Yor processes.

Another strand of work on Bayesian models of language aims to improve unsupervised (or semi-supervised) learning of linguistic structure. Much of this work can be traced back to the latent Dirichlet allocation (LDA) model and related work on document clustering and topic modeling by Blei and colleagues (Blei et al., 2002, 2003, 2004). While LDA takes a bag-of-words approach to language modeling, recent research in the computational linguistics community has focused on using similar Bayesian techniques to develop models of linguistic structure with more sophisticated intra- and inter-word dependencies. For example, Goldwater et al. (2006b) presented a model based on the hierarchical Dirichlet process (Teh et al., 2005) to identify word boundaries in unsegmented text. This model is very similar to the hierarchical Pitman-Yor language model described in Section 6.3 as well as in Teh (2006a). Finkel et al. (2007) and Liang et al. (2007) introduced models for learning better syntactic categories for parsing by extending the idea of the infinite hidden Markov model (Beal et al., 2002; Teh et al., 2005) to probabilistic context-free grammars (PCFGs) and dependency trees. Johnson et al. (2007) described a different kind of infinite Bayesian model for learning grammatical structure, the adaptor grammar, which is more directly based on the two-stage framework presented here. An adaptor grammar can be seen as a two-stage model in which the generator is a PCFG. Adaptor grammars have since been used for learning word segmentation, syllable structure, and morphology in English and Sesotho (Johnson et al., 2007; Johnson, 2008a,b),

as well as for named-entity clustering (Elsner et al., 2009). They have also been extended by Cohn et al. (2009), Post and Gildea (2009), and O'Donnell et al. (2009), who independently proposed very similar generalizations of the adaptor grammar for learning tree substitution grammars.

Finally, although this paper focuses primarily on the general Bayesian framework rather than the specific application to morphological learning that we discuss in Sections 7 and 8, it is worth mentioning a few other notable approaches to the unsupervised learning of morphology. Probably the most well-known systems are Linguistica (Goldsmith, 2001, 2006) and Morfessor (Creutz and Lagus, 2004, 2005), both of which are based on probabilistic models using maximum *a posteriori* estimation, and are freely available for download. A number of other systems use more heuristic approaches; Goldsmith (2001) provides a thorough review. An interesting recent approach uses sentence-aligned multilingual texts to perform simultaneous morphological segmentation on multiple languages (Snyder and Barzilay, 2008). The Bayesian model used in that work can be viewed as an extension of the word segmentation model of Goldwater et al. (2006b) described above.

3. The Two-stage Approach

The key idea behind our two-stage framework is to divide the process of generating text into two parts, one of which is sufficient to produce a power-law distribution over word frequencies. In this section we briefly review mechanisms that give rise to power-law distributions and then formally define our framework.

3.1 Producing Power-law Distributions

Assume we want to generate a sequence of n outcomes, $\mathbf{z} = (z_1, \ldots, z_n)$, with each outcome z_i being drawn from a set of (possibly unbounded) size K. Many of the stochastic processes that produce power laws are based upon the principle of *preferential attachment*, where the probability that the ith outcome, z_i, takes on a particular value k depends upon the frequency of k in $\mathbf{z}_{-i} = (z_1, \ldots, z_{i-1})$ (Mitzenmacher, 2004). For example, the number of links pointing to a given web page is sometimes modeled as a power-law distribution, which can be explained by assuming that new web pages are more likely to include links to already-popular pages (Mitzenmacher, 2004). An early preferential attachment process, due to Simon (1955), chooses z_i according to

$$P(z_i = k \,|\, \mathbf{z}_{-i}) = a\frac{1}{K} + (1-a)\frac{n_k^{(\mathbf{z}_{-i})}}{i-1}$$

where $n_k^{(\mathbf{z}_{-i})}$ is the number of times k occurs in $\mathbf{z}_{-i}$, and $0 < a < 1$ is a parameter of the process. This "rich-get-richer" process means that a few outcomes appear with very high frequency in $\mathbf{z}$, while most outcomes appear with low frequency—the key attribute of a power-law distribution. In this case, the power law has parameter $g = 1/(1-a)$.

One problem with this kind of model is that different permutations of the outcomes $\mathbf{z}$ have different probabilities. While this may be appropriate for some settings, the assumption of a temporal ordering restricts the contexts in which such models can be applied. In particular, it is much more restrictive than the assumption of independent sampling that underlies most statistical language models. Consequently, we will focus on a different preferential attachment scheme, based upon the two-parameter species sampling model (Pitman, 1995; Pitman and Yor, 1997) known as the Pitman-Yor process (Ishwaran and James, 2003). We will refer to this scheme as the Pitman-Yor Chinese

restaurant process (PYCRP), as it is a generalization of the more widely known Chinese restaurant process (CRP; Aldous, 1985). Under these schemes, outcomes follow a power-law distribution, but remain *exchangeable*: the probability of a set of outcomes is not affected by their ordering (Aldous, 1985). In addition to its theoretical benefits, the property of exchangeability has practical value in permitting the use of standard sampling algorithms for inference. We return to discussion of the CRP and PYCRP in Section 4 after introducing the basic conceptual framework of the two-stage language model.

3.2 The Generator and Adaptor

In our two-stage modeling framework, a sequence of word tokens $\mathbf{w} = (w_1, \ldots, w_n)$ is generated as follows:

1. Generate a sequence of lexical items $\boldsymbol{\ell} = (\ell_1, \ldots, \ell_K)$ from some probability distribution P_φ parameterized by φ. For example, $(\ell_1, \ldots, \ell_4)$ = (`the`, `dog`, `a`, `the`). We refer to P_φ as the *lexicon generator* (or simply *generator*). Note that our use of the term *lexical item* is non-standard. Ignoring homophony, a lexicon normally contains one instance of each word type. Here, P_φ is a discrete distribution and the lexical items are generated independently, so the same word type may occur more than once in $\boldsymbol{\ell}$.[1] In the remainder of the paper, we use *lexical item* to refer to the items produced by the generator, *word type* to refer to unique wordforms, and *word* or *token* to refer to word tokens.

2. Generate a sequence of integers $\mathbf{z} = (z_1, \ldots, z_n)$ with $1 \leq z_i \leq K$, where $z_i = k$ indicates that $w_i = \ell_k$ (that is, z_i is the index of the lexical item corresponding to w_i). For example, $(z_1, \ldots, z_9) = (1,2,1,1,3,1,1,4,3)$, so that, in combination with $(\ell_1, \ldots, \ell_4)$ from above, $(w_1, \ldots, w_9)$ = (`the`, `dog`, `the`, `the`, `a`, `the`, `the`, `the`, `a`). The integers $\mathbf{z}$ are assumed to be generated by some stochastic process P_γ with one or more parameters γ. We refer to this process as the *adaptor*.

We use the notation TwoStage(P_γ, P_φ) to refer to a two-stage model with adaptor P_γ and generator P_φ. A graphical model illustrating the dependencies between the variables in this framework is shown in Figure 1.

The two-stage modeling framework is very general: many different distributions could be used for the generator and adaptor. However, given the discussion above, it is sensible to assume that P_γ is chosen so that the frequencies with which different integer outcomes are produced follow a power-law distribution. In this case, when P_φ is a distribution with infinite support, the power-law distribution over integers produced in Step 2 will result in a power-law distribution over the frequencies in the final sequence of words. Thus, the adaptor "adapts" the word frequencies produced by the generator to fit a power-law distribution. Different choices for the generator model will allow different kinds of linguistic structure to be learned. Here, we show that morphological structure can be learned using a generator that produces words by choosing a stem and suffix and concatenating them together. In other work, we have used different generators to discover word boundaries in unsegmented text (Goldwater et al., 2006b; Johnson, 2008a) and to infer tree substitution grammars from parsed corpora or strings (Cohn et al., 2010).

1. The assumption of independence between lexical items is not strictly necessary, but is mathematically and computationally convenient. An example of a more complex distribution over lexical items that enforces uniqueness is given in Brent (1999).

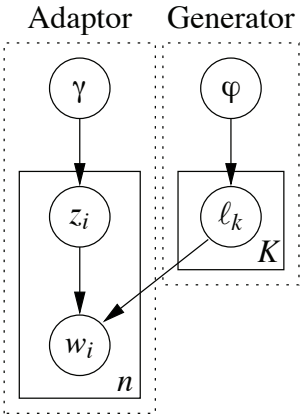


Figure 1: A graphical model representation of the two-stage language modeling framework. Arrows indicate dependencies between variables, and solid-line boxes indicate replicated portions of the model, with the number of copies shown in the lower right hand corner. Variables associated with the generator are on the right; those associated with the adaptor are on the left. Depending on the application, the words w_i may or may not be directly observed.

4. Chinese Restaurant Processes as Adaptors

While any stochastic process that results in a power-law distribution over word frequencies can be used as an adaptor, the choice of adaptor will have significant implications for the resulting model. In this section, we discuss two stochastic processes that are particularly suitable as adaptors in the two-stage framework: the Chinese restaurant process (Aldous, 1985; Pitman, 1995; Griffiths, 2006) and the Pitman-Yor Chinese restaurant process (Pitman, 1995; Pitman and Yor, 1997; Ishwaran and James, 2003). Both the CRP and PYCRP are used in nonparametric Bayesian statistics, with the more widely known CRP arising as the distribution over the sizes of mixture components in infinite mixture models (Rasmussen, 2000). We review the definitions of these processes, discuss the properties that make them useful as adaptors, and define the two-stage models that result from using CRP or PYCRP adaptors.

4.1 The Chinese Restaurant Process

The Chinese restaurant process is a simple stochastic process that can be described using the analogy of a restaurant with an infinite number of tables, each of which has an infinite seating capacity. Customers enter the restaurant one at a time, and choose a table at which to sit. The probability of choosing an occupied table is proportional to the number of people already sitting there, and the probability of choosing an unoccupied table is proportional to some constant parameter α. That is, if z_i is the index of the table chosen by the ith customer, then

$$P(z_i = k \,|\, \mathbf{z}_{-i}, \alpha) = \begin{cases} \frac{n_k^{(\mathbf{z}_{-i})}}{i-1+\alpha} & 1 \leq k \leq K(\mathbf{z}_{-i}) \\ \frac{\alpha}{i-1+\alpha} & k = K(\mathbf{z}_{-i})+1 \end{cases}$$

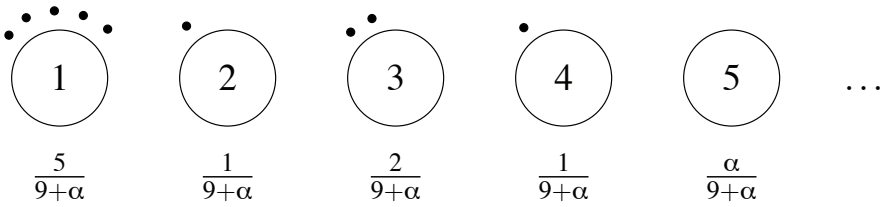


Figure 2: An illustration of the Chinese restaurant process, reproduced from Goldwater et al. (2009). Black dots indicate the number of customers sitting at each table for the example case $\mathbf{z}_{-10} = (1,2,1,1,3,1,1,4,3)$. Below each table is $P(z_{10} = k \,|\, \mathbf{z}_{-10})$. Note that the number of customers at each table—and thus $P(z_{10} = k \,|\, \mathbf{z}_{-10})$, the probability distribution over the next customer—would remain the same for any ordering of the integers in $\mathbf{z}_{-10}$. This is the property of exchangeability.

where $\mathbf{z}_{-i}$ is the seating arrangement of the previous $i-1$ customers, $n_k^{(\mathbf{z}_{-i})}$ is the number of customers already assigned to table k by $\mathbf{z}_{-i}$, $K(\mathbf{z}_{-i})$ is the total number of occupied tables in $\mathbf{z}_{-i}$, and $\alpha \geq 0$ is a parameter of the process determining how "spread out" the customers become. Higher values of α mean that more new tables will be occupied relative to the number of customers, leading to a more uniform distribution of customers across tables. The first customer by definition sits at the first table, so this distribution is well-defined even when $\alpha = 0$. See Figure 2 for an illustration.

Under this model, the probability of a particular sequence of table assignments for n customers is given by

$$\begin{aligned} P(\mathbf{z} \,|\, \alpha) &= 1 \cdot \prod_{i=2}^{n} P(z_i \,|\, \mathbf{z}_{-i}, \alpha) \\ &= \left(\prod_{i=2}^{n} \frac{1}{i-1+\alpha} \right) \left(\alpha^{K(\mathbf{z})-1} \right) \left(\prod_{k=1}^{K(\mathbf{z})} (n_k^{(\mathbf{z})} - 1)! \right) \\ &= \frac{\Gamma(1+\alpha)}{\Gamma(n+\alpha)} \cdot \alpha^{K(\mathbf{z})-1} \cdot \prod_{k=1}^{K(\mathbf{z})} (n_k^{(\mathbf{z})} - 1)! \end{aligned} \tag{1}$$

where the Gamma function is defined as $\Gamma(x) = \int_0^\infty u^{x-1} e^{-u} du$ for $x > 0$, and is a generalized factorial function: $\Gamma(x) = (x-1)!$ for positive integer x, and $\Gamma(x) = (x-1)\Gamma(x-1)$ for any $x > 0$.[2]

It is easy to see that any reordering of the table assignments in $\mathbf{z}$ will result in the same factors in Equation 1, so the CRP is exchangeable.[3] As the number of customers becomes large, the CRP produces a power-law distribution over the number of customers seated at each table, where the power-law exponent g is equal to 1 (Arratia et al., 1992).

2. It is more standard to see the joint distribution of table assignments in the CRP given as $P(\mathbf{z}) = \frac{\Gamma(\alpha)}{\Gamma(n+\alpha)} \cdot \alpha^{K(\mathbf{z})} \cdot \prod_{k=1}^{K(\mathbf{z})} (n_k^{(\mathbf{z})} - 1)!$. This distribution is derived from the Dirichlet process (see Section 5.2), which is defined only for $\alpha > 0$, and is equivalent to Equation 1 in that case. We use the distribution in Equation 1 because it is defined also for $\alpha = 0$, which is a possible (if uninteresting) parameter value in the CRP.

3. When considering exchangeability, the table assignments should be viewed as partitioning the integers $1, \ldots, i$ into equivalence classes. The requirement that $z_i \in 1, \ldots, \max(\mathbf{z}_{i-1}) + 1$ ensures there is a 1-to-1 mapping between equivalence classes and the set of integers in $\mathbf{z}$.

The preceding paragraphs indicate how the CRP can be used to create a power-law distribution over integers, but to create a distribution over words we need to combine it with a lexicon generator to make a full two-stage model. For expository purposes, we continue to use the generic lexicon generator P_φ, a distribution parameterized by φ, so the full model is TwoStage(CRP(α),P_φ). This model can be viewed as a restaurant in which each table is labeled with a lexical item produced by P_φ. Each customer represents a word token, so that the number of customers at a table corresponds to the frequency of the lexical item labeling that table. A new word token is generated by seating a new customer, producing either a new token of an existing lexical item (if the customer sits at an existing table: in this case the new token will have the same word type as the lexical item labeling that table) or the first token of a new lexical item (if the customer sits at a new table: in this case a new label is generated using P_φ, and all later customers at this table will be additional tokens of the same word type).

Under this model, the probability that the ith token in a sequence takes on the value w, given the previous labels and table assignments, can be found by summing over all the existing tables labeled with w, plus a possible new table labeled with w:

$$\begin{aligned}
&P(w_i = w \,|\, \mathbf{z}_{-i}, \boldsymbol{\ell}(\mathbf{z}_{-i}), \alpha, \varphi) \\
&\quad = \sum_{k=1}^{K(\mathbf{z}_{-i})} P(w_i = w \,|\, z_i = k, \ell_k) P(z_i = k \,|\, \mathbf{z}_{-i}, \alpha) \\
&\qquad + P(w_i = w \,|\, z_i = K(\mathbf{z}_{-i}) + 1, \varphi) P(z_i = K(\mathbf{z}_{-i}) + 1 \,|\, \mathbf{z}_{-i}, \alpha) \\
&\quad = \sum_{k=1}^{K(\mathbf{z}_{-i})} I(\ell_k = w) \frac{n_k^{(\mathbf{z}_{-i})}}{i - 1 + \alpha} + P_\varphi(w) \frac{\alpha}{i - 1 + \alpha} \\
&\quad = \frac{n_w^{(\mathbf{w}_{-i})} + \alpha P_\varphi(w)}{i - 1 + \alpha} \qquad (2)
\end{aligned}$$

where $\boldsymbol{\ell}(\mathbf{z}_{-i})$ are the labels of all the tables in $\mathbf{z}_{-i}$, $I(.)$ is an indicator function taking on the value 1 when its argument is true and 0 otherwise, and $n_w^{(\mathbf{w}_{-i})}$ is the number of previous occurrences of the word type w in $\mathbf{w}_{-i}$ (that is, the number of customers that $\mathbf{z}_{-i}$ assigns to tables labeled with w). This distribution is illustrated in Figure 3.

The probability of an entire sequence of words $P(\mathbf{w} \,|\, \alpha, \varphi)$ can be found by marginalizing out $\boldsymbol{\ell}$ and $\mathbf{z}$ from the joint distribution $P(\mathbf{w}, \mathbf{z}, \boldsymbol{\ell} \,|\, \alpha, \varphi)$. Note that unless $\ell_{z_i} = w_i$ for all i, $P(\mathbf{w}, \mathbf{z}, \boldsymbol{\ell} \,|\, \alpha, \varphi) = 0$, so we need only sum over cases where $\ell_{z_i} = w_i$ for all i. In this situation, $P(\mathbf{w}, \mathbf{z}, \boldsymbol{\ell} \,|\, \alpha, \varphi) = P(\mathbf{z}, \boldsymbol{\ell} \,|\, \alpha, \varphi) = P(\mathbf{z} \,|\, \alpha) P_\varphi(\boldsymbol{\ell})$,[4] so we can compute the desired distribution as

$$\begin{aligned}
P(\mathbf{w} \,|\, \alpha, \varphi) &= \sum_{\mathbf{z}, \boldsymbol{\ell}} P(\mathbf{z} \,|\, \alpha) P_\varphi(\boldsymbol{\ell}) \\
&= \sum_{\mathbf{z}, \boldsymbol{\ell}} \frac{\Gamma(1 + \alpha)}{\Gamma(n + \alpha)} \alpha^{K(\mathbf{z}) - 1} \prod_{k=1}^{K(\mathbf{z})} \left(P_\varphi(\ell_k) (n_k^{(\mathbf{z})} - 1)! \right) \qquad (3)
\end{aligned}$$

where the sums range only over those $\boldsymbol{\ell}$ and $\mathbf{z}$ such that $\ell_{z_i} = w_i$ for all i.

4. We use $P_\varphi(\boldsymbol{\ell})$ rather than the equivalent $P(\boldsymbol{\ell} \,|\, \varphi)$ for consistency with our notation for the generator probability of an individual lexical item $P_\varphi(\ell)$; both $P_\varphi(\boldsymbol{\ell})$ and $P(\boldsymbol{\ell} \,|\, \varphi)$ represent the probability of producing lexical items $\boldsymbol{\ell}$ using the generator parameterized by φ. Note that in contrast, $P(\mathbf{w} \,|\, \varphi) \neq P_\varphi(\mathbf{w})$, as the latter requires that all tokens in $\mathbf{w}$ are produced by the generator, whereas the former does not.

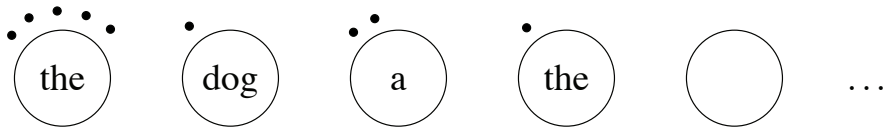


Figure 3: An illustration of the two-stage restaurant, adapted from Goldwater et al. (2009). In this example, $(\ell_1, \ldots, \ell_4)$ = (the, dog, a, the) and $\mathbf{z}_{-10} = (1,2,1,1,3,1,1,4,3)$. Each label ℓ_k is shown on table k. Black dots indicate the number of occurrences of each label in $\mathbf{w}_{-10}$ = (the, dog, the, the, a, the, the, the, a). Under this seating arrangement, $P(w_{10} = \texttt{the}) = \frac{6+\alpha P_\varphi(\texttt{the})}{9+\alpha}$, $P(w_{10} = \texttt{dog}) = \frac{1+\alpha P_\varphi(\texttt{dog})}{9+\alpha}$, $P(w_{10} = \texttt{a}) = \frac{2+\alpha P_\varphi(\texttt{a})}{9+\alpha}$, and for any other word w, $P(w_{10} = w) = \frac{\alpha P_\varphi(w)}{9+\alpha}$.

Notice that the distribution over words given in Equation 2 leads to an alternative way of viewing the TwoStage(CRP(α), P_φ) model, as a cache model. Under this view, each word is generated in one of two ways: from a cache of previously occurring lexical items (with probability $\frac{n}{n+\alpha}$ if we use the CRP adaptor) or as a novel lexical item (with probability $\frac{\alpha}{n+\alpha}$). Items from the cache are chosen with probability proportional to the number of times they have occurred before in $\mathbf{w}$. Novel items are chosen according to the probability distribution of the lexicon generator (which means that, strictly speaking, they are not always "novel"—that is, novel word types—since the generator may produce duplicates). This interpretation clarifies the significance of the parameters α and P_φ. Prior expectations regarding the probability of encountering a novel lexical item are reflected in the value of α, so lower values of α will lead to an expectation of fewer lexical items (and word types) during inference. Prior expectations about the relative probabilities of different novel items are reflected in P_φ, so the choice of generator determines the kinds of lexical items that are likely to be inferred from the data. If the generator is a distribution over an infinite number of items, the cache model makes it clear that the number of different word types that will be observed in a finite corpus is not fixed in advance. Rather, new word types can be generated "on the fly" from an infinite supply. In general, the number of different word types observed in a corpus will slowly grow as the size of the corpus grows.

4.2 The Pitman-Yor Generalization

For much of this paper, we will be focusing on an adaptor based on the Pitman-Yor process. This adaptor is a generalization of the CRP, defined as

$$P(z_i = k \,|\, \mathbf{z}_{-i}, a, b) = \begin{cases} \frac{n_k^{(\mathbf{z}_{-i})} - a}{i-1+b} & 1 \le k \le K(\mathbf{z}_{-i}) \\ \frac{K(\mathbf{z}_{-i})a+b}{i-1+b} & k = K(\mathbf{z}_{-i})+1 \end{cases} \tag{4}$$

where $0 \le a < 1$ and $b \ge 0$ are parameters of the process. As in the CRP, $z_1 = 1$ by definition. When $a = 0$ and $b = \alpha$, this process reduces to the CRP, so we refer to it as the Pitman-Yor Chinese restaurant process (PYCRP). Like the CRP, the PYCRP is exchangeable and produces a power-law distribution on the number of customers seated at each table. In this case, the power-law exponent g

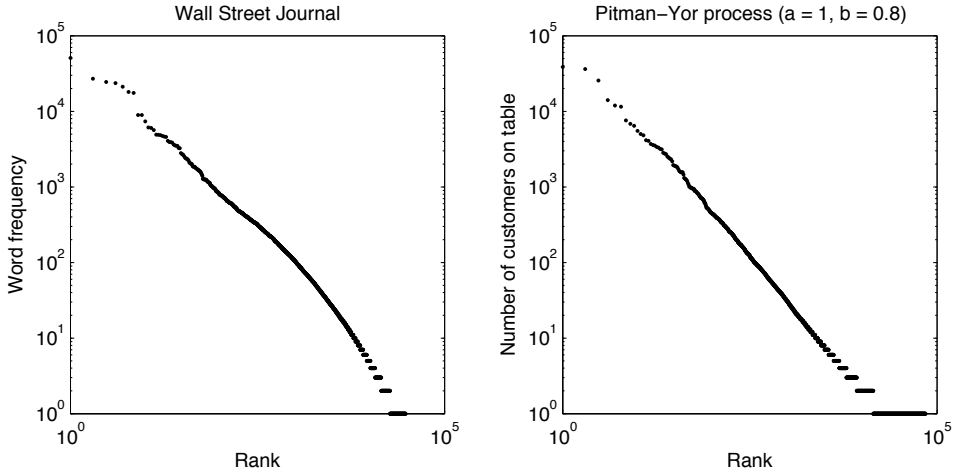


Figure 4: Simulating power laws in natural language, illustrated using Zipf plots. The Zipf plot displays the log frequency of a word as a function of the log of the rank of that frequency (i.e., the number of words with frequency greater than or equal to that word). A power-law distribution in word frequency, with the probability of a frequency of n_w proportional to n_w^{-g}, results in a straight line on the plot with slope $1/(g-1)$. Here, the left-hand plot shows the distribution of word frequencies in sections 0-20 from the Penn Wall Street Journal treebank, while the right-hand plot shows the distribution of the number of customers at each table produced by 500,000 draws from the PYCRP with parameters $a = 0.8$ and $b = 1$. Both plots have a slope of roughly -1.25, corresponding to a power-law distribution with exponent $\gamma = 1.8$.

is equal to $1+a$ (Pitman, 2006), which includes the $g \approx 1.8$ seen for natural languages (see Figure 4). We defer further discussion of the significance of the parameters a and b to Section 6.2.

Under the PYCRP, the probability of a particular seating arrangement $\mathbf{z}$ is

$$\begin{aligned} P(\mathbf{z}\,|\,a,b) &= 1 \cdot \prod_{i=2}^{n} P(z_i\,|\,\mathbf{z}_{-i},a,b) \\ &= \left(\prod_{i=2}^{n} \frac{1}{i-1+b}\right)\left(\prod_{k=1}^{K(\mathbf{z})-1}(ka+b)\right)\left(\prod_{k=1}^{K(\mathbf{z})}\prod_{i=1}^{n_k^{(\mathbf{z})}-1}(i-a)\right) \\ &= \frac{\Gamma(1+b)}{\Gamma(n+b)}\left(\prod_{k=1}^{K(\mathbf{z})-1}(ka+b)\right)\left(\prod_{k=1}^{K(\mathbf{z})}\frac{\Gamma(n_k^{(\mathbf{z})}-a)}{\Gamma(1-a)}\right). \end{aligned}$$

As with the CRP, we can define a generic two-stage model with a PYCRP adaptor by assuming a generator P_φ parameterized by φ. Under this TwoStage(PYCRP$(a,b),P_\varphi$) model, the probability of

generating word w given the seating arrangement and label assignments of the previous words is

$$\begin{aligned}
&P(w_i = w \,|\, \mathbf{z}_{-i}, \boldsymbol{\ell}(\mathbf{z}_{-i}), a, b, \varphi) \\
&\quad = \sum_{k=1}^{K(\mathbf{z}_{-i})} P(w_i = w \,|\, z_i = k, \ell_k) P(z_i = k \,|\, \mathbf{z}_{-i}, a, b) \\
&\qquad + P(w_i = w \,|\, z_i = K(\mathbf{z}_{-i}) + 1, \varphi) P(z_i = K(\mathbf{z}_{-i}) + 1 \,|\, \mathbf{z}_{-i}, a, b) \\
&\quad = \sum_{k=1}^{K(\mathbf{z}_{-i})} I(\ell_k = w) \frac{n_k^{(\mathbf{z}_{-i})} - a}{i - 1 + b} + P_\varphi(w) \frac{K(\mathbf{z}_{-i})a + b}{i - 1 + b} \\
&\quad = \frac{n_w^{(\mathbf{w}_{-i})} - K_w(\mathbf{z}_{-i})a + (K(\mathbf{z}_{-i})a + b) P_\varphi(w)}{i - 1 + b}
\end{aligned} \tag{5}$$

where $K_w(\mathbf{z}_{-i})$ is the number of tables labeled with w in $\mathbf{z}_{-i}$. The joint distribution of a sequence of words $\mathbf{w}$ is given by

$$\begin{aligned}
P(\mathbf{w} \,|\, a, b, \varphi) &= \sum_{\mathbf{z}, \boldsymbol{\ell}} P(\mathbf{z} \,|\, a, b) P_\varphi(\boldsymbol{\ell}) \\
&= \sum_{\mathbf{z}, \boldsymbol{\ell}} \frac{\Gamma(1 + b)}{\Gamma(n + b)} \left(\prod_{k=1}^{K(\mathbf{z})-1} (ka + b) \right) \left(\prod_{k=1}^{K(\mathbf{z})} P_\varphi(\ell_k) \frac{\Gamma(n_k^{(\mathbf{z})} - a)}{\Gamma(1 - a)} \right)
\end{aligned} \tag{6}$$

where, as in Equation 3, the sums are over only those $\boldsymbol{\ell}$ and $\mathbf{z}$ such that $\ell_{z_i} = w_i$ for all i.

5. Relationship to Other Models

The two-stage framework outlined in the previous sections has three special cases that correspond to models that have previously been used in computational linguistics and statistics: the Dirichlet-multinomial model, the Dirichlet process, and the two-parameter Poisson-Dirichlet process. In each of the following subsections, we first present the relevant equivalency, and then show that it holds.

5.1 The Dirichlet-multinomial Model

Proposition 1 *A TwoStage*($CRP(\alpha)$, $Multinomial(\varphi)$) *model is equivalent to a* $Dirichlet(\alpha\varphi)$*-multinomial model.*

As mentioned in Section 1, several researchers have proposed Bayesian language models based on the Dirichlet-multinomial model (MacKay and Peto, 1994; Madsen et al., 2005), also known as the Dirichlet compound multinomial model (Elkan, 2006). In this model, words are drawn from a multinomial distribution:

$$w_i \,|\, \theta \;\sim\; \text{Multinomial}(\theta)$$

where $\theta = (\theta_1, \ldots, \theta_K)$. That is, for a corpus $\mathbf{w} = (w_1, \ldots, w_n)$ made up of a finite lexicon of words $(\ell_1, \ldots, \ell_K)$, $P(w_i = \ell_k \,|\, \theta) = \theta_k$ and $P(\mathbf{w} \,|\, \theta) = \prod_{k=1}^{K} \theta_k^{n_k}$, where n_k is the number of occurrences of ℓ_k in $\mathbf{w}$. In addition, the parameters θ are themselves drawn from a Dirichlet distribution with hyperparameters $\beta = (\beta_1, \ldots, \beta_K)$:

$$\theta \,|\, \beta \;\sim\; \text{Dirichlet}(\beta).$$

The Dirichlet distribution is defined as

$$\begin{aligned} P(\theta\,|\,\beta) &= c\prod_{k=1}^{K}\theta_k^{\beta_k-1} \\ \text{with } c &= \frac{\Gamma(\sum_{k=1}^{K}\beta_k)}{\prod_{k=1}^{K}\Gamma(\beta_k)} \end{aligned}$$

where $\beta_k > 0$. It is *conjugate* to the multinomial, meaning that the posterior distribution over the parameters θ given a corpus $\mathbf{w}$ takes on the same parametric form as the prior—specifically, a Dirichlet distribution with parameters $n_k + \beta_k$, where n_k is the number of occurrences of outcome k in $\mathbf{w}$:

$$\begin{aligned} P(\theta\,|\,\mathbf{w},\beta) &\propto P(\mathbf{w}\,|\,\theta)P(\theta\,|\,\beta) \\ &\propto \prod_{k=1}^{K}\theta_k^{n_k}\prod_{k=1}^{K}\theta_k^{\beta_k-1} \\ &= \prod_{k=1}^{K}\theta_k^{n_k+\beta_k-1}. \end{aligned}$$

Due to the conjugacy of the Dirichlet and multinomial distributions, it is easy to compute the predictive distribution of w_i conditioned on the values of the previously observed words $\mathbf{w}_{-i}$ and the hyperparameters β:

$$\begin{aligned} P(w_i = j\,|\,\mathbf{w}_{-i},\beta) &= \int_\Delta P(w_i = j\,|\,\theta)P(\theta\,|\,\mathbf{w}_{-i},\beta)\,d\theta \\ &= \frac{\Gamma(n+\sum_{k=1}^{K}\beta_k)}{\prod_{k=1}^{K}\Gamma(n_k+\beta_k)}\int_\Delta \theta_j^{n_j+\beta_j}\prod_{k\neq j}\theta_k^{n_k+\beta_k-1}\,d\theta \\ &= \frac{\Gamma(n+\sum_{k=1}^{K}\beta_k)}{\prod_{k=1}^{K}\Gamma(n_k+\beta_k)}\cdot\frac{\Gamma(n_j+\beta_j+1)\prod_{j\neq k}\Gamma(n_k+\beta_k)}{\Gamma(n+\sum_{k=1}^{K}\beta_k+1)} \\ &= \frac{n_j+\beta_j}{n+\sum_{k=1}^{K}\beta_k} \end{aligned} \tag{7}$$

where all counts are with respect to $\mathbf{w}_{-i}$, and Δ indicates the probability simplex: the set of values for $\theta >= 0$ such that $\sum_k \theta_k = 1$. The third line can be derived using elementary calculus and the definition of the Gamma function, but can also be seen to hold by noting that the Dirichlet distribution must sum to 1, and therefore

$$\int_\Delta \prod_{k=1}^{K}\theta_k^{\beta_k-1}\,d\theta = \frac{\prod_{k=1}^{K}\Gamma(\beta_k)}{\Gamma(\sum_{k=1}^{K}\beta_k)}$$

holds for any positive values of β_k. Comparing Equation 7 to Equation 2 reveals that the Dirichlet-multinomial model is a special case of our two-stage framework, with a CRP adaptor and a finite generator distribution. In particular, a TwoStage(CRP(α), Multinomial(φ)) model is equivalent to a Dirichlet-multinomial model with $\beta = \alpha\varphi$.

5.2 The Dirichlet Process

Proposition 2 *A TwoStage($CRP(\alpha), P_\varphi$) model (where P_φ has infinite support) is equivalent to a $DP(\alpha, P_\varphi)$ model.*

The Dirichlet process (DP; Ferguson, 1973), used in nonparametric Bayesian statistics, can be seen as an infinite-dimensional analogue of the symmetric Dirichlet distribution (a Dirichlet distribution where all β_i are equal).[5] Whereas each sample from a Dirichlet distribution returns a distribution θ over a finite set of outcomes, each sample from a Dirichlet process returns a distribution G over a countably infinite set of outcomes. The Dirichlet process has two parameters. The *base distribution*, G_0, (which may be discrete or continuous) determines the probability that any particular outcome will be in the support of G. The *concentration parameter*, α, determines the variance in the probabilities of those outcomes under G.

Typically, the Dirichlet process is used as a prior in infinite mixture models (Lo, 1984; Escobar and West, 1995; Neal, 2000; Rasmussen, 2000), where the concentration parameter determines the relative size of each mixture component, and the base distribution determines the probable parameters for the component distributions. Instead, we can use the Dirichlet process to define a simple language model as follows:

$$\begin{aligned} G \,|\, \alpha, P_\varphi &\sim \mathrm{DP}(\alpha, P_\varphi), \\ w_i \,|\, G &\sim G \end{aligned}$$

where $\mathrm{DP}(\alpha, P_\varphi)$ refers to a Dirichlet process with concentration parameter α and base distribution $G_0 = P_\varphi$. The corresponding graphical model can be seen in Figure 5. Just as we integrated out the θ parameters of the Dirichlet-multinomial model, we can integrate out the distribution G to obtain the following predictive distribution over words (Blackwell and MacQueen, 1973):

$$w_i \,|\, \mathbf{w}_{-i}, \alpha, P_\varphi \;\sim\; \frac{1}{i-1+\alpha} \sum_{j=1}^{i-1} \delta(w_j) + \frac{\alpha}{i-1+\alpha} P_\varphi$$

where $\delta(w_j)$ is a point mass at w_j. Rewriting the predictive distribution as a probability mass function reveals that the $\mathrm{DP}(\alpha, P_\varphi)$ model is equivalent to a $\mathrm{TwoStage}(\mathrm{CRP}(\alpha), P_\varphi)$ model:

$$P(w_i = w \,|\, \mathbf{w}_{-i}, \alpha, P_\varphi) \;=\; \frac{n_w^{(\mathbf{w}_{-i})} + \alpha P_\varphi(w)}{i-1+\alpha}.$$

Note that although G assigns probability to a countably infinite set of outcomes, the predictive distribution can be computed using only the frequencies of previous items and the base distribution P_φ.

It is worth pointing out that this DP language model can still technically be viewed as a mixture model, although a degenerate one. Each lexical item corresponds to a separate mixture component parameterized by its label ℓ_k and with a 0/1 likelihood function: $P(w_i \,|\, \ell_{z_i}) = I(w_i = \ell_{z_i})$. Thus, every data point in a single mixture component is identical. As a result, the potential applications of two-stage models and infinite mixture models are somewhat different. Infinite mixture models are

5. Specifically, as described by Neal (2000), the predictive distribution of a Dirichlet process mixture model can be obtained by taking the limit as k goes to infinity of a k-component finite mixture model with a symmetric Dirichlet prior.

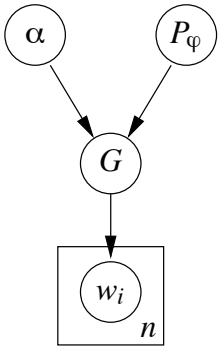


Figure 5: A graphical representation of the Dirichlet process language model.

more appropriate when the base distribution is a simple parameterized distribution (e.g., a Gaussian) and the clusters are expected to have some variability, whereas two-stage models are intended for cases where the base distribution may be more complex (e.g., a PCFG) but there is no variability between data points in a single cluster. An interesting area for future work lies in combining these two features to create models with complex base distributions as well as variability in the output of each cluster.

5.3 The Pitman-Yor Process

Proposition 3 *A TwoStage*$(PYCRP(a,b),P_\varphi)$ *model (where* P_φ *has infinite support) is equivalent to a* $PYP(a,b,P_\varphi)$ *model.*

Above, we described the Dirichlet process as the infinite dimensional analogue of the Dirichlet distribution. Another way of defining the Dirichlet process, which leads to the Pitman-Yor process as a generalization, is through the "stick-breaking" construction (Sethuraman, 1994). Recall that the distribution G produced by the Dirichlet process has two parts: a countably infinite set of possible outcomes drawn from the base distribution G_0, and weights assigned to those outcomes. The stick-breaking construction describes the distribution of these weights. Under this construction, we define a sequence of random variables $(V_1, V_2, \ldots)$, each following a Beta$(1,\alpha)$ distribution. The distribution of the weights from the Dirichlet process is the same as the distribution of the set of random variables in which the kth variable is defined to be $\prod_{j=1}^{k-1}(1-V_j)V_k$. Intuitively, this is the distribution we obtain over portions of a stick of length 1 when we break that stick into two pieces with sizes proportional to $(V_1, 1-V_1)$, then break the remainder into proportions $(V_2, 1-V_2)$, and so forth.

The stick-breaking construction for the Dirichlet process has just one parameter, α, but can be generalized through the introduction of a second parameter to define a new distribution, the Pitman-Yor process (PYP; Pitman, 1995; Pitman and Yor, 1997; Ishwaran and James, 2003). The stick-breaking construction for this two-parameter distribution is similar to that given above, except V_j is drawn from a Beta$(1-a, ja+b)$ distribution. Integrating over the weights in the two-parameter stick-breaking construction gives a predictive distribution that is similar to that of the Dirichlet process. More specifically, if we use $\mathbf{z} = (z_1, \ldots, z_n)$ to index the possible outcomes, we obtain the predictive distribution given in Equation 4, that is, the PYCRP. The relationship

between the PYCRP and the Pitman-Yor process is thus analogous to that between the CRP and the Dirichlet process: the PYCRP is the discrete distribution on partitions obtained by integrating over a distribution (the Pitman-Yor process) with weights generated from the two-parameter stick-breaking process. Therefore, just as TwoStage(CRP(α),P_φ) is equivalent to DP(α,P_φ), we have that TwoStage(PYCRP(a,b),P_φ) is equivalent to PYP(a,b,P_φ).

6. Effects of the Adaptor on Frequencies

We have now defined the two-stage modeling framework, shown that several Bayesian language models proposed elsewhere can be viewed as special cases of this framework, and presented two adaptors that generate power-law distributions over words. In this section, we consider how using these adaptors affects estimates of the parameters of the generator—the process that produces the underlying lexicon. In doing this, we return to our second motivating concern: the issue of how we might explain the damping of word frequencies, with the extreme case being reconciliation of models based on unique word *types* with those based on the observed frequencies of word *tokens*. We first discuss the general implications of using the CRP and PYCRP for estimating the parameters of the generator. We then explain how, in a TwoStage(PYCRP(a,b),P_φ) language model, the parameters of the PYCRP determine whether the parameters of the generator will be inferred based on word types, tokens, or some interpolation between the two. Finally, we show that this Pitman-Yor language model provides a principled explanation for the combination of token counts and type counts found in Kneser-Ney smoothing (Ney et al., 1994; Kneser and Ney, 1995).

6.1 Impact of the Adaptor on Frequencies used for Estimation

By introducing an adaptor into our model, we provide a route by which word tokens can appear in a corpus without having been directly produced by the generator. As a consequence, any estimate of the parameters of the generator will be based only on those tokens for which the generator is considered responsible, which will be a subset of the tokens in the corpus. The adaptor will thus have the effect of damping the frequencies from which the parameters of the generator are estimated, with the nature of this damping depending on the properties of the adaptor. In particular, we will show that using the CRP or PYCRP as adaptors is approximately equivalent to estimating the generator parameters from log transformed or inverse-power transformed token counts, respectively.

We can see how the choice of adaptor affects the frequencies used for estimating the parameters φ of the generator by considering how to estimate φ from the observed corpus $\mathbf{w}$.[6] In general, the parameters of generators can be estimated using Markov chain Monte Carlo methods, as we demonstrate in Section 7. Here, we will present some general results characterizing how the frequencies used in estimation are damped by using the CRP or PYCRP as an adaptor.

For either maximum-likelihood or Bayesian estimation, the relationship between φ and the corpus $\mathbf{w}$ is characterized by the likelihood $P(\mathbf{w}\,|\,\varphi)$ (where we suppress the conditioning on the adaptor parameters α or (a,b) here and in the remainder of this section). As noted in Section 4, the likelihood can be expressed as

$$P(\mathbf{w}|\varphi) = \sum_{\mathbf{z},\boldsymbol{\ell}} P(\mathbf{z}) P_\varphi(\boldsymbol{\ell}) \tag{8}$$

6. Under the interpretation of this model as a Pitman-Yor process mixture model, this is analogous to estimating the base measure G_0 in a Dirichlet process mixture model (e.g., Neal, 2000).

where the sum ranges over those $\mathbf{z}, \boldsymbol{\ell}$ pairs that generate $\mathbf{w}$.

Equation 8 makes it clear that the likelihood is affected not only by φ but also by $P(\mathbf{z})$. Nevertheless, we can still make some basic statements about the relationship between $\mathbf{w}$ and φ by considering the properties of the model as a whole. First, notice that the total frequency n_w of each word type w, as obtained by summing the counts on all tables labeled with that type, will equal the frequency of w in the corpus $\mathbf{w}$. Second, all that matters for the estimation of φ_w (the parameter(s) associated with word type w) is the number of tables labeled with w, since this value is equal to the number of times we have drawn w from the generator—all other instances of w are produced by the adaptor. Thus, we can gain insight into how estimates of φ are likely to be affected by the choice of adaptor by considering how the adaptor affects the relationship between the frequency of a word type and the number of tables labeled with that type.

The analysis given in the previous paragraph suggests that we want to compute the expected number of tables labeled with a given word type under different adaptors. This expectation can be computed from the posterior distribution on $\mathbf{z}$ and $\boldsymbol{\ell}$ given $\mathbf{w}$, which can be decomposed as $P(\mathbf{z}, \boldsymbol{\ell} \,|\, \mathbf{w}) = P(\boldsymbol{\ell} \,|\, \mathbf{z}, \mathbf{w}) P(\mathbf{z} \,|\, \mathbf{w})$. Note that $P(\boldsymbol{\ell} \,|\, \mathbf{z}, \mathbf{w})$ is equal to one if $\mathbf{z}$ and $\mathbf{w}$ are consistent with $\boldsymbol{\ell}$, and zero otherwise, so we can compute $P(\mathbf{z}, \boldsymbol{\ell} \,|\, \mathbf{w})$ by computing $P(\mathbf{z} \,|\, \mathbf{w})$ subject to this consistency constraint, that is, such that for each word type w, the appropriate n_w tokens of $\mathbf{w}$ are of type w. In order to simplify the mathematics, in the rest of this section we assume that each lexical item ℓ_j produced by the generator is independent and identically distributed (i.i.d.) given φ. That is, if $\boldsymbol{\ell} = (\ell_1, \ldots, \ell_K)$, then

$$P_\varphi(\boldsymbol{\ell}) \;=\; \prod_{\ell=1}^{K} P_\varphi(\ell_j).$$

First, we consider the CRP adaptor. In this case, we can obtain a good approximation to the expectation of the number of tables over the posterior distribution. The posterior distribution is exchangeable, so we can calculate the distribution over the number of lexical entries for a given word type w by imagining that the n_w instances of w are the first n_w tokens in our corpus. The posterior probability distribution for the seating assignment of the ith token is

$$P(z_i = k \,|\, w_i = w, \mathbf{z}_{-i}, \boldsymbol{\ell}(\mathbf{z}_{-i}), \varphi) \;=\; \frac{P(z_i = k, w_i = w \,|\, \mathbf{z}_{-i}, \boldsymbol{\ell}(\mathbf{z}_{-i}), \varphi)}{P(w_i = w \,|\, \mathbf{z}_{-i}, \boldsymbol{\ell}(\mathbf{z}_{-i}), \varphi)}$$

where

$$P(z_i = k, w_i = w \,|\, \mathbf{z}_{-i}, \boldsymbol{\ell}(\mathbf{z}_{-i}), \varphi) \;=\; \begin{cases} \frac{n_k^{(\mathbf{z}_{-i})}}{i-1+\alpha} \cdot I(\ell_k = w) & 1 \le k \le K(\mathbf{z}_{-i}) \\ \frac{\alpha}{i-1+\alpha} \cdot P_\varphi(w) & k = K(\mathbf{z}_{-i}) + 1 \end{cases}$$

and the denominator is given by Equation 2. Dividing through yields

$$P(z_i = k \,|\, w_i = w, \mathbf{z}_{-i}, \boldsymbol{\ell}(\mathbf{z}_{-i}), \varphi) \;=\; \begin{cases} \frac{n_k^{(\mathbf{z}_{-i})}}{n_w^{(\mathbf{w}_{-i})} + \alpha P_\varphi(w)} \cdot I(\ell_k = w) & 1 \le k \le K(\mathbf{z}_{-i}) \\ \frac{\alpha}{n_w^{(\mathbf{w}_{-i})} + \alpha P_\varphi(w)} \cdot P_\varphi(w) & k = K(\mathbf{z}_{-i}) + 1 \end{cases} \tag{9}$$

which we can now use to calculate the expected number of occupied tables (i.e., lexical entries) for a word type that occurs n_w times. Taking $w_i = w$ for $i = 1, \ldots, n_w$ means that $P_\varphi(w)$ is fixed

for all n_w decisions, and $\alpha P_\varphi(w)$ simply becomes a constant. Inspection of Equation 9 reveals that the posterior distribution on seating assignments for all tokens of type w is given by the CRP with parameter $\alpha P_\varphi(w)$ and a total of n_w customers. As Antoniak (1974) showed, the expected number of occupied tables in this case is $\alpha P_\varphi(w) \sum_{i=1}^{n_w} 1/(\alpha P_\varphi(w) + i - 1)$, or approximately $\alpha P_\varphi(w) \log \frac{n_w + \alpha P_\varphi(w)}{\alpha P_\varphi(w)} = O(\log(n_w))$.

Unfortunately, we cannot apply a similar analysis for use of the PYCRP adaptor. While the CRP treats each word type independently (that is, ignoring dependencies in the generator, in a CRP the number of tables associated with a word type is independent of the number of tables associated with other word types), this is not true for the PYCRP. As with the CRP, the probabilities defined by the generator multiply with the terms of the PYCRP when we generate a new table, so that

$$P(z_i = k \,|\, w_i = w, \mathbf{z}_{-i}, \boldsymbol{\ell}(\mathbf{z}_{-i}), \varphi) \;\propto\; \begin{cases} (n_k^{(\mathbf{z}_{-i})} - a) \cdot I(\ell_k = w) & 1 \le k \le K(\mathbf{z}_{-i}) \\ (K(\mathbf{z}_{-i})a + b) \cdot P_\varphi(w) & k = K(\mathbf{z}_{-i}) + 1. \end{cases} \tag{10}$$

However, this distribution does not take the form of another PYCRP. We can only say that the probability of choosing a new table under this distribution is bounded above by the probability of choosing a new table under a PYCRP with parameters a and $bP_\varphi(w)$. Ignoring the effect of the number of tables associated with other word types, we expect the number of tables to be less than the number produced by simply running a PYCRP$(a, bP_\varphi(w))$ over the n_w tokens of w. The expectation of the number of tables occupied after seating n_w customers increases as $O(n_w^a)$ for the PYCRP (Teh, 2006a), providing an upper bound on the number of tables we should expect a word with frequency n_w to produce when the PYCRP is used as an adaptor.[7]

These results provide a rough heuristic for understanding how using the CRP and the PYCRP as adaptors damps the frequencies from which the parameters of the generator are estimated: using the CRP and PYCRP as adaptors will be approximately equivalent to estimation from log and inverse-power transformed frequencies respectively. To evaluate the accuracy of these approximations, we conducted an experiment using a corpus derived from sections 0-20 from the Penn Wall Street Journal treebank (Marcus et al., 1993). The corpus consisted of 30,114 unique word types, with a total of 831,190 tokens. We then examined the parameter estimates produced by several two-stage models, varying both the generator and the adaptor.

In all models, the generator was taken to be a multinomial distribution over the full vocabulary, with a symmetric Dirichlet(β) prior. This generator was used because it is relatively generic, since any distribution over a discrete set ultimately grounds out in a multinomial, and because it allows us to parametrically explore the consequences of varying the strength of the prior. We used three different kinds of prior, corresponding to different settings of the hyperparameters: $\beta = 0.001, \beta = 1$, and $\beta \to \infty$. With $\beta = 0.001$, the prior prefers sparse multinomial distributions, which means that the number of tables assigned to w has a strong effect on the resulting estimate of φ_w: word types with many tables will tend to have high φ_w, while the sparse prior will push the estimated parameters for the remaining word types closer to zero. With $\beta = 1$, the prior is uniform over multinomials, which provides some regularization of the resulting estimates towards the uniform distribution. With $\beta \to \infty$, the prior forces the estimated parameters to be the uniform distribution over all word types, so the number of tables assigned to any given word type has no effect on the estimates. Note that the i.i.d. generator assumption made above only holds when $\beta \to \infty$.

7. We recently became aware of work by Buntine and Hutter (2010), in which the expected number of occupied tables in the PYCRP is derived. In future work, we hope to include this result in our analysis.

We combined these three generators with a total of fifteen different adaptors. For each generator, five models used a CRP adaptor with $\alpha = \{1, 10, 100, 1000, 10000\}$ and ten others used a PYCRP adaptor with $a = \{0.1, 0.2, \ldots, 1.0\}$ and $b = 1$. For each combination of generator and adaptor, a Markov chain Monte Carlo (MCMC) algorithm was used to calculate the expected number of occupied tables (from which the corresponding multinomial parameters were estimated) for each word in the corpus. Details of this algorithm are provided in Appendix A. Figure 6 displays the results: the expected number of occupied tables is shown, plotted as black dots, as a function of n_w for all combinations of generators and adaptors. To produce the figure, words were binned by frequency using bins that were uniform on a log scale, and the posterior mean of the number of occupied tables per word was averaged within bins.

Figure 6 also shows as gray lines the number of tables predicted by the heuristic approximations described above. The predictions for the CRP (left column) assume that the number of tables is equal to $\alpha P_\varphi(w) \sum_{i=1}^{n_w} 1/(\alpha P_\varphi(w) + i - 1)$, using the appropriate value of α but taking $P_\varphi(w)$ to be uniform over all words. The result is accurate when $P_\varphi(w)$ is constrained to be uniform (row (c); $\beta \to \infty$), but underestimates the number of tables for high frequency words when $P_\varphi(w)$ is itself more sensitive to the number of tables (rows (a) and (b); $\beta = 0.001$ or 1). The predictions for the PYCRP (right column) assume that the number of tables is equal to n_w^a, and provide a good approximate upper bound on the number of tables, with the actual numbers being closer to this upper bound when $P_\varphi(w)$ is free to become higher for high-frequency words (row (a)). In general, the heuristic of the number of tables increasing as $O(n_w^a)$ seems more accurate when n_w is small.

The influence of the prior on the number of tables per word under the two-stage model with PYCRP adaptor can be understood in terms of how the prior affects the difference between the posterior distribution on the number of tables and the simpler PYCRP we use to approximate it. The approximation PYCRP always assigns a higher probability to new tables than the posterior distribution, but the difference between the two for a word w will depend on the value of $P_\varphi(w)$, since the approximation assumes the probability of a new table is proportional to $K(\mathbf{z}_{-i})a + bP_\varphi(w)$, while the true probability is proportional to $K(\mathbf{z}_{-i})aP_\varphi(w) + bP_\varphi(w)$. With a prior that allows the number of tables to have a strong influence on $P_\varphi(w)$ (row (a)), the most frequent words will tend to have much larger values of $P_\varphi(w)$ than the less frequent words, so the difference between the approximation and the true distribution for the most frequent words will not be very great. However, when $P_\varphi(w)$ is constrained to be more uniform (rows (b) and (c)), the difference between the approximation and the true distribution for frequent words is much larger, so the approximation is bad.

A surprising feature of the PYCRP models is the nonmonotonic relationship between n_w and the true number of tables occupied by w, which is noticeable with higher values of a (except $a = 1$) in the bottom two plots on the right. This behavior is due to a confluence of factors, which include both the high value of a and the very large number of tables required to account for all the words in the corpus (a result of the large number of word types). Under these circumstances, when the total number of tokens of w is small, it is not possible to have a table with enough tokens of w so that the probability of placing another token of w on that table is much higher than placing the token on a new table.[8] Thus, the posterior distribution over the number of tables for w will be

8. Empirically, the total number of tables K inferred by our sampler is around 65,000, so the posterior probability of assigning a token of w to a new table with $a = .9$, $b = 1$, and uniform $P_\varphi(w)$ is roughly proportional to $((65{,}000)(0.9) + 1)\frac{1}{30{,}114} \approx 2$, wheras the probability of assigning w to an old table is proportional to $n_k^{(\mathbf{z}_{-i})} - 0.9$, which is actually less than two unless there are already more than two tokens on the old table. Even with five tokens already on the old table, the probability of using the old table is only about twice that of using the new table.

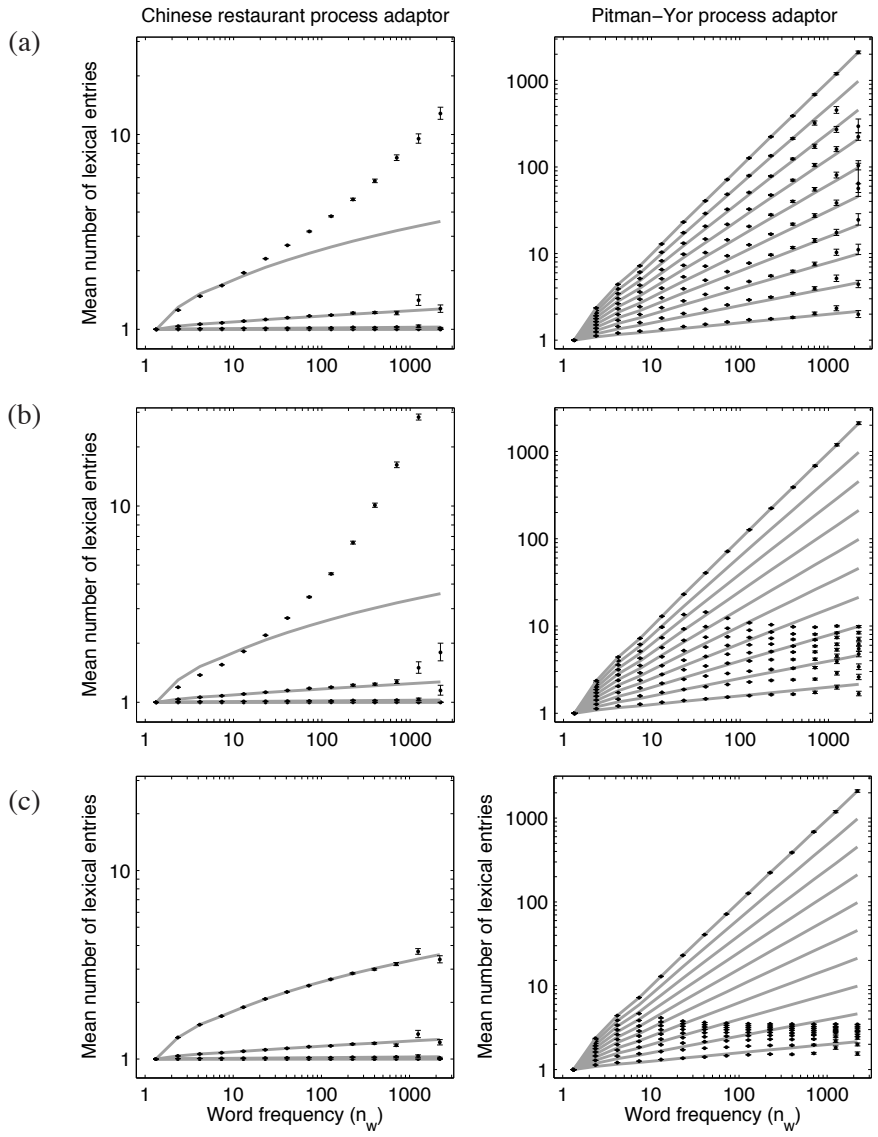


Figure 6: Mean number of occupied tables as a function of word frequency (n_w) under models of the text of sections 0-20 of the Penn Wall Street Journal treebank. The three rows of panels correspond to multinomial generators with Dirichlet(β) priors and (a) $\beta = 0.001$, (b) $\beta = 1$, and (c) $\beta \to \infty$. Each row shows the results of using the CRP (left) and PYCRP (right) as adaptors. All axes are on a log scale. Black dots and error bars show the empirical means and standard errors computed using MCMC; gray lines indicate approximations described in the text. The left-hand column shows results for the CRP with parameter $\alpha = \{1, 10, 100, 1000, 10000\}$ (from bottom to top; results for the first three are nearly identical and lie on top of each other in the graphs). The right-hand column shows results for the PYCRP with $b = 1$ and $a = \{0.1, 0.2, \ldots, 1.0\}$ (from bottom to top).

relatively uniform, and the average number of inferred tables will be large relative to the size of n_w. However, as n_w increases, it becomes possible to cluster the tokens so as to place a larger number on each table. There is a bigger win in probability for inferring a configuration with fewer tables when n_w is large, because this situation implies that more of the tokens were generated from old tables with many existing tokens, which have much higher probability than tables with zero or even a handful of existing tokens. Note that when P_φ is uniform (i.e., low for all words, as in row (c)), the probability of a new table is quickly outweighed by the probability of an existing table even with low counts. However, when β is small so that P_φ is estimated to be higher for more frequent words, the nonmonotonicity is not observed until n_w becomes much larger.

Overall, the theoretical and empirical results presented in this section suggest that our two-stage approach can provide a way to justify the use of logarithmic and inverse-power frequency damping in text processing applications. More significantly, this justification explains why adopting these schemes improves performance: it compensates for the kind of "rich-get-richer" processes that produce power-law distributions in natural language.

6.2 Types and Tokens

The most extreme kind of frequency damping is throwing away all but a single instance of each word type, and only keeping track of the unique word types that appear in the corpus. Just as we can explain other forms of frequency damping in terms of our two-stage framework, we can show that the TwoStage(PYCRP$(a,b),P_\varphi$) model provides a justification for the role of word types in formal analyses of natural language. We will now show that estimation schemes based upon type and token frequencies are special cases of the Pitman-Yor language model, corresponding to the extreme values of the parameter a. Values of a between these extremes identify estimation methods that interpolate between types and tokens.

Recall the joint distribution over words defined by the TwoStage(PYCRP$(a,b),P_\varphi$) model (from Equation 6):

$$P(\mathbf{w}\,|\,\varphi) \;=\; \sum_{\mathbf{z},\boldsymbol{\ell}} \frac{\Gamma(1+b)}{\Gamma(n+b)} \left(\prod_{k=1}^{K(\mathbf{z})-1} (ka+b) \right) \left(\prod_{k=1}^{K(\mathbf{z})} P_\varphi(\ell_k) \frac{\Gamma(n_k^{(\mathbf{z})}-a)}{\Gamma(1-a)} \right)$$

where the sum ranges over those $\mathbf{z}$ and $\boldsymbol{\ell}$ that generate $\mathbf{w}$. When $b=0$, this equation reduces to

$$\begin{aligned} P(\mathbf{w}\,|\,\varphi) &= \sum_{\mathbf{z},\boldsymbol{\ell}} \frac{\Gamma(1)}{\Gamma(n)} \cdot a^{K(\mathbf{z})-1}(K(\mathbf{z})-1)! \cdot \prod_{k=1}^{K(\mathbf{z})} P_\varphi(\ell_k) \frac{\Gamma(n_k^{(\mathbf{z})}-a)}{\Gamma(1-a)} \\ &= \sum_{\mathbf{z},\boldsymbol{\ell}} \frac{(K(\mathbf{z})-1)!}{(n-1)!} \cdot a^{K(\mathbf{z})-1} \cdot \prod_{k=1}^{K(\mathbf{z})} P_\varphi(\ell_k) \frac{\Gamma(n_k^{(\mathbf{z})}-a)}{\Gamma(1-a)}. \end{aligned} \tag{11}$$

The distribution $P(\mathbf{w}\,|\,\varphi)$ determines how the data $\mathbf{w}$ influence estimates of φ, so we will consider how $P(\mathbf{w}\,|\,\varphi)$ changes under different limits of a.

When $a \to 0$, the $a^{K(\mathbf{z})-1}$ term in Equation 11 causes the sum over $(\mathbf{z},\boldsymbol{\ell})$ to be dominated by the partition of customers with the smallest value of $K(\mathbf{z})$, that is, the fewest number of tables. Since seating arrangements are restricted so that $\ell_{z_i} = w_i$, the dominant arrangement contains exactly one table, and one occurrence of $P_\varphi(w)$, per word type w. Therefore estimates of φ will be based on word types.

When $a \to 1$, $a^{K(\mathbf{z})-1} \to 1$. If $n_k = 1$ then $\frac{\Gamma(n_k^{(\mathbf{z})}-a)}{\Gamma(1-a)} = 1$, but otherwise this term approaches 0. Therefore all terms in the sum approach 0 except for those where there is only a single token assigned to each table. In this case, $K(\mathbf{z}) = n$ and $\ell_k = w_k$, which means that P_φ is responsible for generating all the word tokens in the data. Estimates of φ will consequently be based on word tokens.

The extreme values of the a parameter in the PYCRP thus correspond to type-based inference ($a = 0$) or token-based inference ($a = 1$), while choosing other values of a between 0 and 1 provides a systematic way of smoothly interpolating between the type-based and token-based extremes.

6.3 Pitman-Yor Processes and Kneser-Ney Smoothing

In addition to justifying the role of types in formal analyses of language in general, using the PYCRP as an adaptor to create a Pitman-Yor language model can provide an explanation of the assumptions behind a specific scheme for combining token and type frequencies: Kneser-Ney smoothing. In this section, we outline the relationship between Kneser-Ney smoothing and the PYCRP, showing that the predictive distribution of the Kneser-Ney smoother can be viewed as an approximation to that of the Pitman-Yor language model. This relationship was first pointed out in a conference paper presenting preliminary versions of some of the results in this paper (Goldwater et al., 2006a), and then independently identified by Teh (2006a,b), who expanded on this observation and presented the first empirical comparisons of the two methods. We return to the results of empirical comparisons briefly below.

The Kneser-Ney smoother estimates the probability that a word token will belong to a particular type by combining type and token frequencies, and has proven particularly effective for n-gram models (Ney et al., 1994; Kneser and Ney, 1995; Chen and Goodman, 1998). To use an n-gram language model, we need to estimate the probability distribution over word types given a particular *history*, that is, the $n-1$ preceding tokens. Assume we are given a multiset $\mathbf{w}$ of N tokens that all share a common history, and we want to predict the next token, w_{N+1}, that will occur with that history. For example, the history might be *in the*, with $\mathbf{w}$ = (*house book way school house* ...). (We use a multiset rather than a vector because we care only about the counts of the word types in $\mathbf{w}$, not their ordering.) Assume that we also have H other multisets $\mathbf{w}^{(1)}, \ldots, \mathbf{w}^{(H)}$, each associated with one of H other histories. The interpolated Kneser-Ney (IKN) smoother (Chen and Goodman, 1998) makes the prediction

$$P(w_{N+1} = w \,|\, \mathbf{w}) = \frac{n_w^{(\mathbf{w})} - I(n_w^{(\mathbf{w})} > D)D}{N} + \frac{\sum_{w'} I(n_{w'}^{(\mathbf{w})} > D)D}{N} \frac{\sum_h I(w \in \mathbf{w}^{(h)})}{\sum_{w'} \sum_h I(w' \in \mathbf{w}^{(h)})} \qquad (12)$$

where D is a "discount factor" specified as a parameter of the model, the sum over h includes $\mathbf{w}$, and we have suppressed the dependence on $\mathbf{w}^{(1)}, \ldots, \mathbf{w}^{(H)}$.

We can define a two-stage model that approximates the Kneser-Ney smoother by assuming that each $\mathbf{w}^{(h)}$ is produced by a two-stage restaurant with a PYCRP adaptor (i.e., a separate restaurant for each history), where all the restaurants share the same generator, parameterized by φ. We assume φ is a multinomial distribution, which we estimate using maximum-likelihood estimation. Under this model, the probability that token w_{N+1} takes on the value w given $\mathbf{w}$ and φ is

$$P(w_{N+1} = w \,|\, \mathbf{w}, \varphi) = \sum_{\mathbf{z}} P(w_{N+1} = w | \mathbf{w}, \mathbf{z}, \varphi) P(\mathbf{z} | \mathbf{w}, \varphi)$$

where $\mathbf{z}$ is the seating assignment for $\mathbf{w}$, and $P(w_{N+1} = w|\mathbf{w}, \mathbf{z}, \varphi)$ is equivalent to $P(w_{N+1} = w|\boldsymbol{\ell}(\mathbf{z}), \mathbf{z}, \varphi)$, given by Equation 5. Substituting in Equation 5 and assuming $b = 0$, this becomes

$$\begin{aligned}
&P(w_{N+1} = w \,|\, \mathbf{w}, \varphi) \\
&= \sum_{\mathbf{z}} \frac{n_w^{\mathbf{w}} - K_w(\mathbf{z})a + K(\mathbf{z})aP_\varphi(w)}{N} P(\mathbf{z}|\mathbf{w}, \varphi) \\
&= \sum_{\mathbf{z}} \frac{n_w^{\mathbf{w}} P(\mathbf{z}|\mathbf{w}, \varphi)}{N} - \sum_{\mathbf{z}} \frac{K_w(\mathbf{z})aP(\mathbf{z}|\mathbf{w}, \varphi)}{N} + \sum_{\mathbf{z}} \frac{K(\mathbf{z})aP_\varphi(w)P(\mathbf{z}|\mathbf{w}, \varphi)}{N} \\
&= \frac{n_w^{\mathbf{w}}}{N} - \sum_{\mathbf{z}} \frac{K_w(\mathbf{z})aP(\mathbf{z}|\mathbf{w}, \varphi)}{N} + \sum_{\mathbf{z}} \frac{\sum_{w'} K_{w'}(\mathbf{z})aP_\varphi(w)P(\mathbf{z}|\mathbf{w}, \varphi)}{N} \\
&= \frac{n_w^{\mathbf{w}} - E_{\mathbf{z}}[K_w(\mathbf{z})]\, a}{N} + \frac{\sum_{w'} E_{\mathbf{z}}[K_{w'}(\mathbf{z})]\, a}{N} P_\varphi(w) \qquad (13)
\end{aligned}$$

where $E_{\mathbf{z}}[K_w(\mathbf{z})] = \sum_{\mathbf{z}} K_w(\mathbf{z})P(\mathbf{z}|\mathbf{w}, \varphi)$, and $K_w(\mathbf{z})$ is the number of tables with label w under the seating assignment $\mathbf{z}$. The other histories enter into this expression via φ. Since all the $\mathbf{w}^{(h)}$ are assumed to be produced from a single set of parameters φ, the maximum-likelihood estimate of $P_\varphi(w)$ will approach

$$P_\varphi(w) = \frac{\sum_h I(w \in \mathbf{w}^{(h)})}{\sum_{w'} \sum_h I(w' \in \mathbf{w}^{(h)})}$$

as a approaches 0, since only a single instance of each word type in each context will contribute to the estimate of φ. Substituting this value of $P_\varphi(w)$ into Equation 13 reveals the correspondence to the Kneser-Ney smoother (Equation 12). The only difference is that the constant discount factor D is replaced by $aE_{\mathbf{z}}[K_w(\mathbf{z})]$, which will increase slowly as n_w increases.

Note that the formulation given above is very general in that we do not specify a particular generator model P_φ. However, to complete the correspondence with IKN n-gram smoothing, we can assume that the generator for the model that computes the distribution over word types conditioned on a history of size n is another two-stage PYCRP model that computes probabilities conditioned on histories of size $n-1$. The recursion bottoms out with a uniform distribution over the W word types in the vocabulary, $P_0(w) = 1/W$. This *hierarchical* Pitman-Yor language model (Teh, 2006b) is analogous to the hierarchical Dirichlet process introduced by Teh (2006a). Intuitively, we can imagine a separate restaurant for each history of size n, where the counts in that restaurant correspond to the distribution of word tokens given that history. If a customer sits at a new table in one of these restaurants, the label on that table is distributed according to the counts in a "backoff" restaurant with history size $n-1$. All restaurants with the same final $n-1$ history words will share the same backoff restaurant.

As noted above, there are slight differences between the predictions of this Pitman-Yor language model and IKN smoothing due to the replacement of the constant discount factor D in IKN with an expression that increases as a function of n_w. Interestingly, *modified Kneser-Ney* (MKN) smoothing (Chen and Goodman, 1998) also replaces the single constant D in IKN with a small set of D values that increase as a function of n_w (Chen and Goodman 1998 use three values, for $n_w = 1$, 2, and 3 or more). MKN was introduced by Chen and Goodman (1998) as an alternative to IKN that was shown to work better in practice. So it has been known for a number of years that increasing D with n_w seems to provide better predictions, and initial experiments with the Pitman-Yor language model (Teh, 2006a,b) did not show improvements over MKN (although they did show improvements over

IKN). However, these experiments were performed on a relatively small corpus of text (16 million words of newswire). More recently, Huang and Renals (2010) developed a parallel approximate training algorithm for the Pitman-Yor language model and performed a more thorough set of experiments comparing IKN, MKN, and the Pitman-Yor language model within a speech recognition system. The models were trained on a large corpus of conversational speech (200 million words) and evaluated on perplexity and word error rate. The Pitman-Yor model achieved the best results on both measures, and gains over the other two models became larger as corpus size increased. So although empirical investigation was sufficient to develop a very close approximation to the Pitman-Yor language model, discovery of the true model has nevertheless led to better language models in practice.

7. Types and Tokens in Modeling Morphology

Our attempt to develop statistical models of language that generically produce power-law distributions was motivated by the possibility that models that account for this statistical regularity might be able to learn linguistic information better than those that do not. Our two-stage language modeling framework allows us to create exactly these sorts of models, with the generator producing individual lexical items, and the adaptor producing the power-law distribution over words. In this section, we show that adding a PYCRP adaptor to a simple generative model for morphology can vastly improve unsupervised learning of the morphological structure of English, and we explore the effects of varying the PYCRP parameters in this task. Morphology provides a particularly interesting case for testing our model, as it is one context in which formal linguists focus on accounting for the appearance of word types (e.g., Pierrehumbert, 2003), while computational linguists have typically developed supervised models based on the token frequencies of those words (e.g., Hakkani-Tür et al., 2002). Interestingly, previous work on *unsupervised* learning of morphology often ignores token frequencies, instead using word types as input (Goldsmith, 2001, 2006; Snover and Brent, 2003; Monson et al., 2004).[9] This fact suggests that the additional information provided by token frequencies may actually be harmful for learning morphology using standard models. Indeed, the results we report below support this hypothesis; we provide some possible explanations in the Section 8.1.2, where we discuss the results of our first set of experiments.

Previous morphology learning models have sidestepped the problems presented by token frequencies by simply ignoring them and using only a list of unique word types as input instead. It is worth reiterating here that our own two-stage model can be made to behave equivalently: with appropriate values of the PYCRP parameters (specifically, $a = b = 0$), our two-stage model assigns every token of the same word type to the same table, so that the parameters of the generator model (here, the morphology model) are inferred based on a list of unique word types. The result is equivalent to that of a model consisting only of the generator, where the input is a list of word types, as in the systems mentioned above. However, our full two-stage model is more flexible than these other systems. First, by choosing different adaptor parameters, different damping regimes can be achieved. Although these too could be simulated through different preprocessing schemes (e.g., taking logs of token frequencies rather than removing frequencies entirely), our model is more

9. Descriptions of Goldsmith's Linguistica system (Goldsmith, 2001, 2006) do not mention that frequencies are discarded before analysis. However, the version of the program we downloaded from `http://humanities.uchicago.edu/faculty/goldsmith` produced the same results when run on a full corpus as when run on a list of the unique word types in the corpus.

promising precisely because it can achieve the effects of damping while leaving the actual input frequencies unchanged. Thus, unlike previous models, ours can be used to learn directly from a corpus without preprocessing. This makes it possible to extend the model to incorporate additional information available from the corpus but not from a word list, such as contextual information. The experiments presented here are intended only to explore the effects of different parameter values, and do not take immediate advantage of this difference between our model and previous unsupervised systems. However, recent work using adaptor grammars has suggested some ways in which context can be incorporated into models based on the two-stage framework, for example by learning collocations between words at the same time as sub-word units (Johnson, 2008a; Johnson and Goldwater, 2009). Another example of using contextual information might be a hidden Markov model for part-of-speech tagging, where the standard multinomial emission distributions could be replaced with our morphology model, so that the learned part-of-speech classes would be informed both by corpus context and morphological structure. It is difficult to see how this kind of joint learning could take place in a probabilistic model requiring one instance of each word type as input.

7.1 A Lexicon Generator for Morphology

Many languages contain words built up of smaller units of meaning, or *morphemes*. These units can contain lexical information (as stems) or grammatical information (as affixes). For example, the English word *walked* can be parsed into the stem *walk* and the past-tense suffix *-ed*. Knowledge of morphological structure enables language learners to understand and produce novel wordforms, and is important for many natural language processing tasks in morphologically rich languages (Collins et al., 1999; Larkey et al., 2002; Cowan and Collins, 2005; Koehn and Hoang, 2007).

As a basic model of morphology, we assume that each word consists of a single stem and (possibly empty) suffix, and belongs to some inflectional class. Each class is associated with a stem distribution and a suffix distribution. We assume that stems and suffixes are independent given the class, so the joint probability of generating a particular class c, stem t, and suffix f is defined as

$$P(c,t,f) = P(c)P(t\,|\,c)P(f\,|\,c)$$

where the distributions on the right hand side are all assumed to be multinomial, generated from symmetric Dirichlet priors with hyperparameters κ, τ, and ϕ respectively. So far, we have been assuming that the generator in a two-stage model is a distribution over lexical items that are strings. However, in this morphology model, the generator produces analyses of strings (class, stem, suffix), rather than the strings themselves. We will therefore distinguish between the label ℓ_k on each table, which we continue to assume is a string, and the analysis of that label $A(\ell_k)$, which is an object produced by the generator. We can, if we wish, compute the probability of a label regardless of its analysis as

$$P(\ell) = \sum_{(c,t,f)} I(\ell = t.f)P(c)P(t\,|\,c)P(f\,|\,c)$$

where $t.f$ is the concatenation of t and f, and $I(.)$ is an indicator function taking on the value 1 when its argument is true, and 0 otherwise.

Our generator model for morphology is inspired by the model described by Goldsmith (2001), and is intended to encode two basic linguistic intuitions. The first is that different morphological classes contain different sets of stems and suffixes. Also, although stems and suffixes are not truly independent even within a morphological class, morphological boundaries do tend to coincide with

points of low predictability in a string of phonemes or characters (Harris, 1955). That is, there is greater independence between stems and suffixes than between other possible substrings. Another way of looking at this is that, if we know that, for example, past and present tense verbs are each relatively common, then if we see a particular verb very frequently in the past tense, we would expect to see it very frequently in the present tense as well (Yarowsky and Wicentowski, 2000).

We also note two important differences between our model and that of Goldsmith. First, Goldsmith's model is recursive (i.e., a word stem can be further split into a smaller stem plus suffix), which makes it better able to deal with complex morphology than the model presented here. However, the simplifying assumption of a single stem and suffix per word is often sufficient for English inflectional morphology. We emphasize that our primary goal here is to illustrate the effects of the generator-adaptor framework rather than to develop a state-of-the-art morphology learning system.

The second difference between Goldsmith's model and our own is that Goldsmith's model assumes that all occurrences of each word type have the same analysis. The model here allows different tokens with the same observed form to have different analyses when $a > 0$ or $b > 0$. This feature could be important for representing homonymous words with different morphological analyses.

7.2 Gibbs Sampler

Our goal in defining this morphology model is to be able to automatically infer the morphological structure of a language. Since our model is exchangeable, this can be done using Gibbs sampling, a standard Markov chain Monte Carlo method (Gilks et al., 1996). In Markov chain Monte Carlo, variables in the model are repeatedly sampled, with each sample conditioned on the current values of all other variables in the model. This process defines a Markov chain whose stationary distribution is the posterior distribution over model variables given the input data.

Rather than sampling all the variables in our two-stage model simultaneously, our Gibbs sampler alternates between sampling the variables in the generator and those in the adaptor (here, a PYCRP). Our algorithm iterates over the following two steps, as illustrated in Figure 7:

1. Fix the assignment $\mathbf{z}$ of words to tables, and sample a new morphological analysis $A(\ell_k)$ for the label on each table.

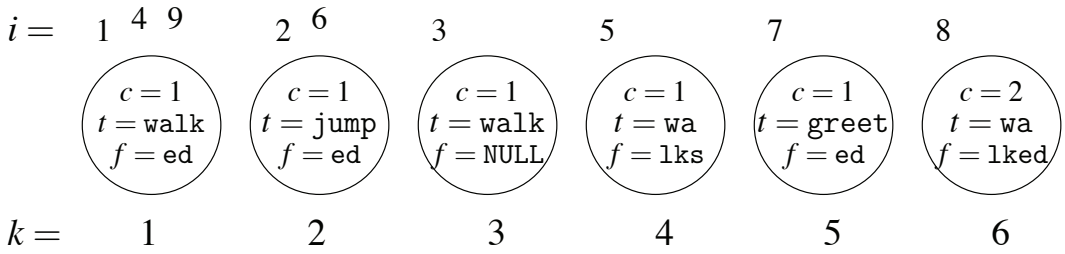
2. Fix the morphological analyses $A(\boldsymbol{\ell})$ of the labels, and sample a new table assignment z_i for each word token w_i.

In Step 1, we compute the probability distribution over analyses of the current label $A(\ell_k)$ conditioned on the analyses of all other labels $A(\boldsymbol{\ell}_{-k})$:

$$\begin{aligned} P(A(\ell_k) = (c,t,f) \,|\, A(\boldsymbol{\ell}_{-k}), \kappa, \tau, \phi) & \\ \propto \;& I(\ell_k = t.f) \cdot P(c,t,f \,|\, A(\boldsymbol{\ell}_{-k}), \kappa, \tau, \phi) \\ =\; & I(\ell_k = t.f) \cdot P(c \,|\, \mathbf{c}_{-i}, \mathbf{z}, \kappa) \cdot P(t \,|\, \mathbf{t}_{-i}, \mathbf{c}, \mathbf{z}, \tau) \cdot P(f \,|\, \mathbf{f}_{-i}, \mathbf{c}, \mathbf{z}, \phi) \\ =\; & I(\ell_k = t.f) \cdot \frac{m_c + \kappa}{m + \kappa C} \cdot \frac{m_{t,c} + \tau}{m_c + \tau T} \cdot \frac{m_{f,c} + \phi}{m_c + \phi F} \end{aligned} \tag{14}$$

where the notation $\mathbf{x}_{-i}$ is now used to indicate $(x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_n)$ (by exchangeability, we can nevertheless treat x_i as though it is the last of the n variables when computing probabilities); C, T, and F are the total possible number of classes, stems, and suffixes; and m_x is the number of tables in

(a)



(b)

$$
\begin{aligned}
A(\ell_1) = (1, \texttt{NULL}, \texttt{walked}) \,|\, A(\boldsymbol{\ell}_{-1}), \kappa, \tau, \phi) &\propto \frac{5+\kappa}{6+2\kappa} \cdot \frac{\tau}{5+21\tau} \cdot \frac{\phi}{5+25\phi} \\
A(\ell_1) = (1, \texttt{w}, \texttt{alked}) \,|\, A(\boldsymbol{\ell}_{-1}), \kappa, \tau, \phi) &\propto \frac{5+\kappa}{6+2\kappa} \cdot \frac{\tau}{5+21\tau} \cdot \frac{\phi}{5+25\phi} \\
A(\ell_1) = (1, \texttt{wa}, \texttt{lked}) \,|\, A(\boldsymbol{\ell}_{-1}), \kappa, \tau, \phi) &\propto \frac{5+\kappa}{6+2\kappa} \cdot \frac{1+\tau}{5+21\tau} \cdot \frac{\phi}{5+25\phi} \\
&\cdots \\
A(\ell_1) = (2, \texttt{wa}, \texttt{lked}) \,|\, A(\boldsymbol{\ell}_{-1}), \kappa, \tau, \phi) &\propto \frac{1+\kappa}{6+2\kappa} \cdot \frac{1+\tau}{1+21\tau} \cdot \frac{1+\phi}{1+25\phi} \\
&\cdots \\
A(\ell_1) = (2, \texttt{walked}, \texttt{NULL}) \,|\, A(\boldsymbol{\ell}_{-1}), \kappa, \tau, \phi) &\propto \frac{1+\kappa}{6+2\kappa} \cdot \frac{\tau}{1+21\tau} \cdot \frac{\phi}{1+25\phi}
\end{aligned}
$$

(c)

$$
\begin{aligned}
P(z_1 = 1 \,|\, w_i = w, \mathbf{z}_{-i}, \boldsymbol{\ell}(\mathbf{z}_{-i}), \varphi, a, b) &\propto 2 - a \\
P(z_1 = 6 \,|\, w_i = w, \mathbf{z}_{-i}, \boldsymbol{\ell}(\mathbf{z}_{-i}), \varphi, a, b) &\propto 1 - a \\
P(z_1 = 7 \,|\, w_i = w, \mathbf{z}_{-i}, \boldsymbol{\ell}(\mathbf{z}_{-i}), \varphi, a, b) &\propto (6a + b) P_\varphi(\texttt{walked})
\end{aligned}
$$

Figure 7: An example illustrating our Gibbs sampler. In this example, the corpus $\mathbf{w}$ = (walked, jumped, walk, walked, walks, jumped, greeted, walked, walked), and initially $\mathbf{z}$ = (1, 2, 3, 1, 4, 2, 5, 6, 1). (a) illustrates the current seating arrangement, with numbers above each table indicating the indices i of customers seated there and the number below each table indicating the index k of the table. The morphological analysis associated with each table is also shown. T and F for this corpus (the total number of possible stems and suffixes) are 21 and 25, and we let $C = 2$. To complete a full Gibbs iteration, we first resample the analyses, and then the table assignments. In this case, we start by removing the current analysis of walked on table 1 (and its associated counts), and computing the probability of each of the 14 possible new analyses, as shown in (b). We sample from this distribution, replace the new analysis on table 1 (incrementing the associated counts), and repeat for the remaining five tables. Then, we sample new values for $z_1 \ldots z_9$ in a similar fashion. (c) shows the computations for z_1, which is restricted to taking on the values 1, 6, or 7 (a new table) because only these tables may be labeled with walked.

$A(\boldsymbol{\ell}_{-z})$ whose label includes x. (We use m to distinguish these counts over labels from the n counts over tokens.) The last line is obtained by integrating over the multinomial parameters for the classes, stems, and suffixes as in Equation 7; for example, $P(c\,|\,\mathbf{c}_{-i},\mathbf{z},\kappa) = \int P(c\,|\,\theta_c)P(\theta_c\,|\,\mathbf{c}_{-i},\mathbf{z},\kappa)d\theta_c$ where θ_c are the parameters of the multinomial distribution over classes.

In the experiments presented here, C is fixed empirically and T and F are determined for each set of input data by computing the number of possible segmentations of the words in the data into stems and suffixes (i.e., determining all the prefix and suffix strings for those words; the empty string is considered as a possible stem as well as a possible suffix).

In Step 2 of our sampler, we compute the distribution over table assignments z_i for the ith word token using Equation 10, repeated below with the conditioning adaptor parameters included:

$$P(z_i = k\,|\,w_i = w,\mathbf{z}_{-i},\boldsymbol{\ell}(\mathbf{z}_{-i}),a,b,\varphi) \propto \begin{cases} (n_k^{(\mathbf{z}_{-i})} - a)\cdot I(\ell_k = w) & 1 \le k \le K(\mathbf{z}_{-i}) \\ (K(\mathbf{z}_{-i})a + b)\cdot P_\varphi(w) & k = K(\mathbf{z}_{-i})+1 \end{cases}$$

where $P_\varphi(w)$ is found using Equation 14 by summing over all possible analyses.

Note that in Step 2, tables may appear or disappear, which will cause the label counts to change. When a table is removed, the class, stem, and suffix counts of its label are decremented. When a new table is added, a morphological analysis is chosen at random according to Equation 14, and the appropriate counts are incremented.

8. Experiments

In this section, we use the simple morphology model defined above as an example to demonstrate that applying an appropriate adaptor can significantly improve the learning of linguistic structure. We also examine how the choice of parameters in the PYCRP affects learning behavior. We perform two experiments, one using verbs in standard written form from a corpus of newspaper text, and the other using all words from a corpus of phonemically transcribed child-directed speech. In each experiment, evaluations were performed on a single sample taken after 1000 iterations of our Gibbs sampler, with $C = 6$ classes, $\kappa = .5$ and $\tau = \phi = .001$.[10] For the PYCRP parameters, we fixed $b = 0$ and experimented with values of a between 0 and 1.[11]

8.1 Experiment 1: Verbs

We begin by describing the data and evaluation method used in this experiment, followed by the experimental results.

8.1.1 DATA AND EVALUATION

We prepared a data set consisting of English verbs in written form from the Penn Wall Street Journal treebank (Marcus et al., 1993), a corpus of hand-tagged and parsed text from the Wall Street Journal. Using the part-of-speech tags, we extracted all the verbs from sections 0-21 of the corpus, which yielded 137,997 tokens belonging to 7,761 types. This list of verbs served as the input to the

10. Although we fixed the values for the hyperparameters in our experiments, all of our models can be extended to include prior distributions over the hyperparameters. In that case the hyperparameter values can be inferred by sampling. (West, 1992).

11. Technically, setting $a = 0$ and $b = 0$ leads to undefined results, but algorithmically one can simulate $\lim_{a\to 0}$ by using exactly one table for each word type, which is what we did.

morphological segmentation system. In this data set, the total number of unique prefix strings T is 22,396, and the total number of unique suffix strings F is 21,544.

To create a gold standard for evaluation, we automatically segmented each verb in the input corpus using heuristics based on its part-of-speech tag and spelling. For example, verbs tagged as VBD (past tense) or VBN (past participle) and ending in *-ed* were assigned a morpheme boundary before the *-ed*, while most verbs tagged as VBZ (third person present singular) and ending in *-s* were assigned a boundary before the *-s*. (The VBZ forms *does* and *goes*, as well as forms ending in *-xes* or *-ches*, such as *mixes*, were assigned a boundary before *-es* instead.) Potentially irregular forms such as past participles ending in *-n* were examined by hand to ensure correct segmentation.

It is important to note that any choice of segmentation will lead to some inconsistencies due to spelling rules that insert or delete characters before certain endings. The segmentation we used prefers consistency among suffixes rather than stems when there is a conflict. That is, suffixes will be the same across words such as *jump.ed* and *stat.ed*, or *jump.s* and *state.s*, but the stems in *stat.ed* and *state.s* will be different.

Given the gold standard analysis for each word and a sample analysis from our algorithm, segmentation accuracy was computed in two different ways. First, for each word type, the most frequent suffix for that type (in the sampled hypothesis) was determined and counted once to evaluate the proportion of types with each suffix. Second, since different tokens of the same type may be assigned different analyses, the proportion of word tokens with each suffix is also displayed. This analysis gives more weight to the results of frequent words, and also takes into account any uncertainty in the model (although in fact less than 1.5% of types have multiple analyses for any value of a).

8.1.2 RESULTS

As a model for learning morphology, our generator by itself is not very effective. Only 55.4% of word types and 62.2% of word tokens are segmented correctly. For comparison, baseline accuracy for a system that always leaves words unsegmented is 30.7% for types and 57.1% for tokens. It turns out that for most words, the segmentation identified by the generator model is actually the same as the unsegmented baseline, as illustrated in Figure 8. In other words, the model simply memorizes full words rather than splitting off (non-empty) suffixes. This is particularly true of frequent words, which is why token accuracy is so similar for the baseline and the generator model.

One might expect that the sparse Dirichlet priors used in our generator, which encourage fewer total stems and suffixes overall, would push the system towards a more parsimonious solution (i.e., fewer complete memorized wordforms). We know that the priors do have some effect, because the maximum-likelihood solution for this model is the baseline described above, with each word left unsegmented. However, even with much stronger Dirichlet priors than the ones reported here, the performance of the generator model alone is underwhelming. The reason is twofold. First, our generator model assumes complete independence between stem and suffix probabilities given the class of the word. In reality, stem and suffix probabilities are not completely independent (e.g., *announce* tends to occur more often with *-ed* than does *head*). As the amount of data for a particular verb accumulates, any deviation from independence becomes more apparent, and the model resolves this by memorizing entire words rather than segmenting them. This tendency is compounded by a second factor, which is that the most frequent words in the data are almost all irregular (e.g., *rise*/*rose*). Since our model deals only with segmentation, irregular words must be analyzed as

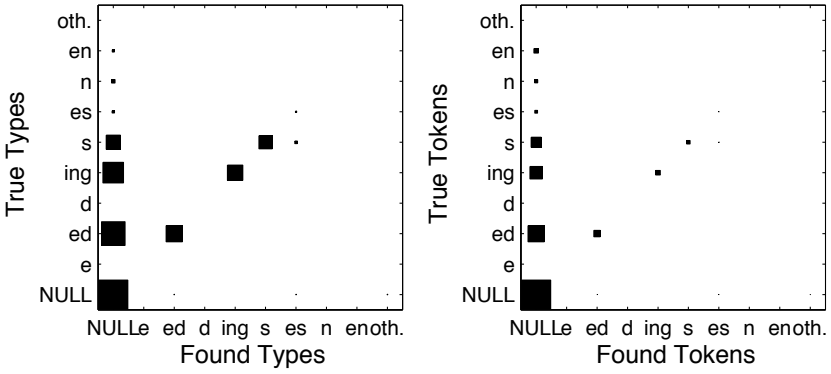


Figure 8: Confusion matrices for the morphological generator model alone (equivalent to the two-stage morphology model with $a = 1$) on the verb data set. The area of a square at location (i, j) is proportional to the number of word types (left) or tokens (right) with true suffix i and found suffix j.

having empty suffixes. This raises the overall probability of empty suffixes, making the model less likely to propose non-empty suffixes even when these are appropriate.

These issues may seem particular to our very simple model, or to the problem of morphological learning in English. However, we would argue that they are far more general. While it is true that English verbal morphology is notorious for its large number of irregular verbs, irregularity is found to varying degrees across all languages and types of linguistic structure. For example, in English, idiomatic expressions such as *X has got it made* or *X is fit to be tied*[12] can be viewed as syntactically irregular forms, in the sense that they both use the passive construction but have no corresponding active version. And, like other idioms, they also have irregular (that is, non-compositional) semantics. Importantly, the relationship between frequency and regularity observed in the current experiment (i.e., that irregular forms tend to be the most frequent) seems to be a very general property of language (Greenberg, 1966; Bybee, 1985). Together with the power-law distribution of linguistic forms, this fact implies that irregular forms will often dominate the input to statistical learning systems, which in turn may cause significant problems for an unsupervised model that does not take these facts into account.

One solution to these problems would be to simply change the input by removing repeated tokens of each type, that is, to present the system with only a list of unique word types. As discussed in the introduction to this section, many previous morphology learning systems have taken this approach. Instead, we address the problem by applying our two-stage framework, adding a PYCRP adaptor to our generator model. With this approach, we find that for a wide range of a, from 0 up to about 0.6 or 0.7, results are stable and considerably better than when using the generator model alone (or, equivalently, the 2-stage model with $a = 1$). Accuracy scores are shown in Figure 9, and confusion matrices for the model with $a = 0.6$ are shown in Figure 10. Given our discussion above, it should be no surprise that the better performance is due to the system finding more non-empty

12. These examples are due to Jackendoff (2002).

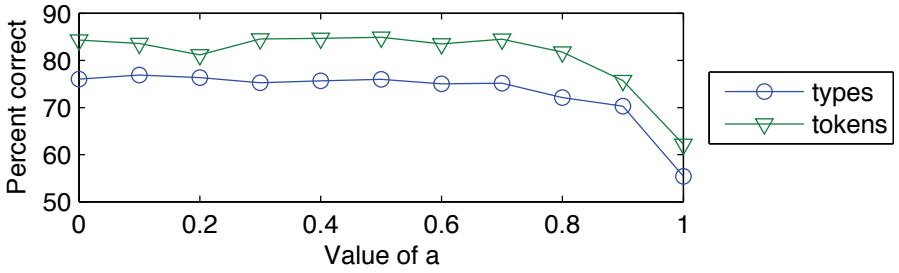


Figure 9: Percentage of verb types and tokens assigned the gold standard analysis.

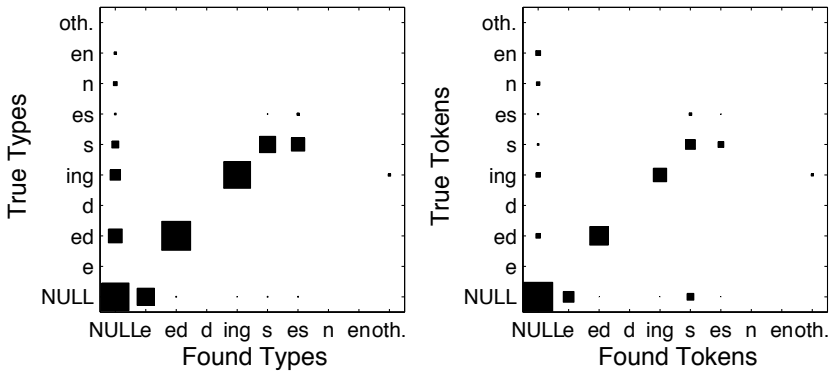


Figure 10: Confusion matrices for the 2-stage morphology model with $a = 0.6$.

suffixes overall. This is illustrated both in the confusion matrices and in Figure 11, which shows the true distribution of words with each suffix and the distribution found by the two-stage system for various values of a. Again, we see that the distribution is stable for $0 \leq a \leq 0.7$. For $a > 0.7$, empty suffixes begin to take over, causing performance to drop. Figure 12 indicates that the average number of tables per word type for $a \leq .7$ rises slowly from one to about four, whereas higher values of a cause a sharp increase in the average number of tables per type, up to almost 18. It is this increase that seems to be problematic for learning.

Finally, we provide a summary of the final sample in each of two runs of our sampler, with $a = 0.1$ and $a = 0.6$, in Table 1. An interesting feature seen in Table 1(b) is that the system has created a separate class for verbs with irregular past tense forms (second from the top). Also, in both runs, the system frequently hypothesizes analyses in which stem identity is kept constant across forms (as in *stat.e, stat.ing, stat.ed, stat.es*), whereas the gold standard maintains suffix identity (*state, stat.ing, stat.ed, state.s*). This leads the system to assume *-e* and *-es* suffixes where the gold standard has *NULL* and *-s*, and to place stems ending in *e* in separate classes from the other stems. This kind of problem is common to many morphological learning systems, and cannot be solved with a purely concatenative approach to morphology. It is also worth noting that, if the goal is to achieve a segmentation with the fewest total number of stems plus suffixes (minimizing storage cost) then the choice of segmentation taken by the system is actually better than the gold standard,

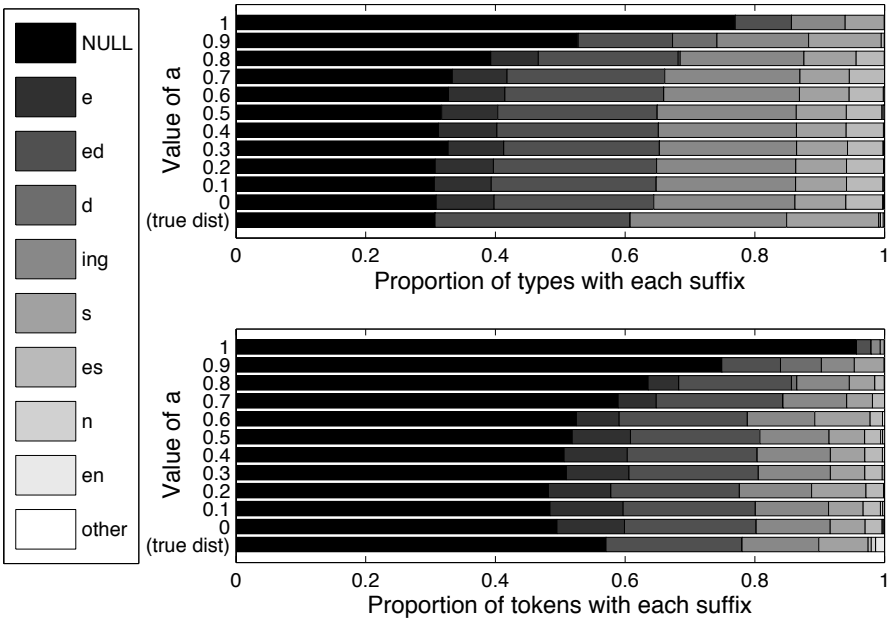


Figure 11: Results of the two-stage morphology learner for various values of a on the verb data set. The proportion of word types (top) and tokens (bottom) found with each suffix is shown, along with the distribution of suffixes in the gold standard.

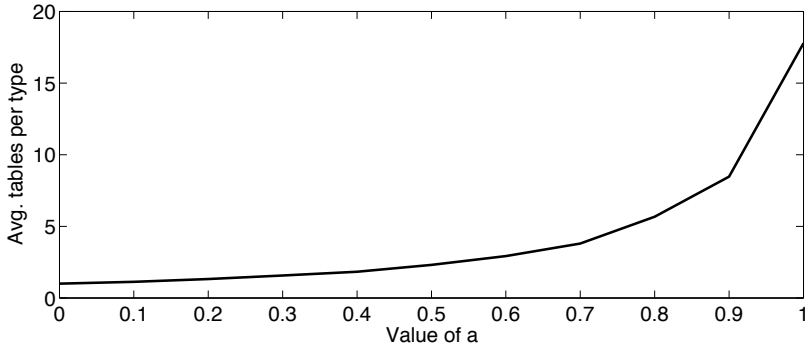


Figure 12: Average number of tables used per word type for each value of a.

since the total number of distinct stems plus suffixes is smaller. Only a few extra suffixes must be included to avoid near duplication of a large number of stems.

The primary remaining source of error that can be seen in the confusion matrices comes from wordforms analyzed as containing no suffix, where actually some non-empty suffix was present. In most cases, these were words where only a single inflected form was present in the data, so there was no reason for the system to postulate a complex analysis.

8.2 Experiment 2: Child-directed Speech

Experiment 1 used a corpus of verbs in orthographic form as input data, partly because learning English verbs is a standard task for computational models of morphology, and partly because this choice of corpus makes it possible to evaluate against a gold standard. However, using a single part of speech is a gross oversimplification of the learning problem. We therefore performed a second experiment using a corpus of phonemically transcribed child-directed speech, as described below.

8.2.1 DATA

The original source of the data used in this experiment was the Brown corpus (Brown, 1973) from the CHILDES database (MacWhinney and Snow, 1985), which contains transcribed parent-child interactions from long-term observational studies on three English-learning children. We extracted all the words spoken by caretakers, and converted the representations of these from standard written form to phonemic form using a phonemic dictionary.[13] Variations in pronunciation indicated in the original transcriptions (e.g., *going* vs. *goin'*) were preserved as much as possible in the phonemic forms (`goIN`, `goIn`),[14] and many non-words (e.g., *hm*) were also retained, making this corpus somewhat noisy. There are a total of 369,443 word tokens in the corpus belonging to 6,807 types. The total number of unique prefix strings T is 14,639, and the total number of unique suffix strings F is 16,313. Since there is no gold standard for this corpus, our evaluation is qualitative, based on examining the output of the algorithm.

8.2.2 RESULTS

Qualitatively, the results of varying the PYCRP parameter a are similar for this data set and the corpus of English verbs. Table 2 shows that as a increases, the number of different suffixes found decreases, and the proportion of word types analyzed with empty suffixes increases. As an indicator of the effect on other suffixes, the proportion of words found to contain the most common non-empty suffix `z` is also shown. As in the verb corpus, the highest values of a lead to analyses with almost no interesting morphological structure, while for lower values, many words are found to contain non-empty suffixes.

An interesting difference between the results from the two corpora is noticeable for the lowest values of a. In the verb corpus, results were very similar for values of $a \leq .7$. Here, there is a more graded effect, and for $a \leq .2$ the system actually produces too many different suffix types. Examining the output of the system with $a = 0$ (summarized in Table 3) illustrates the problem. Five of the classes are reasonable: three contain primarily nouns, with possible suffixes NULL and

13. We thank James Morgan and the Metcalf Infant Research Lab at Brown University for providing the phonemic dictionary for this corpus.
14. We use `typewriter` font to indicate phonemic symbols. The phonemic alphabet used in this data set is provided in Appendix B.

(a) $a = 0.1$

Tables	Stems		Suffixes	
1473	advis	9	ed	499
	rang	8	ing	371
	eliminat	8	e	255
	pass	8	NULL	177
	settl	8	es	171
	compar	8		
	...			
1936	remov	13	ed	615
	assum	10	e	539
	enabl	9	ing	480
	produc	9	es	296
	continu	9	en	6
	prov	8		
	...			
1333	represent	9	NULL	612
	back	9	ed	305
	contend	8	ing	250
	list	8	s	166
	maintain	8		
	walk	8		
	...			
1255	see	13	NULL	650
	adjust	12	ed	228
	yield	10	ing	217
	want	9	s	148
	limit	8	n	12
	fill	8		
	...			
1319	total	13	NULL	674
	work	10	ed	255
	respond	9	ing	244
	add	9	s	146
	equal	8		
	shift	8		
	...			
1531	open	11	NULL	715
	ask	9	ed	337
	fund	8	ing	285
	turn	8	s	194
	reflect	8		
	demand	8		
	...			

(b) $a = 0.6$

Tables	Stems		Suffixes	
2684	reach	44	NULL	1240
	discuss	42	ed	859
	push	42	ing	466
	match	38	es	70
	learn	37	s	49
	talk	35		
	...			
4127	say	138	NULL	3697
	think	96	s	267
	see	91	ing	132
	know	70	ting	15
	keep	63	n	13
	find	60	th	3
	...			
3672	includ	113	ed	1485
	increas	111	e	1003
	requir	73	ing	849
	involv	68	es	335
	reduc	66		
	indicat	64		
	...			
4351	us	182	ed	1712
	continu	110	e	1293
	mov	81	ing	933
	provid	68	es	413
	fac	67		
	receiv	63		
	...			
4268	offer	97	NULL	1851
	add	78	ed	1084
	report	73	ing	872
	boost	66	s	461
	start	56		
	follow	56		
	...			
3902	reflect	76	NULL	1601
	help	68	ed	1204
	develop	64	ing	721
	show	61	s	375
	consider	55	-sorting	1
	allow	52		
	...			

Table 1: Sample solutions for the WSJ verb corpus with (a) $a = .1$ and (b) $a = .6$, with boundaries initialized at random. The number of tables assigned to each class is shown in column 1, followed by the most frequent stems and suffixes in that class, and their table counts.

a	Suffix types	% NULL	% -z
0	78	58.0	10.2
.1	76	64.1	9.6
.2	40	73.8	8.8
.3	17	80.8	7.7
.4	17	84.9	6.6
.5	13	88.0	5.4
.6	12	90.5	4.8
.7	13	94.3	2.9
.8	10	99.6	2.2
.9	12	98.7	0.8
1	11	99.8	0.2

Table 2: Effects of varying the parameter a on the results from the Bernstein-Ratner-Morgan corpus. Columns show the total number of suffix types found, percentage of word types with empty suffixes, and percentage of word types with the suffix -z.

-z, and two contain large numbers of verbs with a variety of inflectional and derivational suffixes (including allomorphic and phonetic variants). The final class, however, contains a set of words that are phonologically rather than morphosyntactically similar. In particular, the words dominating this class are very short (mostly monosyllabic) and consist of common sequences of phonemes. Among these words, the hypothesized "stems" consist of the initial consonant(s) and vowel of a syllable, and the "suffixes" are the final consonant(s), or occasionally a second syllable. Rather than morphological structure, the system has discovered phonological structure.

Interestingly, as the value of a is increased, the system's tendency to split words into half-syllables decreases faster than its tendency to split words at morpheme boundaries. Moving from $a = 0$ to $a = .3$ reduces the number of hypothesized suffix types from 78 to 17 (those found in the noun and verb classes in Table 3, plus -n, -6n, -l, -&d, and -lnz) and reduces the percentage of words with non-empty suffixes by 54%, but only reduces the percentage of words with the -z suffix by 25%. All six classes in this condition correspond roughly to either nouns or verbs. We hypothesize that adding just a small amount of frequency information (with $a = .3$, the sampled solution contained 12,463 tables, versus 6,807 with $a = 0$) is enough for the system to realize that half-syllables do not have the same kind of near-independence between "stem" and "suffix" that true stem-suffix words do. Unfortunately, since there is no gold standard for this corpus, we don't know the true percentage of morphologically complex types, or types with the -z suffix. In future work, it would be useful to perform a more detailed analysis of a representative sample of the corpus to get a better sense of the accuracy of the system and the kinds of errors it makes.

8.3 Discussion

Our two experiments demonstrate how the PYCRP adaptor can be used within our two-stage framework to interpolate between type and token frequencies in a model for learning non-trivial linguistic structure. Our results suggest that, for induction of regular morphology, statistics derived from the

Tables	Stems		Suffixes		Tables	Stems		Suffixes	
915	gArti	2	NULL	777	1212	jAmp	6	NULL	736
	barbar6	2	z	138		fOl	6	z	153
	kIC1n	2				spIl	6	1N	83
	kro	2				slip	6	s	64
	k&m6l	2				kUk	6	d	49
	TIN	2				yEl	5	1n	38
	Cer	2				f9t	5	i	32
	skQt	2				r9d	5	6r	25
	pIkC6r	2				sp&Nk	5	t	16
	nobadi	2				pIk	5	6l	16
	bAt6rfl9	2				tep	5		
	b&nded	2				tArn	5		
	...					...			
867	EvribAdi	2	NULL	761	1437	ple	9	NULL	687
	notbUk	2	z	106		muv	8	1N	170
	lEp6rd	2				kQnt	7	1n	98
	fAn6l	2				slIp	7	z	97
	pl&n	2				klin	7	6r	79
	wUd	2				tiC	6	d	65
	brAD6r	2				wOk	6	s	59
	r&mbl6r	2				mark	6	t	57
	duti	2				rol	6	i	53
	kartun	2				dr9v	6	6z	45
	f9rm6n	2				rAb	6	6rz	27
	dorbEl	2				k&ri	6		
	...					...			
862	kUS6n	2	NULL	735	1514	NULL	22	NULL	255
	p6tuny6	2	z	127		p&	19	t	89
	meri6n	2				&	19	n	84
	DEm	2				bi	18	z	73
	pEns1l	2				hI	16	d	72
	pep6r	2				e	16	l	65
	bAlb	2				pE	15	r	52
	fom	2				ste	15	k	44
	stAf1n	2				t9	15	p	41
	b9s1k6l	2				dI	15	s	40
	hEv6n	2				w9	14	ni	38
	tEl6fon	2				bE	14	nz	36
	...					...		...	

Table 3: Sample solution for the Brown-Morgan corpus with $a = 0$. For each class, the number of tables assigned to that class is shown in column 1, followed by the most frequent stems and suffixes in that class, with their table counts. Note that since $a = 0$, table counts in this case are equal to type counts.

lexicon are more useful than statistics derived from corpus frequencies. This result agrees with the previous computational work of Albright and Hayes (2003), and supports the conclusions of Bybee (2001). It also justifies the use of word lists in many previous morphological learning systems (Plaut and Gonnerman, 2000; Regier et al., 2001; Snover and Brent, 2003). Interestingly, our experiments also suggest that partially damping corpus frequencies may be as effective, or perhaps even more effective, than fully damping frequencies (i.e., using only lexical statistics).

Of course, the experiments described here are limited in scope. The evidence against token-based learning of morphology would be stronger if additional experiments were performed with a larger variety of data from multiple languages, and if more detailed analysis were undertaken on the output from the Brown-Morgan corpus of child-directed speech. It would also be desirable to extend our model to account for more complex morphology, since the limitation to a single stem and suffix is inadequate to account for the morphology of most languages (including English, if derivational as well as inflectional morphology is considered). However, we emphasize that our focus here was not to develop a state-of-the-art morphological induction system, but rather to explore the consequences of using the PYCRP adaptor and its different parameter settings. We found that, with appropriate parameter settings, our model was sufficient to identify common suffixes in both corpora, and distinguish roughly between noun stems and verb stems in the Brown-Morgan corpus.

We have proposed that there are two main reasons that using the PYCRP adaptor to damp corpus frequencies yields better morphological segmentations than learning directly from corpus frequencies. First, the generator model assumes that stems and suffixes are independent given the morphological class, but this assumption is only approximately correct. Damping corpus frequencies brings the assumptions of the model and the data more in line, whereas using full corpus frequencies provides more evidence that stems and suffixes are not truly independent and therefore should not be split. Second, the most frequent words in any language tend to be irregular, and due to the power-law distribution of word frequencies, these words strongly dominate the corpus statistics. The effect of these suffix-less words is so strong that, despite a prior preference for solutions with fewer stems and suffixes, the system learns that most words should have no suffix. This causes many regular forms to go unsegmented.

Finally, we note that there are other important connections between our two-stage model and psycholinguistic theories of morphological processing. One question of concern to many psycholinguists is the extent to which morphologically complex words are stored and processed as single lexical units, as opposed to being decomposed into individual morphemes (Alegre and Gordon, 1999; Hay, 2001; Hay and Baayen, 2005). Our model provides an answer to this question, predicting specific testable relationships between word frequency, statistical independence of stem and suffix, and the probability of decomposition. While a thorough examination of these predictions and a comparison to behavioral data is beyond the scope of this paper, we note that an extension of our model (described further in the following section) has produced promising preliminary results in this area (O'Donnell, in preparation).

9. Further Applications and Extensions

The morphological segmentation model considered in the preceding sections illustrates how different assumptions about word frequency can result in different conclusions about the latent structure expressed in linguistic data. However, the potential of the two-stage approach to modeling language lies in its generality, with any existing probabilistic model of language potentially acting as a gen-

erator that can be combined with different adaptors. In this section, we consider how the two-stage framework can be applied to some other popular probabilistic models, how it can be extended to work with other kinds of linguistic structure, and how the challenges of scaling to larger corpora that arise with these applications and extensions can be addressed.

9.1 Applying the Two-stage Framework to Other Models

While the tension between types and tokens has been most explicit in computational linguistics, similar issues arise in other areas of research involving the analysis of text. For example, information retrieval systems typically represent documents in one of two ways: as a binary vector indicating which words appear in the document, or as a vector of word frequency counts (Baeza-Yates and Ribeiro-Neto, 1999). These two kinds of representations have different strengths and weaknesses, with the basic issue being that multiple occurrences of a word in a document do carry some information about the relevance of that document to a query, but not in a way that increases linearly with the number of instances. As a consequence, information retrieval systems typically make use of some kind of scheme for damping word frequencies.

Our two-stage framework provides a way to define an adaptive damping scheme for information retrieval models that have a probabilistic interpretation, such as the naïve Bayes classifier. In the standard naïve Bayes classifier, each class is assumed to be associated with a multinomial distribution over words, and the words that appear in each document are assumed to be drawn independently from that distribution. This model can be used as the generator for a two-stage model, with an adaptor such as the PYCRP being used to guarantee that the resulting word frequency distribution has statistical properties closer to natural language. This is essentially the model used in our analysis in Section 6.1, where we show that multinomial generators estimated using this model are similar to those that damp word frequencies. Evidence that this approach should lead to good empirical results comes from the work of Elkan (2006), who used a Dirichlet compound multinomial model (which is a special case of our framework, as noted above) to improve performance on several information retrieval tasks.

More complex machine learning models that have been applied to text also face a choice between representing documents in terms of types or tokens. For example, latent Dirichlet allocation (Blei et al., 2003) treats each document as a "bag of words", represented by a vector of word frequencies, as does its nonparametric analogue based on the hierarchical Dirichlet process (Teh et al., 2005). In contrast, a recent hierarchical nonparametric Bayesian model based on the beta process treats documents as binary vectors of word types (Thibaux and Jordan, 2007). It is straightforward to define a two-stage model in which LDA is used as a generator, which would provide a way to automatically interpolate between these two extremes. Probabilistic inference in this model is comparable in computational complexity to the Gibbs sampling scheme commonly used with LDA (Griffiths and Steyvers, 2004): to return to the restaurant metaphor used above, while a new random variable is introduced for each word indicating the table from which it is drawn, the number of random variables that need to be sampled in the LDA model scales with the total number of tables rather than the total number of words.

9.2 Extending the Framework to Other Linguistic Structures

We argued briefly above that the tendency of irregular forms to dominate corpus statistics is not specific to the problem addressed here, but can be expected to occur in many linguistic learning

tasks. Similarly, nearly all probabilistic models used for language learning (most notably, hidden Markov models and PCFGs) encode strong independence assumptions similar to those in our morphology generator model. Thus, we extrapolate from the results of our experiments to suggest that using the PYCRP or other power-law adaptors in combination with more standard models as generators may be able to improve unsupervised learning in many areas of language. Indeed, in other recent work we have developed several two-stage models for learning linguistic structure, achieving results comparable to, and in some cases better than, the best existing systems. For example, adaptor grammars (Johnson et al., 2007) combine a PYCRP adaptor with a PCFG generator to create a model for learning linguistic tree structures without the strong independence assumptions made by a standard PCFG. The adaptor effectively caches entire subtrees so that frequent structures can be reused, and will be assigned probabilities that are higher than the product of the PCFG rules that would be needed to create them anew. Although PCFGs are typically associated with syntactic constituency structure, they can also be used to express other types of linguistic relationships, and adaptor grammars have been used to learn word segmentation, syllable structure, morphology, dependency parses, and named-entity clusters (Johnson et al., 2007; Johnson, 2008a,b; Johnson and Goldwater, 2009; Cohen et al., 2010; Elsner et al., 2009). In fact, it is even possible to express the standard LDA model using the adaptor grammar framework (Johnson, 2010).

In addition to adaptor grammars, the two-stage framework provides the basis of another recent model for learning trees, independently introduced by Cohn et al. (2009), Post and Gildea (2009), and O'Donnell et al. (2009).[15] This model can be viewed as a generalization of the adaptor grammar. In an adaptor grammar, all trees produced by the generator are complete, with terminal symbols at all leaf nodes. In contrast, the model presented by the authors above allows the generator to produce incomplete tree fragments or *elementary trees*, with either terminal or non-terminal symbols as leaves. It therefore instantiates a nonparametric Bayesian model of tree-substitution grammar (Joshi, 2003). So far, the model has been used in NLP research to induce tree-substitution grammars from parsed sentences (Cohn et al., 2009; Post and Gildea, 2009) and to induce dependency structure from strings (Cohn et al., 2010). It has also shown promise as a model of human language processing, with applications to children's acquisition of syntax (O'Donnell et al., 2009) and adult morphological processing (O'Donnell, in preparation).

9.3 Strategies for Scaling to Larger Corpora

Using the two-stage framework with adaptors based on the CRP introduces a potentially challenging problem of probabilistic inference. In these models, each word is associated with a random variable indicating its source (or the table from which it was generated, under the restaurant analogy). The number of random variables in the model thus grows linearly with the number of words. While this is not unusual for probabilistic models of language that involve latent variables (for example, LDA has the same property), it means that alternatives to the simple Gibbs sampling algorithm we used in our morphological segmentation example will need to be developed in order to apply these models to large corpora of the kind used in modern machine learning and computational linguistics. There are three strategies for dealing with this scaling problem: using the two-stage framework to justify heuristic approximations but not explicitly performing inference, exploring parallelization schemes,

15. There are actually very slight differences in formulation between the model introduced by O'Donnell et al. (2009) and the other two, but they are conceptually similar.

and applying approximate inference techniques such as variational inference. We consider these options in turn.

Part of the motivation for our detailed treatment of the relationship between our two-stage framework and existing smoothing methods was to point out that these highly successful methods can be viewed as heuristic approximations to a model that makes reasonable assumptions about the structure of natural language. Kneser-Ney smoothing approximates a simple application of our two-stage framework, suggesting that it might be possible to derive similar heuristic approximations for more complex models. Some very simple approximations are the *minimal* and *maximal* schemes discussed by Cowans (2006) and Wallach (2008) in relation to other Bayesian language models. These make the respective assumptions that only one token of each type is drawn from the base distribution, or that all tokens of each type are drawn from the base distribution. However, the prospect of developing better approximations to more complex models seems promising, especially given recent results on the approximate and asymptotic properties of discrete models based on the Pitman-Yor process (e.g., Teh, 2006a; Buntine and Hutter, 2010). One strategy for applying two-stage models to large corpora may thus be to avoid performing inference explicitly, and instead derive approximations based on these results.

A second strategy is parallelization. As noted above, the property that makes probabilistic inference potentially problematic in two-stage models—the number of latent variables increasing linearly with the number of words in a corpus—is shared with other probabilistic models such as LDA. Parallelization has proven to be an effective strategy for applying models such as LDA to very large corpora (e.g., Newman et al., 2009). Recent work has already examined how parallelization can be used to increase the scale of the corpora on which language models based on the Pitman-Yor process can be applied, making it possible to use these models on a corpus containing 200 million words (Huang and Renals, 2010).

Finally, variational inference presents a third avenue for developing two-stage models that can be applied to large corpora, trading the stochastic approximation produced by Gibbs sampling for a deterministic approximation to the posterior distribution over the latent variables in the model. Recent work has focused on applying this strategy with adaptor grammars, which can be used to express many two-stage models as noted above. This work suggests that variational inference may yield a different pattern of scaling in the computational cost of using these models, making it more plausible that they can be applied to large corpora (Cohen et al., 2010).

10. Conclusion

In this paper we have introduced a framework for developing statistical models of language that breaks those models into two stages: one stage in which a basic set of lexical items is generated, and one stage in which the frequencies of those items are adapted to match the statistical structure of natural language. This two-stage framework solves two basic problems for statistical models of language: defining models that can generically exhibit power-law frequency distributions, and understanding how the observed frequencies of words should be damped when estimating parameters. Surprisingly, our work shows that these two problems are directly related, with damping of frequencies falling naturally out of our framework when we take into account the possibility that a secondary "rich-get-richer" process might be responsible for the power-law distribution in word frequencies.

More generally, the framework we have introduced in this paper illustrates how ideas from nonparametric Bayesian statistics can be valuable in the context of computational linguistics. The key innovation in nonparametric Bayesian statistics is the idea of defining models with potentially infinite complexity, allowing the structures recovered by those models to grow as more data are observed. In many ways, computational linguistics is the ideal application of this idea, since larger corpora always bring with them new vocabulary items, new constituents, and new constructions to be incorporated into a model. Recent work provides many other examples suggesting that nonparametric Bayesian statistics and natural language may be well suited to one another (Beal et al., 2002; Liang et al., 2007; Goldwater et al., 2006a,b; Teh et al., 2005; Teh, 2006a,b; Cohn et al., 2010) and we anticipate that this relationship will continue to be fruitful.

Acknowledgments

This paper expands on work that was presented at the Neural Information Processing Systems conference (Goldwater et al., 2006a). TLG was supported by National Science Foundation grant number BCS-0631518 and the DARPA CALO project. MJ was supported by NSF awards 0544127 and 0631667.

Appendix A. Details of Table Count Approximation Experiments

The generator used in this model was assumed to be a multinomial distribution over 30,114 word types, with φ being the probabilities assigned to these types. Estimation of φ was performed using Markov chain Monte Carlo. Taking a symmetric Dirichlet(β) prior over φ, the posterior distribution over φ given $\mathbf{w}$ and a particular value of $\mathbf{z}$ and $\boldsymbol{\ell}$ is Dirichlet with hyperparameters $m_w + \beta$, where m_w is the number of lexical items corresponding to the word type w (ie. the number of tables on which w appears). The mean probability of w under this distribution is proportional to $m_w + \beta$. Consequently, we can compute the posterior mean of φ by drawing samples of $\mathbf{z}$ and $\boldsymbol{\ell}$ from $P(\mathbf{z}, \boldsymbol{\ell} | \mathbf{w})$, computing the mean probability of each word type w given each of these samples, and then averaging the results across samples.

To draw samples from $P(\mathbf{z}, \boldsymbol{\ell} | \mathbf{w})$ we used a Gibbs sampling procedure very similar to that used with the morphology model in the main text. Since the lexical items had no internal analyses, it was only necessary to sample the table assignment z_i for each word token in the corpus in each sweep of sampling. This was done by drawing a value from the distribution

$$P(z_i = z | \mathbf{z}_{-i}, \mathbf{w}, \boldsymbol{\ell}(\mathbf{z}_{-i})) \propto \begin{cases} I(\ell_z = w_i)(n_z^{(\mathbf{z}_{-i})} - a) & 1 \le z \le K(\mathbf{z}_{-i}) \\ P(\ell_z = w_i)(K(\mathbf{z}_{-i})a + b) & z = K(\mathbf{z}_{-i}) + 1 \end{cases}$$

where $\mathbf{z}_{-i}$ is all $\mathbf{z}$ but z_i, $n_z^{(\mathbf{z}_{-i})}$ is the number of times z occurs in $\mathbf{z}_{-i}$, $K(\mathbf{z}_{-i})$ is the number of unique values in $\mathbf{z}_{-i}$, and a and b are the parameters of the PYCRP adaptor (the CRP adaptor was simulated by taking $a = 0$, in which case b plays the same role as α). $P(\ell_z = w_i)$ was obtained by integrating over the posterior distribution on φ given $\mathbf{z}_{-i}$ and $\boldsymbol{\ell}(\mathbf{z}_{-i})$, namely $(m_{w_i} + \beta) / \sum_w (m_w + \beta)$.

A total of 1000 sweeps of sampling were conducted for each adaptor, and the posterior mean of φ was computed for each sweep, which involved finding the mean number of lexical entries for each word type w. These values where then averaged over the last 500 iterations, discarding the initial sweeps to allow convergence of the Markov chain. The results shown in Figure 6 are thus

the posterior mean of the number of lexical entries assigned to each word type given the corpus **w**, and provide an indication of how word frequency translates into the frequencies from which the generator is estimated in this model.

Appendix B. Phonemic Symbols

The following ASCII characters are used in the phonemic transcriptions in the Brown-Morgan corpus, which was used as input to the morphological learner in Section 8.2.

Consonants

ASCII	Example	ASCII	Example
D	THe	k	Cut
N	siNG	l	Lamp
S	SHip	m	Man
T	THin	n	Net
Z	aZure	p	Pipe
C	CHip	r	Run
b	Boy	s	Sit
d	Dog	t	Toy
f	Fox	v	View
g	Go	w	We
h	Hat	y	You
j	Jump	z	Zip

Vowels

ASCII	Example	ASCII	Example
&	thAt	e	bAY
1	hopelEss	i	bEE
6	About	o	bOAt
7	bOY	u	bOOt
9	flY		
A	bUt		
E	bEt		
I	bIt		
O	lAW		
Q	bOUt		
U	pUt		
a	hOt		

References

Adam Albright and Bruce Hayes. Rules vs. analogy in English past tenses: a computational/experimental study. *Cognition*, 90:118–161, 2003.

David Aldous. Exchangeability and related topics. In *École d'été de probabilités de Saint-Flour, XIII—1983*, pages 1–198. Springer, Berlin, 1985.

Maria Alegre and Peter Gordon. Frequency effects and the representational status of regular inflections. *Journal of Memory and Language*, 40(1):41–61, 1999.

Charles Antoniak. Mixtures of Dirichlet processes with applications to Bayesian nonparametric problems. *The Annals of Statistics*, 2:1152–1174, 1974.

Richard Arratia, A. D. Barbour, and Simon Tavaré. Poisson process approximations for the Ewens sampling formula. *Annals of Applied Probability*, 2:519–535, 1992.

Ricardo Baeza-Yates and Berthier Ribeiro-Neto. *Modern Information Retrieval*. ACM press, New York, 1999.

Matthew Beal, Zoubin Ghahramani, and Carl Rasmussen. The infinite hidden Markov model. In *Advances in Neural Information Processing Systems 14*, Cambridge, MA, 2002. MIT Press.

David Blackwell and James B. MacQueen. Ferguson distributions via Pólya urn schemes. *Annals of Statistics*, 1:353–355, 1973.

David Blei, Andrew Ng, and Michael Jordan. Latent Dirichlet allocation. In T. Dietterich, S. Becker, and Z. Ghahramani, editors, *Advances in Neural Information Processing Systems 14*, Cambridge, MA, 2002. MIT Press.

David Blei, Andrew Ng, and Michael Jordan. Latent Dirichlet allocation. *Journal of Machine Learning Research*, 3:993–1022, 2003.

David Blei, Thomas L. Griffiths, Michael Jordan, and Joshua B. Tenenbaum. Hierarchical topic models and the nested Chinese restaurant process. In S. Thrun, L. Saul, and B. Schölkopf, editors, *Advances in Neural Information Processing 16*, Cambridge, MA, 2004. MIT Press.

Michael Brent. An efficient, probabilistically sound algorithm for segmentation and word discovery. *Machine Learning*, 34:71–105, 1999.

Roger Brown. *A First Language: The Early Stages*. Harvard University Press, Cambridge, MA, 1973.

Wray L. Buntine and Marcus Hutter. A Bayesian review of the Poisson-Dirichlet process. http://arxiv.org/abs/1007.0296, Jul 2010.

Joan Bybee. *Morphology: a Study of Relation between Meaning and Form*. Benjamins, Amsterdam, 1985.

Joan Bybee. *Phonology and Language Use*. Cambridge University press, Cambridge, UK, 2001.

Stanley F. Chen and Joshua Goodman. An empirical study of smoothing techniques for language modeling. Technical Report TR-10-98, Center for Research in Computing Technology, Harvard University, 1998.

Shay B. Cohen, David M. Blei, and Noah A. Smith. Variational inference for adaptor grammars. In *Proceedings of Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 564–572, Los Angeles, California, 2010.

Trevor Cohn, Sharon Goldwater, and Phil Blunsom. Inducing compact but accurate tree-substitution grammars. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 548–556, Boulder, Colorado, 2009.

Trevor Cohn, Sharon Goldwater, and Phil Blunsom. Inducing tree substitution grammars. *Journal of Machine Learning Research*, 11:3053–3096, Nov. 2010.

Michael Collins, Lance Ramshaw, Jan Hajič, and Christoph Tillmann. A statistical parser for Czech. In *Proceedings of the 37th Annual Meeting of the Association for Computational Linguistics*, pages 505–512, College Park, Maryland, USA, 1999.

Brooke Cowan and Michael Collins. Morphology and reranking for the statistical parsing of Spanish. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 795–802, Vancouver, 2005.

Philip Cowans. *Probabilistic Document Modelling*. PhD thesis, Cambridge University, 2006.

Mathias Creutz and Krista Lagus. Induction of a simple morphology for highly-inflecting languages. In *Proceedings of the Seventh Meeting of the ACL Special Interest Group in Computational Phonology (SIGPHON)*, pages 43–51, Barcelona, Spain, 2004.

Mathias Creutz and Krista Lagus. Inducing the morphological lexicon of a natural language from unannotated text. In *Proceedings of the International and Interdisciplinary Conference on Adaptive Knowledge Representation and Reasoning*, pages 51–59, Espoo, Finland, 2005.

Charles Elkan. Clustering documents with an exponential-family approximation of the Dirichlet compound multinomial distribution. In *Proceedings of the 23rd International Conference on Machine Learning*, pages 289–296, Pittsburgh, Pennsylvania, 2006.

Micha Elsner, Eugene Charniak, and Mark Johnson. Structured generative models for unsupervised named-entity clustering. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 164–172, Boulder, Colorado, 2009.

Michael D. Escobar and Mike West. Bayesian density estimation and inference using mixtures. *Journal of the American Statistical Association*, 90(430):577–588, 1995.

Thomas S. Ferguson. A Bayesian analysis of some nonparametric problems. *Annals of Statistics*, 1:209–230, 1973.

Jenny Finkel, Trond Grenager, and Christopher D. Manning. The infinite tree. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 272–279, Prague, Czech Republic, 2007.

Walter R. Gilks, Sylvia Richardson, and David J. Spiegelhalter, editors. *Markov Chain Monte Carlo in Practice*. Chapman and Hall, Suffolk, 1996.

John Goldsmith. Unsupervised learning of the morphology of a natural language. *Computational Linguistics*, 27:153–198, 2001.

John Goldsmith. An algorithm for the unsupervised learning of morphology. *Journal of Natural Language Engineering*, 12:353–371, 2006.

Sharon Goldwater, Thomas L. Griffiths, and Mark Johnson. Interpolating between types and tokens by estimating power-law generators. In *Advances in Neural Information Processing Systems 18*, Cambridge, MA, 2006a. MIT Press.

Sharon Goldwater, Thomas L. Griffiths, and Mark Johnson. Contextual dependencies in unsupervised word segmentation. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, Syndney, Australia, 2006b.

Sharon Goldwater, Thomas L. Griffiths, and Mark Johnson. A Bayesian framework for word segmentation: Exploring the effects of context. *Cognition*, 112(1):21–54, 2009.

Joseph Greenberg. Language universals. In T. A. Sebok, editor, *Current Trends in Linguistics III*. Mouton, The Hague, 1966.

Thomas L. Griffiths. Power-law distributions and nonparametric Bayes. Unpublished manuscript, 2006.

Thomas L. Griffiths and Mark Steyvers. Finding scientific topics. *Proceedings of the National Academy of Science*, 101:5228–5235, 2004.

Dilek Z. Hakkani-Tür, Kemal Oflazer, and Gökhan Tür. Statistical morphological disambiguation for agglutinative languages. *Computers and the Humanities*, 36(4):381–410, 2002.

Zelig Harris. From phoneme to morpheme. *Language*, 31:190–222, 1955.

Jennifer Hay. Lexical frequency in morphology: Is everything relative? *Linguistics*, 39(6):1041–1070, 2001.

Jennifer Hay and R. Harald Baayen. Shifting paradigms: gradient structure in morphology. *Trends in Cognitive Sciences*, 9(7):342–348, 2005.

Songfang Huang and Steve Renals. Hierarchical Bayesian language models for conversational speech recognition. *IEEE Transactions on Audio, Speech and Language Processing*, 18(8):1941–1954, 2010.

Hemant Ishwaran and Lancelot James. Generalized weighted Chinese restaurant processes for species sampling mixture models. *Statistica Sinica*, 13:1211–1235, 2003.

Ray Jackendoff. *Foundations of Language: Brain, Meaning, Grammar, Evolution*. Oxford University Press, Oxford/New York, 2002.

Mark Johnson. Using adaptor grammars to identify synergies in the unsupervised acquisition of linguistic structure. In *Proceedings of ACL-08: HLT*, Columbus, Ohio, 2008a.

Mark Johnson. Unsupervised word segmentation for Sesotho using adaptor grammars. In *Proceedings of the Tenth Meeting of ACL Special Interest Group on Computational Morphology and Phonology (SIGMORPHON)*, Columbus, Ohio, 2008b.

Mark Johnson. PCFGs, topic models, adaptor grammars and learning topical collocations and the structure of proper names. In *Proceedings of the 48th Annual Meeting of the Association of Computational Linguistics*, pages 1148–1157, Uppsala, Sweden, 2010.

Mark Johnson and Sharon Goldwater. Improving nonparametric Bayesian inference: Experiments on unsupervised word segmentation with adaptor grammars. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, Boulder, Colorado, 2009.

Mark Johnson, Thomas L. Griffiths, and Sharon Goldwater. Adaptor grammars: a framework for specifying compositional nonparametric Bayesian models. In B. Schölkopf, J. Platt, and T. Hoffman, editors, *Advances in Neural Information Processing Systems 19*, Cambridge, MA, 2007. MIT Press.

Aravind Joshi. Tree adjoining grammars. In Ruslan Mikkov, editor, *The Oxford Handbook of Computational Linguistics*, pages 483–501. Oxford University Press, Oxford, England, 2003.

Reinhard Kneser and Hermann Ney. Improved backing-off for *n*-gram language modeling. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 181–184, Detroit, Michigan, 1995.

Philipp Koehn and Hieu Hoang. Factored translation models. In *Proceedings of the Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 868–876, Prague, Czech Republic, 2007.

Leah S. Larkey, Lisa Ballesteros, and Margaret Connell. Improving stemming for Arabic information retrieval: Light stemming and co-occurrence analysis. In *Proceedings of the 25th International Conference on Research and Development in Information Retrieval (SIGIR)*, pages 275–282, Tampere, Finland, 2002.

Percy Liang, Slav Petrov, Michael Jordan, and Dan Klein. The infinite PCFG using hierarchical Dirichlet processes. In *Proceedings of the Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 688–697, Prague, Czech Republic, 2007.

Albert Lo. On a class of Bayesian nonparametric estimates. *Annals of Statistics*, 12:351–357, 1984.

David MacKay and Linda Bauman Peto. A hierarchical Dirichlet language model. *Natural Language Engineering*, 1(1), 1994.

Brian MacWhinney and Catharine Snow. The child language data exchange system. *Journal of Child Language*, 12:271–296, 1985.

Rasmus Madsen, David Kauchak, and Charles Elkan. Modeling word burstiness using the Dirichlet distribution. In *Proceedings of the 22nd International Conference on Machine Learning*, pages 545–552, Bonn, Germany, 2005.

Mitchell P. Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. Building a large annotated corpus of English: the Penn treebank. *Computational Linguistics*, 19(2):331–330, 1993.

Michael Mitzenmacher. A brief history of generative models for power law and lognormal distributions. *Internet Mathematics*, 1(2):226–251, 2004.

Christian Monson, Alon Lavie, Jaime Carbonell, and Lori Levin. Unsupervised induction of natural language morphology inflection classes. In *Proceedings of the Seventh Meeting of the ACL Special Interest Group in Computational Phonology (SIGPHON)*, pages 52–61, Barcelona, Spain, 2004.

Radford Neal. Markov chain sampling methods for Dirichlet process mixture models. *Journal of Computational and Graphical Statistics*, 9:249–265, 2000.

David Newman, Arthur Asuncion, Padhraic Smyth, and Max Welling. Distributed algorithms for topic models. *Journal of Machine Learning Research*, 10(Aug):1801–1828, 2009.

Hermann Ney, Ufe Essen, and Reinhard Kneser. On structuring probabilistic dependencies in stochastic language modeling. *Computer, Speech, and Language*, 8:1–38, 1994.

Timothy O'Donnell. *Computation and Reuse in Language*. PhD thesis, Harvard University, in preparation.

Timothy O'Donnell, Noah Goodman, and Joshua B. Tenenbaum. Fragment grammars: Exploring computation and reuse in language. Technical Report MIT-CSAIL-TR-2009-013, MIT, 2009.

Janet Pierrehumbert. Probabilistic phonology: Discrimination and robustness. In R. Bod, J. Hay, and S. Jannedy, editors, *Probabilistic Linguistics*. MIT Press, Cambridge, MA, 2003.

Jim Pitman. Exchangeable and partially exchangeable random partitions. *Probability Theory and Related Fields*, 102:145–158, 1995.

Jim Pitman. *Combinatorial Stochastic Processes*. Springer-Verlag, New York, 2006.

Jim Pitman and Marc Yor. The two-parameter Poisson-Dirichlet distribution derived from a stable subordinator. *Annals of Probability*, 25:855–900, 1997.

David Plaut and Laura Gonnerman. Are non-semantic morphological effects incompatible with a distributed connectionist approach to lexical processing? *Language and Cognitive Processes*, 15: 445–485, 2000.

Matt Post and Daniel Gildea. Bayesian learning of a tree substitution grammar. In *Proceedings of the ACL-IJCNLP 2009 Conference Short Papers*, pages 45–48, Suntec, Singapore, August 2009.

Carl Rasmussen. The infinite Gaussian mixture model. In *Advances in Neural Information Processing Systems 12*, Cambridge, MA, 2000. MIT Press.

Terry Regier, Bryce Corrigan, Rachael Cabasaan, Amanda Woodward, Michael Gasser, and Linda Smith. The emergence of words. In *Proceedings of the 23rd Annual Conference of the Cognitive Science Society*, pages 815–820, Mahwah, NJ, 2001. Erlbaum.

Gerard Salton and Christopher Buckley. Term-weighting approaches in automatic text retrieval. *Information Processing and Management: an International Journal*, 24(5):513–523, 1988.

Jayazam Sethuraman. A constructive definition of Dirichlet priors. *Statistica Sinica*, 4:639–650, 1994.

Herbert Simon. On a class of skew distribution functions. *Biometrika*, 42(3/4):425–440, 1955.

Matthew Snover and Michael Brent. A probabilistic model for learning concatenative morphology. In *Advances in Neural Information Processing Systems 15*, Cambridge, MA, 2003. MIT Press.

Benjamin Snyder and Regina Barzilay. Unsupervised multilingual learning for morphological segmentation. In *Proceedings of ACL-08: HLT*, pages 737–745, Columbus, Ohio, 2008.

Yee Whye Teh. A Bayesian interpretation of interpolated Kneser-Ney. Technical Report TRA2/06, National University of Singapore, School of Computing, 2006a.

Yee Whye Teh. A hierarchical Bayesian language model based on Pitman-Yor processes. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pages 985–992, Syndney, Australia, 2006b.

Yee Whye Teh, Michael Jordan, Matthew Beal, and David Blei. Hierarchical Dirichlet processes. In *Advances in Neural Information Processing Systems 17*. MIT Press, Cambridge, MA, 2005.

Romain Thibaux and Michael I. Jordan. Hierarchical beta processes and the Indian buffet process. In *Proceedings of the Eleventh International Conference on Artificial Intelligence and Statistics*, San Juan, Puerto Rico, 2007.

Hanna M. Wallach. *Structured Topic Models for Language*. PhD thesis, University of Cambridge, 2008.

Mike West. Hyperparameter estimation in Dirichlet process mixture models. Technical Report 92-A03, Institute of Statistics and Decision Sciences, Duke University, 1992.

Frank Wood and Yee Whye Teh. A hierarchical, hierarchical Pitman Yor process language model. In *Proceedings of the ICML/UAI Workshop on Nonparametric Bayes*, Helsinki, Finland, 2008.

Frank Wood and Yee Whye Teh. A hierarchical nonparametric Bayesian approach to statistical language model domain adaptation. In *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, Clearwater Beach, Florida, 2009.

David Yarowsky and Richard Wicentowski. Minimally supervised morphological analysis by multimodal alignment. In *Proceedings of the 38th Annual Meeting of the Association for Computational Linguistics*, pages 207–216, Hong Kong, 2000.

George Zipf. *Selective Studies and the Principle of Relative Frequency in Language*. Harvard University Press, Cambridge, MA, 1932.

Journal of Machine Learning Research 12 (2011) 2383-2387 Submitted 6/10; Revised 3/11; Published 7/11

Waffles: A Machine Learning Toolkit

Mike Gashler MIKE@AXON.CS.BYU.EDU
Department of Computer Science
Brigham Young University
Provo, UT 84602, USA

Editor: Soeren Sonnenburg

Abstract

We present a breadth-oriented collection of cross-platform command-line tools for researchers in machine learning called *Waffles*. The *Waffles* tools are designed to offer a broad spectrum of functionality in a manner that is friendly for scripted automation. All functionality is also available in a C++ class library. *Waffles* is available under the GNU Lesser General Public License.

Keywords: machine learning, toolkits, data mining, C++, open source

1. Introduction

Although several open source machine learning toolkits already exist (Sonnenburg et al., 2007), many of them implicitly impose requirements regarding how they can be used. For example, some toolkits require a certain platform, language, or virtual machine. Others are designed such that tools can only be connected together with a specific plug-in, filter, or signal/slot architecture. Unfortunately, these interface differences create difficulty for those who have become familiar with a different methodology, and for those who seek to use tools from multiple tookits together. Toolkits that use a graphical interface may be convenient for performing common experiments, but become cumbersome when the user wishes to use a tool in a manner that was not foreseen by the interface designer, or to automate common and repetitive tasks.

Waffles is a collection of tools that seek to provide a wide diversity of useful operations in machine learning and related fields without imposing unnecessary process or interface restrictions on the user. This is done by providing simple command-line interface (CLI) tools that perform basic tasks. The CLI is ideal for this purpose because it is well-established, it is available on most common operating systems, and it is accessible through most common programming languages. Since these tools perform operations at a fairly granular level, they can be used in ways not foreseen by the interface designer.

As an example, consider an experiment involving the following seven steps:

1. Use cross-validation to evaluate the accuracy of a bagging ensemble of one-hundred decision trees for classifying the *lymph* data set (available at `http://MLData.org`).
2. Separate this data set into a matrix of input-features and a matrix of output-labels.
3. Convert input-features to real-valued vectors by representing each nominal attribute as a categorical distribution over possible values.
4. Use principal component analysis to reduce the dimensionality of the feature-vectors.

5. Use cross-validation to evaluate the accuracy of the same model on the data with reduced features.
6. Train the model using all of the reduced-dimensional data.
7. Visualize the model-space represented by the ensemble.

These seven operations can be performed with Waffles tools using the following CLI commands:

```
1. waffles_learn crossvalidate lymph.arff bag 100 decisiontree end
2. waffles_transform dropcolumns lymph.arff 18 > features.arff
   waffles_transform dropcolumns lymph.arff 0-17 > labels.arff
3. waffles_transform nominaltocat features.arff > f_real.arff
4. waffles_dimred pca f_real.arff 2 > f_reduced.arff
5. waffles_transform mergehoriz f_reduced.arff labels.arff > all.arff
   waffles_learn crossvalidate all.arff bag 100 decisiontree end
6. waffles_learn train all.arff bag 100 decisiontree end > ensemble.model
7. waffles_plot model ensemble.model all.arff 0 1
```

The cross-validation performed in step 1 returns a predictive accuracy score of 0.781. Step 5 returns a predictive accuracy score of 0.705. The plot generated by step 7 is shown in Figure 1.

It is certainly conceivable that a graphical interface could be developed that would make it easy to perform an experiment like this one. Such an interface might even provide some mechanism to automatically perform the same experiment over an array of data sets, and using an array of different models. If, however, the user needs to vary a parameter specific to the experiment, such as the number of principal components, or a model-specific parameter, such as the number of trees in the ensemble, the benefits of a graphical interface are quickly overcome by additional complexity. By contrast, a simple script that calls CLI commands to perform machine learning operations can be directly modified to vary any of the parameters. Additionally, the scripting method can incorporate tools from other toolkits, or even custom-developed tools. Because nearly all programming languages can target CLI applications, there are few barriers to adding custom operations. Graphical tools are unlikely to offer such flexibility.

Figure 1: The model-space visualization generated by the command in step 7.

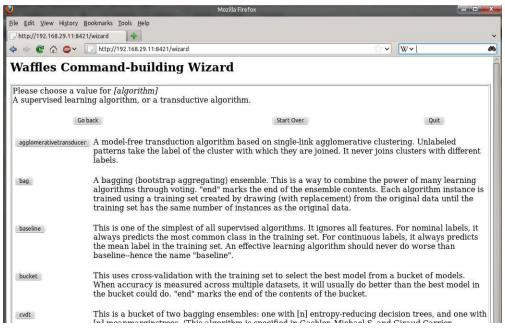


Figure 2: A partial screen shot of the Waffles Wizard tool displayed in a web browser.

2. Wizard

One significant reason many people prefer to use tools unified within a graphical interface over scriptable CLI tools is that it can be cumbersome to remember which options are available with CLI tools, and to remember how to construct a syntactically-correct command. We solve this problem by providing a "Wizard" tool that guides the user through a series of forms to construct a command that will perform the desired task. A screen shot of this tool (displayed in a web browser) is shown in Figure 2.

Rather than execute the selected operation directly, as most GUI tools do, the Waffles Wizard tool merely displays the CLI command that will perform the operation. The user may paste it directly into a command shell to perform the operation immediately, or the user may choose to incorporate it into a script. This gives the user the benefits of a GUI, without the undesirable tendency to lock the user into an interface that is inflexible for scripted automation.

3. Capabilities

In order to hilight the capabilities of Waffles, we compare its functionality with that found in Weka (Hall et al., 2009), which at the time of this writing is the most popular machine learning toolkit by a significant margin. Our intent is not to persuade the reader to choose Waffles instead of Weka, but rather to show that many useful capabilities can be gained by using Waffles in conjunction with Weka, and other toolkits that offer a CLI.

One notable strength of Waffles is in unsupervised algorithms, particularly dimensionality reduction techniques. Waffles tools implement principal component analysis (PCA), isomap (Tenenbaum et al., 2000), locally linear embedding (Roweis and Saul, 2000), manifold sculpting (Gashler et al., 2011a), breadth-first unfolding, neuro-PCA, cycle-cut (Gashler et al., 2011b), unsupervised backpropagation and temporal nonlinear dimensionality reduction (Gashler et al., 2011c). Of these, only PCA is found in Weka. Waffles contains clustering techniques including k-means, k-medoids, agglomerative clustering, and related transduction algorithms including agglomerative transduction, and max-flow/min-cut transduction (Blum and Chawla, 2001).

Waffles provides some of the most-common supervised learning techniques, such as decision trees, multi-layer neural networks, k-nearest neighbor, naive Bayes, and some less-common algorithms, such as Mean-margin trees (Gashler et al., 2008). Waffles' collection of supervised algo-

rithms is much smaller than that of Weka, which implements more than 50 classification algorithms. Waffles, however, provides an interface that offers several advantages in many situations. For example, Weka requires the user to set up filters that convert data to types that each algorithm can handle. Waffles automatically handles type conversion when an algorithm receives a type that it is not implicitly designed to handle, while still permitting advanced users to specify custom filters. The Waffles algorithms also implicitly handle multi-dimensional labels. As some algorithm-specific examples, the Waffles implementation of multi-layer perceptron provides the ability to use a diversity of activation functions, and also supplies methods for training recurrent networks. The *k*-nearest neighbor algorithm automatically supports acceleration structures and sparse training data, so it is suitable for use with problems that require high scalability, such as document classification.

As was demonstrated in the first example in this paper, Waffles features a particularly convenient mechanism for creating bagging ensembles. It also provides a diversity of collaborative filtering algorithms and optimization techniques that are not found in Weka. Waffles also provides tools to perform linear-algebraic operations, and various data-mining tools, including attribute selection and several methods for visualization.

4. Architecture

The Waffles tools are organized into several executable applications. These include:

1. **waffles_wizard**, a graphical command-building assistant,
2. **waffles_learn**, a collection of supervised learning techniques and algorithms,
3. **waffles_transform**, a collection of unsupervised data transformations,
4. **waffles_plot**, tools related to visualization,
5. **waffles_dimred**, tools for dimensionality reduction and attribute selection,
6. **waffles_cluster**, tools for clustering data,
7. **waffles_generate**, tools for sampling distributions, manifolds, etc.,
8. **waffles_recommend**, tools related to collaborative filtering, and
9. **waffles_sparse**, tools for learning with sparse matrices.

Each tool contained in each of these applications is implemented as a thin wrapper around functionality in a C++ class library, called *GClasses*. This library is included with Waffles so that any of the functionality available in the Waffles CLI tools can also be linked into C++ applications, or into applications developed in other languages that are capable of linking with C++ libraries. The entire Waffles project is licensed under the GNU Lesser General Public License (LGPL) version 2.1, and also later versions of the LGPL (`http://www.gnu.org/licenses/lgpl.html`). Also, some components are additionally granted more permissive licenses. Waffles uses a minimal set of dependency libraries, and is carefully designed to support cross-platform compatibility. It builds on Linux (with g++), Windows (with Visual C++ Express Edition), OSX (with g++), and most other common platforms. A new version of Waffles has been released approximately every six months since it was first released to the public in 2005. The latest version can be downloaded from `http://waffles.sourceforge.net`. Full documentation for the CLI tools, including many examples, and also documentation for developers seeking to link with the GClasses library can also be found at that site. In order to augment the developer documentation, several demo applications are also included with Waffles, showing how to build machine learning tools that link with functionality in the GClasses library.

References

A. Blum and S. Chawla. Learning from labeled and unlabeled data using graph mincuts. In *Proceedings of the Eighteenth International Conference on Machine Learning*, pages 19–26. Morgan Kaufmann Publishers Inc., 2001. ISBN 1558607781.

M. Gashler, C. Giraud-Carrier, and T. Martinez. Decision tree ensemble: Small heterogeneous is better than large homogeneous. In *Seventh International Conference on Machine Learning and Applications, 2008. ICMLA '08.*, pages 900–905. Dec. 2008. doi: 10.1109/ICMLA.2008.154.

M. Gashler, D. Ventura, and T. Martinez. Manifold learning by graduated optimization. *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, PP(99):1 –13, 2011a. ISSN 1083-4419. doi: 10.1109/TSMCB.2011.2151187.

M. Gashler, D. Ventura, and T. Martinez. Tangent space guided intelligent neighbor finding. In *The 2011 International Joint Conference on Neural Networks (IJCNN)*, July 2011b.

M. Gashler, D. Ventura, and T. Martinez. Temporal nonlinear dimensionality reduction. In *The 2011 International Joint Conference on Neural Networks (IJCNN)*, July 2011c.

M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. Witten. The WEKA data mining software: an update. *SIGKDD Explor. Newsl.*, 11(1):10–18, 2009. ISSN 1931-0145. doi: 10.1145/1656274.1656278.

S. Roweis and L. Saul. Nonlinear dimensionality reduction by locally linear embedding. *Science*, 290:2323–2326, 2000.

S. Sonnenburg, M.L. Braun, C.S. Ong, S. Bengio, L. Bottou, G. Holmes, Y. LeCun, K.R. Müller, F. Pereira, C.E. Rasmussen, et al. The need for open source software in machine learning. *Journal of Machine Learning Research*, 8:2443–2466, 2007.

J. Tenenbaum, V. de Silva, and J. Langford. A global geometric framework for nonlinear dimensionality reduction. *Science*, 290:2319–2323, 2000.

Journal of Machine Learning Research 12 (2011) 2389-2410 Submitted 3/10; Revised 10/10; Published 7/11

Universality, Characteristic Kernels and RKHS Embedding of Measures

Bharath K. Sriperumbudur BHARATH@GATSBY.UCL.AC.UK
Gatsby Computational Neuroscience Unit
University College London
Alexandra House, 17 Queen Square
London WC1N 3AR, UK

Kenji Fukumizu FUKUMIZU@ISM.AC.JP
The Institute of Statistical Mathematics
10-3 Midori-cho, Tachikawa
Tokyo 190-8562, Japan

Gert R. G. Lanckriet GERT@ECE.UCSD.EDU
Department of Electrical and Computer Engineering
University of California, San Diego
La Jolla, CA 92093-0407, USA

Editor: John Shawe-Taylor

Abstract

Over the last few years, two different notions of positive definite (pd) kernels—universal and characteristic—have been developing in parallel in machine learning: universal kernels are proposed in the context of achieving the Bayes risk by kernel-based classification/regression algorithms while characteristic kernels are introduced in the context of distinguishing probability measures by embedding them into a reproducing kernel Hilbert space (RKHS). However, the relation between these two notions is not well understood. The main contribution of this paper is to clarify the relation between universal and characteristic kernels by presenting a unifying study relating them to RKHS embedding of measures, in addition to clarifying their relation to other common notions of strictly pd, conditionally strictly pd and *integrally strictly pd* kernels. For *radial* kernels on $\mathbb{R}^d$, all these notions are shown to be equivalent.

Keywords: kernel methods, characteristic kernels, Hilbert space embeddings, universal kernels, strictly positive definite kernels, integrally strictly positive definite kernels, conditionally strictly positive definite kernels, translation invariant kernels, radial kernels, binary classification, homogeneity testing

1. Introduction

Kernel methods have been popular in machine learning and pattern analysis for their superior performance on a wide spectrum of learning tasks. They are broadly established as an easy way to construct nonlinear algorithms from linear ones, by embedding data points into higher dimensional reproducing kernel Hilbert spaces (RKHSs) (Schölkopf and Smola, 2002; Shawe-Taylor and Cristianini, 2004). In the regularization approach to learning (Evgeniou et al., 2000), it is well known that kernel-based algorithms (for classification/regression) generally invoke the *representer theorem* (Kimeldorf and Wahba, 1970; Schölkopf et al., 2001) and learn a function in a RKHS that has the

representation,

$$f := \sum_{j \in \mathbb{N}_n} c_j k(\cdot, x_j), \tag{1}$$

where $\mathbb{N}_n := \{1, 2, \ldots, n\}$, $k : X \times X \to \mathbb{R}$ is a symmetric positive definite (pd) kernel on some arbitrary space, X and $\{c_j : j \in \mathbb{N}_n\} \subset \mathbb{R}$ are parameters typically obtained from training data, $\{x_j : j \in \mathbb{N}_n\} \subset X$. As noted in Micchelli et al. (2006), one can ask whether the function, f in (1) approximates any real-valued target function arbitrarily *well* as the number of summands increases without bound. This is an important question to consider because if the answer is affirmative, then the kernel-based learning algorithm can be *consistent* in the sense that for any target function, $f^\star$, the discrepancy between f (which is learned from the training data) and $f^\star$ goes to zero (in some appropriate sense) as the sample size goes to infinity. Since the linear hull of $\{k(\cdot, x) : x \in X\}$ is dense in the RKHS, $\mathcal{H}$ associated with k (Aronszajn, 1950), and assuming that the kernel-based algorithm makes f "converge to an appropriate function" in $\mathcal{H}$ as $n \to \infty$, the above question of approximating $f^\star$ arbitrarily *well* by f in (1) as n goes to infinity is equivalent to the question of whether $\mathcal{H}$ is rich enough to approximate any $f^\star$ arbitrarily *well* (such an RKHS is referred to as a universal RKHS and the corresponding kernel as a universal kernel). Depending on the choice of X, the choice of target function space and the type of approximation, various notions of universality—c-universality (Steinwart, 2001), cc-universality (Micchelli et al., 2006; Caponnetto et al., 2008), c_0-universality (Carmeli et al., 2010; Sriperumbudur et al., 2010a) and L_p-universality (Steinwart and Christmann, 2008; Carmeli et al., 2010)—have been proposed and characterized in literature.

Recently, a seemingly related (to universality) notion of characteristic kernel has been proposed and characterized (Fukumizu et al., 2004, 2008, 2009; Gretton et al., 2007; Sriperumbudur et al., 2008, 2009, 2010b), which has found applications in testing for homogeneity (Gretton et al., 2007), independence (Gretton et al., 2008), conditional independence (Fukumizu et al., 2008), to find the most predictive subspace in regression (Fukumizu et al., 2004), etc. Formally, given the set of all Borel probability measures defined on the topological space X, a measurable and bounded kernel, k is said to be characteristic if

$$\mathbb{P} \mapsto \int_X k(\cdot, x)\, d\mathbb{P}(x), \tag{2}$$

is injective, that is, $\mathbb{P}$ is embedded to a unique element, $\int_X k(\cdot, x)\, d\mathbb{P}(x)$ in $\mathcal{H}$. The motivation to consider such an embedding is that it provides a powerful and straightforward method of dealing with higher-order statistics of random variables, which has been exploited in the above mentioned applications. Gretton et al. (2007) related characteristic and universal kernels by showing that if k is c-universal—see Section 2 for the definition—then it is characteristic. Besides this result, not much is known or understood about the relation between universal and characteristic kernels.

The main contribution of this paper is to clarify the relation between universal and characteristic kernels by presenting a unifying study relating them to RKHS embedding of measures (Suquet, 2009), in addition to clarifying their relation to other common notions of strictly pd, conditionally strictly pd and *integrally strictly pd* kernels, which extends our preliminary study in Sriperumbudur et al. (2010b, Section 3.4). This is done by first reviewing all the existing characterizations for universal and characteristic kernels, which is then used to clarify not only the relation between them but also their relation to other notions of pd kernels (see Section 3). Since the existing characterizations do not explain the complete relationship between all these various notions of pd kernels, we raise open questions in Section 3 about the relationships to be clarified, which are then addressed in Section 4 by deriving new results. In particular, in Section 4, we establish the relation between (a)

c_0-universality and RKHS embedding of finite signed Borel measures, (b) universal and integrally strictly pd kernels, (c) characteristic and conditionally strictly pd kernels and (d) all the above mentioned notions when the pd kernel is *radial* on $\mathbb{R}^d$. A summary of the relation between all these notions of pd kernels is shown in Figure 1, which shows the equivalence between these notions for *radial* kernels on $\mathbb{R}^d$. Supplementary results are collected in appendices. Throughout the paper, we assume X to be a Polish space,[1] the reason for which is discussed in the paragraph following (3).

In the following section, we introduce the notation and collect all definitions that are used throughout the paper.

2. Definitions and Notation

Let X be a topological space. $C(X)$ denotes the space of all continuous real-valued functions on X. $C_b(X)$ is the space of all bounded, continuous real-valued functions on X. For a locally compact Hausdorff space (examples include $\mathbb{R}^d$, infinite discrete sets, topological manifolds, etc.), X, $f \in C(X)$ is said to *vanish at infinity* if for every $\varepsilon > 0$ the set $\{x : |f(x)| \geq \varepsilon\}$ is compact.[2] The class of all continuous f on X which vanish at infinity is denoted as $C_0(X)$. The spaces $C_b(X)$ and $C_0(X)$ are endowed with the uniform norm, $\|\cdot\|_u$ defined as $\|f\|_u := \sup_{x\in X} |f(x)|$ for $f \in C_0(X) \subset C_b(X)$.

Radon measure: A signed Radon measure μ on a Hausdorff space X is a Borel measure on X satisfying

(*i*) $\mu(C) < \infty$ for each compact subset $C \subset X$,

(*ii*) $\mu(B) = \sup\{\mu(C) \,|\, C \subset B, C \text{ compact}\}$ for each B in the Borel σ-algebra of X.

μ is said to be finite if $\|\mu\| := |\mu|(X) < \infty$, where $|\mu|$ is the total-variation of μ. $M_b^+(X)$ denotes the space of all finite Radon measures on X while $M_b(X)$ denotes the space of all finite signed Radon measures on X. The space of all Radon probability measures is denoted as $M_1^+(X) := \{\mu \in M_b^+(X) : \mu(X) = 1\}$. For $\mu \in M_b(X)$, the support of μ is defined as

$$\text{supp}(\mu) = \{x \in X \,|\, \text{for any open set } U \text{ such that } x \in U, |\mu|(U) \neq 0\}. \tag{3}$$

$M_{bc}(X)$ denotes the space of all compactly supported finite signed Radon measures on X. We refer the reader to Berg et al. (1984, Chapter 2) for a general reference on the theory of Radon measures. If X is a Polish space, then by Ulam's theorem, every finite Borel measure is Radon (Dudley, 2002, Theorem 7.1.4). Therefore, for the simplicity of not requiring to distinguish between Borel and Radon measures, throughout the paper, we assume X to be a Polish space.

Positive definite (pd), strictly pd, conditionally strictly pd and integrally strictly pd: A symmetric function $k : X \times X \to \mathbb{R}$ is called positive definite (pd) (*resp.* conditionally pd) if, for all $n \in \mathbb{N}$ (*resp.* $n \geq 2$), $\alpha_1, \ldots, \alpha_n \in \mathbb{R}$ (*resp.* with $\sum_{j=1}^n \alpha_j = 0$) and all $x_1, \ldots, x_n \in X$, we have

$$\sum_{l,j=1}^{n} \alpha_l \alpha_j k(x_l, x_j) \geq 0. \tag{4}$$

1. A topological space (X, τ) is called a Polish space if the topology τ has a countable basis and there exists a complete metric defining τ. An example of a Polish space is $\mathbb{R}^d$ endowed with its usual topology.
2. LCH spaces have a rich supply of continuous functions that vanish outside compact sets—see Tietze extension theorem (Folland, 1999, Theorem 4.34).

Furthermore, k is said to be strictly pd (*resp*. conditionally strictly pd) if, for mutually distinct $x_1,\ldots,x_n \in X$, equality in (4) only holds for $\alpha_1 = \cdots = \alpha_n = 0$.

A measurable, symmetric and bounded kernel, k is said to be integrally strictly pd if

$$\iint_X k(x,y)\,d\mu(x)\,d\mu(y) > 0, \forall \mu \in M_b(X)\backslash\{0\}.$$

This definition is a generalization of *integrally strictly positive definite functions* on $\mathbb{R}^d$ (Stewart, 1976, Section 6): $\iint_{\mathbb{R}^d} k(x,y)f(x)f(y)\,dx\,dy > 0$ for all $f \in L^2(\mathbb{R}^d)$, which is the strictly positive definiteness of the integral operator given by the kernel.

c-, cc-, c_0- and L_p-universal kernels: A continuous pd kernel k on a compact Hausdorff space X is called c-universal if the RKHS, $\mathcal{H}$ induced by k is dense in $C(X)$ w.r.t. the uniform norm, that is, for every function $g \in C(X)$ and all $\varepsilon > 0$, there exists an $f \in \mathcal{H}$ such that $\|f-g\|_u \leq \varepsilon$ (Steinwart, 2001).

A continuous pd kernel k on a Hausdorff space X is said to be cc-universal if the RKHS, $\mathcal{H}$ induced by k is dense in $C(X)$ endowed with the topology of compact convergence, that is, for any compact set $Z \subset X$, for any $g \in C(Z)$ and all $\varepsilon > 0$, there exists an $f \in \mathcal{H}|_Z$ such that $\|f-g\|_u \leq \varepsilon$, where $\mathcal{H}|_Z := \{f|_Z : f \in \mathcal{H}\}$ is the restriction of $\mathcal{H}$ to Z and $f|_Z$ is the restriction of f to Z (Carmeli et al., 2010; Sriperumbudur et al., 2010a).

A pd kernel, k is said to be a c_0-kernel if it is bounded with $k(\cdot,x) \in C_0(X), \forall x \in X$, where X is a locally compact Hausdorff (LCH) space. A c_0-kernel on an LCH space, X is said to be c_0-universal if the RKHS, $\mathcal{H}$ induced by k is dense in $C_0(X)$ w.r.t. the uniform norm (Carmeli et al., 2010; Sriperumbudur et al., 2010a).[3]

A measurable and bounded kernel, k defined on a Hausdorff space, X is said to be L_p-universal if the RKHS, $\mathcal{H}$ induced by k is dense in $L^p(X,\mu)$ w.r.t. the *p-norm*, defined as

$$\|f\|_p := \left(\int_X |f(x)|^p\,d\mu(x)\right)^{1/p},$$

for all Borel probability measures, μ, defined on X and some $p \in [1,\infty)$. Here $L^p(X,\mu)$ is the Banach space of p-integrable μ-measurable functions on X (Steinwart and Christmann, 2008).

We would like to stress that in the above definitions of universality, the assumptions on k ensure that the associated RKHS, $\mathcal{H}$ is continuously included in the target space. Steinwart and Christmann (2008, Lemma 4.28) showed that k is bounded and $k(\cdot,x)$ is continuous for all $x \in X$ (X being a topological space) if and only if every $f \in \mathcal{H}$ is bounded and continuous. In addition, the inclusion $\mathrm{id} : \mathcal{H} \to C_b(X)$ is continuous. Similarly, by modifying the proof of Lemma 4.28 in Steinwart and Christmann (2008), it can be easily shown that k is bounded and $k(\cdot,x) \in C_0(X), \forall x \in X$ (X being an LCH space) if and only if every $f \in \mathcal{H}$ is in $C_0(X)$, and the inclusion $\mathrm{id} : \mathcal{H} \to C_0(X)$ can be shown to be continuous (also see Carmeli et al., 2010, Proposition 2.2). Steinwart and Christmann (2008, Theorem 4.26) showed that if k is measurable and bounded on a measurable space X, then

3. Note that cc-universality (*resp*. c-universality) deals with X being a non-compact (*resp*. compact) Hausdorff space, whereas c_0-universality requires X to be an LCH space. While X being Hausdorff ensures that it has an abundance of compact subsets (as required in cc-universality), the stronger condition of X being an LCH space ensures that it has an abundance of continuous functions that vanish outside compact sets (see footnote 2). In addition, this choice of X being an LCH space ensures the existence of topological dual of $C_0(X)$ through the Riesz representation theorem, which is required in the characterization of c_0-universality. See Proposition 2 in Section 4 for details.

$\mathcal{H}$ consists of p-integrable (w.r.t. any Borel probability measure, μ) functions and the inclusion $\mathrm{id} : \mathcal{H} \to L^p(X,\mu)$ is continuous for some $p \in [1,\infty)$.

Characteristic kernel: A bounded measurable kernel, k is said to be characteristic if $\mu \mapsto \int_X k(\cdot,x)\,d\mu(x)$ is injective, where μ is a Borel probability measure on X.

Translation invariant and Radial kernels on $\mathbb{R}^d$: A pd kernel, $k : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$ is said to be translation invariant if $k(x,y) = \psi(x-y)$, where ψ is a pd function. If k is bounded and continuous, then by Bochner's theorem (Wendland, 2005, Theorem 6.6), $\psi \in C_b(\mathbb{R}^d)$ is the Fourier transform of $\Lambda \in M_b^+(\mathbb{R}^d)$, that is,

$$\psi(x) = \int_{\mathbb{R}^d} e^{-\sqrt{-1}x^T\omega}\,d\Lambda(\omega),\, x \in \mathbb{R}^d. \tag{5}$$

A bounded continuous kernel, k is said to be radial on $\mathbb{R}^d \times \mathbb{R}^d$ if there exists $\nu \in M_b^+([0,\infty))$ such that

$$k(x,y) = \int_{[0,\infty)} e^{-t\|x-y\|_2^2}\,d\nu(t),\, x,y \in \mathbb{R}^d. \tag{6}$$

It is easy to see that a radial kernel is also bounded translation invariant on $\mathbb{R}^d$ (see Appendix A). Examples of radial kernels include the Gaussian kernel, $k(x,y) = e^{-\sigma\|x-y\|_2^2}$, $\sigma > 0$; inverse multiquadrics, $k(x,y) = (c + \|x-y\|_2^2)^{-\beta}$, $\beta > d/2$, etc.

A continuous pd kernel is said to be translation invariant on $\mathbb{T}^d := [0,2\pi)^d$ if $k(x,y) = \psi((x-y)_{\mathrm{mod}\,2\pi})$, where $\psi \in C(\mathbb{T}^d)$ is such that

$$\psi(x) = \sum_{n\in\mathbb{Z}^d} A_\psi(n) e^{\sqrt{-1}x^T n},\, x \in \mathbb{T}^d, \tag{7}$$

with $A_\psi : \mathbb{Z}^d \to \mathbb{R}_+$, $A_\psi(-n) = A_\psi(n)$ and $\sum_{n\in\mathbb{Z}^d} A_\psi(n) < \infty$.

3. Relation Between Various Notions of Positive Definite Kernels Based on Known Characterizations

In this section, we review existing results on the characterization of universal and characteristic kernels, which are then used to clarify not only the relation between them but also their relation to other notions like strictly pd, conditionally strictly pd and integrally strictly pd kernels. In Section 3.1, we discuss various notions of universality, review all their existing characterizations and then summarize the relation between them. In Section 3.2, we discuss and summarize the relation between characteristic and universal kernels based on their existing characterizations. The relation of universal and characteristic kernels to strictly pd, conditionally strictly pd and integrally strictly pd kernels are summarized in Section 3.3. Since the existing characterizations do not explain the complete relationship between all these various notions of pd kernels, we raise questions at the end of each subsection that need to be addressed to obtain a complete understanding of the relationships between all these notions. A summary of the relationships between various notions of pd kernels based on the existing characterizations is shown in Figure 1.

Before proceeding further, we would like to highlight a possible confusion that can raise while comparing these various notions of pd kernels. Suppose we would like to compare c_0-universal vs. characteristic kernels, that is, (a) Is a c_0-universal kernel characteristic? (b) Is the converse true? While (a) is a valid question, answering (b) trivially yields that characteristic kernels are not c_0-

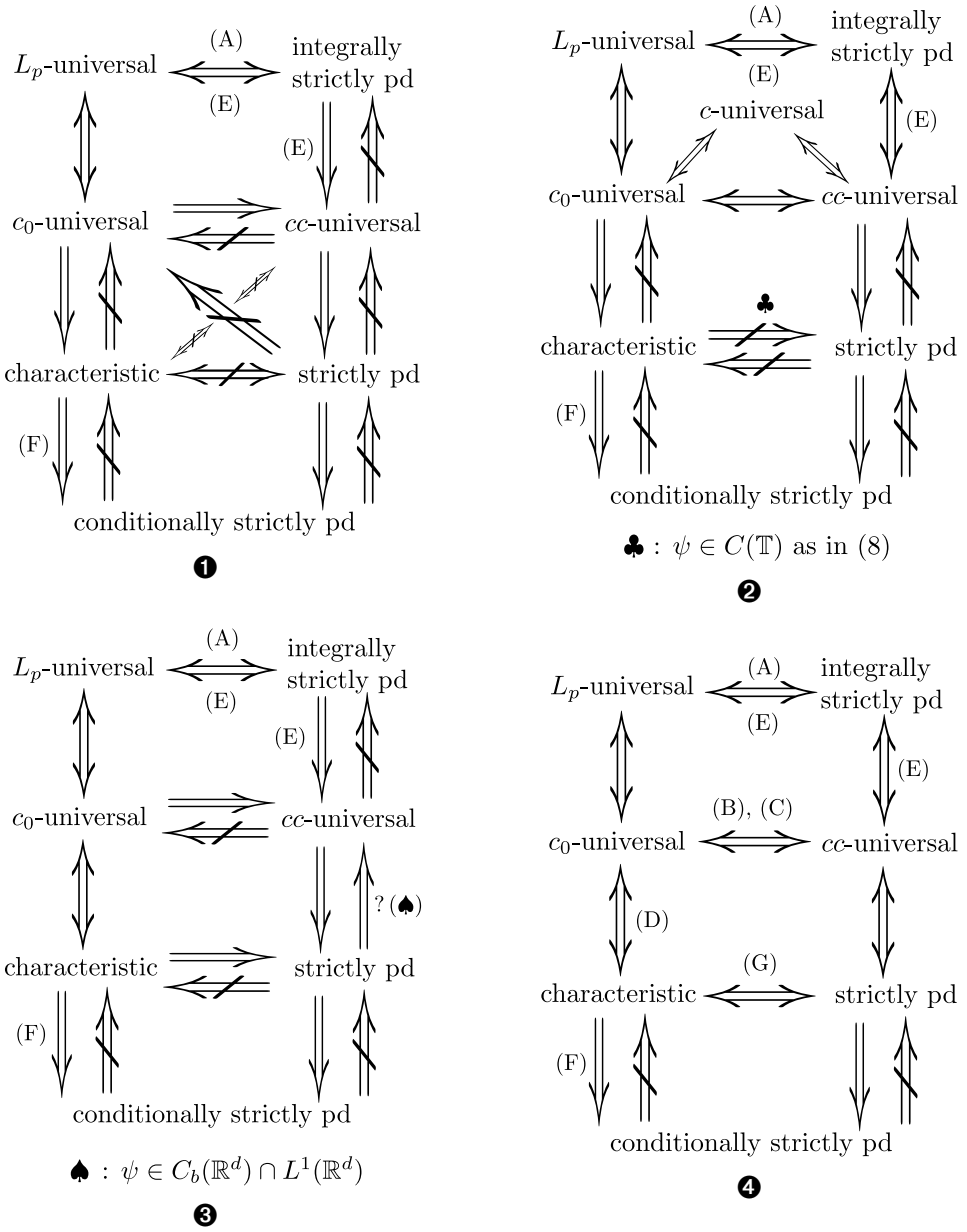


Figure 1: Summary of the relations between various families of c_0-kernels: The implications shown without any reference are based on the review of existing results (see Section 3) while the ones with a reference are based on new results derived in Section 4 that addresses the open questions (A)–(G). The implications which are still open are shown with "?". ❶ X is an LCH space. ❷ The implications shown hold for any compact Hausdorff space, X. When $X = \mathbb{T}$ and k is continuous and translation invariant on $\mathbb{T}$—see (7)—then k being characteristic implies it is strictly pd, which is shown as ♣. ❸ The implications shown hold for bounded continuous translation invariant kernels on $\mathbb{R}^d$—see (5). If $\psi \in C_b(\mathbb{R}^d) \cap L^1(\mathbb{R}^d)$, then the implication shown as (♠) holds, that is, strictly pd kernels are cc-universal. Otherwise, it is not clear whether the implication holds. ❹ Radial kernels on $\mathbb{R}^d$—see (6).

universal. This is because k need not be a c_0-kernel for it to be characteristic.[4] Therefore, to make a non-trivial comparison between characteristic and c_0-universal kernels, it is important that we assume k to be a c_0-kernel before answering the questions in (a) and (b). In extending this reasoning for the non-trivial comparison of any two notions of pd kernels, it is important to assume that k satisfies the strongest possible condition. Therefore, in order to present a concise summary of the relationships between these various notions, in Figure 1, we assume k to be a c_0-kernel—this is the strongest condition to be satisfied in order to compare all these notions of pd kernels.

3.1 Relation Between Various Notions of Universality

As mentioned before, a universal kernel is such that its corresponding RKHS, $\mathcal{H}$ is rich enough to approximate any target function (belonging to some target space) arbitrarily well. Therefore, depending on the choice of X, the choice of target space and the type of approximation, various notions of universality—c, cc, c_0 and L^p—have been proposed. In the following, we review the existing characterizations for all these notions of universal kernels and summarize the relation between them.

c-universality: Steinwart (2001) proposed the notion of c-universality, wherein X is a compact metric space with $C(X)$ being the target space and $\mathcal{H}$ being dense in $C(X)$ w.r.t. the uniform norm. By applying the Stone-Weierstraß theorem (Folland, 1999, Theorem 4.45), Steinwart (2001, Theorem 9) provided sufficient conditions for a kernel to be c-universal—a continuous kernel, k on a compact metric space, X is c-universal if the following hold: (a) $k(x,x) > 0, \forall x \in X$, (b) there exists an injective feature map $\Phi : X \to \ell_2$ of k with $\Phi(x) = \{\Phi_n(x)\}_{n\in\mathbb{N}}$ and (c) $\text{span}\{\Phi_n : n \in \mathbb{N}\}$ is an algebra—using which the Gaussian kernel is shown to be c-universal on every compact subset of $\mathbb{R}^d$. Micchelli et al. (2006, Proposition 1) related c-universality to the injective RKHS embedding of finite signed Borel measures by showing that k is c-universal if and only if

$$\mu \mapsto \int_X k(\cdot,x)\, d\mu(x), \mu \in M_b(X), \tag{8}$$

is injective.

cc-universality: One limitation in the notion of universality considered by Steinwart (2001) is that X is assumed to be compact, which excludes many interesting spaces, such as $\mathbb{R}^d$ and infinite discrete sets. To overcome this limitation, Carmeli et al. (2010, Definition 4.1, Theorem 4.3) and Sriperumbudur et al. (2010a) introduced the notion of cc-universality which can handle non-compact Hausdorff spaces, X. Carmeli et al. (2010, Proposition 2.3, Theorems 4.3 and 4.4) showed that a bounded continuous pd kernel, k is cc-universal if and only if the following embedding is injective for all $\mu \in M_{bc}(X)$ and some $p \in [1,\infty)$:

$$f \mapsto \int_X k(\cdot,x) f(x)\, d\mu(x), f \in L^p(X,\mu). \tag{9}$$

In addition, Carmeli et al. (2010, Remark 4.1) showed that k being cc-universal is equivalent to it being universal in the sense of Micchelli et al. (2006) and Caponnetto et al. (2008): for any compact $Z \subset X$, the set $K(Z) := \overline{\text{span}}\{k(\cdot,y) : y \in Z\}$ is dense in $C(Z)$ in the uniform norm, which is shown by

4. Let k_1 be a characteristic kernel on $\mathbb{R}$. Define $k_2(x,y) = 1$ if $x = y \in \mathbb{R}$ and $k_2(x,y) = 0$ if $x \neq y \in \mathbb{R}$. Clearly k_2 is not continuous and therefore $k_1 + k_2$ is not a c_0-kernel, even if k_1 is a c_0-kernel. However, it is easy to verify that $k_1 + k_2$ is characteristic.

Micchelli et al. (2006, Proposition 1) to be equivalent to the following embedding being injective:

$$\mu \mapsto \int_Z k(\cdot,x)\, d\mu(x), \mu \in M_b(Z). \tag{10}$$

Since (10) holds for any compact $Z \subset X$, the universality in the sense of Micchelli et al. and Caponnetto et al. is equivalent to the following embedding being injective:

$$\mu \mapsto \int_X k(\cdot,x)\, d\mu(x), \mu \in M_{bc}(X). \tag{11}$$

Therefore, k being *cc*-universal is equivalent to the injectivity of (11)—in Section 4, we present a more direct proof of this result (see Remark 3). It is clear from the definitions of *c*- and *cc*-universality that these notions are equivalent when X is compact, which also follows from their characterizations in (8) and (11).

As special cases, Micchelli et al. (2006, Propositions 14, Theorem 17) showed that a translation invariant kernel on $\mathbb{R}^d$ is *cc*-universal if $\text{supp}(\Lambda)$ is a uniqueness subset[5] of $\mathbb{C}^d$, while a radial kernel on $\mathbb{R}^d$ is *cc*-universal if and only if $\text{supp}(\nu) \neq \{0\}$—see (5) and (6) for the definitions of Λ and ν. Using these characterizations, many popular kernels on $\mathbb{R}^d$ are shown to be *cc*-universal (Micchelli et al., 2006, Section 4): Gaussian, Laplacian, B_{2l+1}-spline, sinc kernel, etc.

c_0-universality: Although *cc*-universality solves the limitation of *c*-universality by handling non-compact X, the topology of compact convergence considered in *cc*-universality is weaker than the topology of uniform convergence, that is, a sequence of functions, $\{f_n\} \subset C(X)$ converging to $f \in C(X)$ in the topology of uniform convergence ensures that they converge in the topology of compact convergence but not vice-versa. So, the natural question to ask is whether we can characterize $\mathcal{H}$ that are rich enough to approximate any $f^\star$ on non-compact X in a stronger sense, that is, uniformly, by some $g \in \mathcal{H}$. Carmeli et al. (2010, Definition 2.2, Theorem 4.1) and Sriperumbudur et al. (2010a) answered this through the notion of c_0-universality, wherein X is an LCH space with $C_0(X)$ being the target space and $\mathcal{H}$ being dense in $C_0(X)$ w.r.t. the uniform norm (note that a notion of universality that is stronger than c_0-universality can be defined by choosing X to be a Hausdorff space, $C_b(X)$ to be the target space and $\mathcal{H}$ being dense in $C_b(X)$ w.r.t. the uniform norm. However, this notion of universality does not enjoy a nice characterization as c_0-universality—see (12) and (13) for the characterization of c_0-universality—and therefore, we did not include it in our study of relationships between various notions of pd kernels. See Appendix C for details).

Carmeli et al. (2010, Theorem 4.1) showed that a c_0-kernel k is c_0-universal if and only if it is L_p-universal, which by Proposition 2.3 and Theorem 4.2 of Carmeli et al. (2010) is equivalent to the injectivity of the following embedding for all $\mu \in M_b(X)$ and some $p \in [1,\infty)$:

$$f \mapsto \int_X k(\cdot,x) f(x)\, d\mu(x), f \in L^p(X,\mu). \tag{12}$$

We provide an alternate characterization for c_0-universality in Section 4 (see Proposition 2) that k is c_0-universal if and only if the following embedding is injective:

$$\mu \mapsto \int_X k(\cdot,x)\, d\mu(x), \mu \in M_b(X). \tag{13}$$

5. A subset S of $\mathbb{C}^d$ is a uniqueness set if an entire function on $\mathbb{C}^d$ vanishes on S then it is everywhere zero on $\mathbb{C}^d$. Non-empty interior is sufficient for a set to be a uniqueness set.

As a special case, Carmeli et al. (2010, Proposition 5.6) showed that a translation invariant k on $\mathbb{R}^d$ is c_0-universal if and only if $\text{supp}(\Lambda) = \mathbb{R}^d$. Examples of c_0-universal kernels on $\mathbb{R}^d$ include the Gaussian, Laplacian, B_{2l+1}-spline, inverse multiquadrics, Matérn class, etc.

Summary: The following statements summarize the relation between various notions of universality, which are depicted in Figure 1.

- c- and cc-universality are related to the injective RKHS embedding of finite signed Borel measures, as shown in (8) and (11).
- For c_0-kernels defined on an LCH space X, c_0-universality implies cc-universality, which follows from (9) and (12). The converse is however not true as a bounded continuous translation invariant c_0-kernel on $\mathbb{R}^d$ is c_0-universal if and only if $\text{supp}(\Lambda) = \mathbb{R}^d$ while $(\text{supp}(\Lambda))^\circ \neq \emptyset$ is sufficient for cc-universality, where A° represents the interior of A.
- When X is compact, then c-, cc- and c_0-universality are equivalent.
- For an LCH space X, a c_0-kernel is c_0-universal if and only if it is L_p-universal.
- If k is a radial kernel on $\mathbb{R}^d$, then k is cc-universal if and only if $\text{supp}(\nu) \neq \{0\}$.

Open questions: The following relationships need to be clarified, which we do in Section 4.

(A) As mentioned in the summary, c- and cc-universality are related to the injective RKHS embedding of finite signed Borel measures. However, the relation between c_0-universality and the injective RKHS embedding of finite signed Borel measures as shown in (13) is not clear, which we clarify in Section 4.1.

(B) For c_0-kernels defined on an LCH space X (that is not compact), it is clear from the summary that c_0-universality implies cc-universality. Is there a case for which cc-universality implies c_0-universality? We address this in Section 4.3.

(C) While cc-universality is characterized for radial kernels on $\mathbb{R}^d$, the characterization of c_0-universality for radial kernels is not known. In Section 4.3, we provide a characterization of c_0-universality for radial kernels on $\mathbb{R}^d$ and then establish the relation between c_0-universality and cc-universality for such kernels.

3.2 Relation Between Characteristic and Universal Kernels

In this section, we comprehensively clarify the relation between various notions of universality and characteristic kernels, based on already existing characterizations for characteristic kernels and the results summarized in Section 3.1 for universal kernels.

c-universal kernels vs. Characteristic kernels: Gretton et al. (2007) related universal and characteristic kernels by showing that if k is c-universal, then it is characteristic. In our preliminary study in Sriperumbudur et al. (2010b, Section 3.4), we showed that the converse is not true: as an example, a translation invariant kernel, k on $\mathbb{T}^d \times \mathbb{T}^d$ is characteristic if and only if $A_\psi(0) \geq 0$, $A_\psi(n) > 0, \forall\, n \in \mathbb{Z}_+^d$ while it is universal if and only if $A_\psi(n) > 0, \forall\, n \in \mathbb{Z}^d$.

cc-universal kernels vs. Characteristic kernels: cc-universal kernels on a non-compact Hausdorff space need not be characteristic: for example, a bounded continuous translation invariant

kernel on $\mathbb{R}^d$ is *cc*-universal if $(\text{supp}(\Lambda))^\circ \neq \emptyset$ (see the summary of Section 3.1) while it is characteristic if and only if $\text{supp}(\Lambda) = \mathbb{R}^d$ (Sriperumbudur et al., 2008, Theorem 7). Although, this example shows that a bounded continuous translation invariant kernel on $\mathbb{R}^d$ is *cc*-universal if it is characteristic, it is not clear whether such a relation holds on a general non-compact Hausdorff space (not necessarily $\mathbb{R}^d$). The following example shows that continuous kernels that are characteristic on non-compact Hausdorff space, X also need not be *cc*-universal.

Example 1 *Let $X = \mathbb{N}$. Define $k(x,y) = \delta_{xy}, x,y \in X\backslash\{1\}$, $k(x,1) = 0$ for any $x \in X$, where δ represents the Kronecker delta. Suppose $\mu = \delta_1 \in M_{bc}(X)\backslash\{0\}$, where δ_j represents the Dirac measure at j. Then $\|\int_X k(\cdot,x)\,d\mu(x)\|^2_{\mathcal{H}} = \|k(\cdot,1)\|^2_{\mathcal{H}} = k(1,1) = 0$, which means there exists $\mu \in M_{bc}(X)\backslash\{0\}$ such that $\int_X k(\cdot,x)\,d\mu(x) = 0$, that is, (11) is not injective and therefore k is not cc-universal. However, k is characteristic as we show below.*

Let $\mathbb{P}$ and $\mathbb{Q}$ be probability measures on X such that $\mathbb{P} = \sum_{j\in\mathbb{N}} p_j\delta_j$, $\mathbb{Q} = \sum_{j\in\mathbb{N}} q_j\delta_j$ with $p_j \geq 0, q_j \geq 0$ for all $j \in \mathbb{N}$ and $\sum_{j\in\mathbb{N}} p_j = \sum_{j\in\mathbb{N}} q_j = 1$. Consider

$$\begin{aligned} B &:= \left\| \int_X k(\cdot,x)\,d(\mathbb{P}-\mathbb{Q})(x)\right\|^2_{\mathcal{H}} = \left\| \sum_{j\in\mathbb{N}} (p_j - q_j)k(\cdot,j)\right\|^2_{\mathcal{H}} = \sum_{l,j\in\mathbb{N}} (p_l - q_l)(p_j - q_j)k(l,j) \\ &= (p_1-q_1)^2k(1,1) + 2(p_1-q_1)\sum_{j\in\mathbb{N}\backslash\{1\}} (p_j-q_j)k(j,1) + \sum_{l,j\in\mathbb{N}\backslash\{1\}} (p_j-q_j)(p_l-q_l)k(j,l) \\ &= \sum_{j\in\mathbb{N}\backslash\{1\}} (p_j-q_j)^2. \end{aligned}$$

Suppose $B = 0$, which means $p_j = q_j, \forall\, j \in \mathbb{N}\backslash\{1\}$. Since $\sum_{j\in\mathbb{N}} p_j = \sum_{j\in\mathbb{N}} q_j = 1$, we have $p_1 = q_1$ and so $\mathbb{P} = \mathbb{Q}$, that is, (2) is injective and therefore k is characteristic.

c_0-universal kernels vs. Characteristic kernels: Fukumizu et al. (2008, 2009) have shown that a measurable and bounded kernel, k is characteristic if and only if $\mathcal{H} + \mathbb{R}$ (the direct sum of $\mathcal{H}$ and $\mathbb{R}$ is defined as $\mathcal{H} + \mathbb{R} := \{f + c : f \in \mathcal{H}, c \in \mathbb{R}\}$) is dense in $L^p(X,\mathbb{P})$ for all $\mathbb{P} \in M_1^+(X)$ and for some $p \in [1,\infty)$. Using this, it is easy to see that if $\mathcal{H}$ is dense in $L^p(X,\mathbb{P})$ for all $\mathbb{P} \in M_1^+(X)$ and for some $p \in [1,\infty)$, then k is characteristic. Based on the results summarized in Section 3.1, it is clear that for an LCH space, X, if k is c_0-universal, which means k is L_p-universal, then $\mathcal{H}$ is dense in $L^p(X,\mathbb{P})$ for all $\mathbb{P} \in M_1^+(X)$ and for some $p \in [1,\infty)$ and therefore is characteristic. In Section 4, we provide an alternate proof for this relation between c_0-universal and characteristic kernels by answering (A). Clearly, the converse is not true, that is, a c_0-kernel that is characteristic need not be c_0-universal (see Proposition 4 and footnote 8). However, for bounded continuous translation invariant kernels on $\mathbb{R}^d$, the converse is true, that is, a translation invariant c_0-kernel that is characteristic[6] is also c_0-universal. This is because of the fact that a translation invariant kernel on $\mathbb{R}^d$ is characteristic if and only if $\text{supp}(\Lambda) = \mathbb{R}^d$ (Sriperumbudur et al., 2008, Theorem 7), which is also the same characterization summarized in Section 3.1 for c_0-universal kernels.

Summary: The following statements summarize the relation between universal and characteristic kernels, which are depicted in Figure 1.

6. Let $k(x,y) = \psi(x-y)$ be a bounded continuous translation invariant kernel on $\mathbb{R}^d$, which by Bochner's theorem is of the form in (5). Suppose $\psi \in L^1(\mathbb{R}^d)$. Then by the Fourier inversion theorem (Dudley, 2002, Theorem 9.5.4), Λ has a density, $\hat{\psi}$ w.r.t. the Lebesgue measure such that $\hat{\psi} \in L^1(\mathbb{R}^d)$. Therefore, since ψ is the Fourier transform of $\hat{\psi}$, by the Riemann-Lebesgue lemma (Rudin, 1991, Theorem 7.5), $\psi \in C_0(\mathbb{R}^d)$, that is, k is a c_0-kernel. Most of the well-known characteristic kernels satisfy the condition of $\psi \in L^1(\mathbb{R}^d)$ and therefore are c_0-kernels. This means, for all practical purposes, we can assume bounded continuous translation invariant kernels to be c_0-kernels.

- For c_0-kernels defined on an LCH space, X, L_p-universal $\Leftrightarrow$ c_0-universal $\Rightarrow$ characteristic. But in general, c_0-kernels that are characteristic need not be c_0-universal. However, for translation invariant kernels on $\mathbb{R}^d$, c_0-universal $\Leftrightarrow$ characteristic.
- When X is compact, c-universal $\Rightarrow$ characteristic but not vice-versa.
- For translation invariant kernels on $\mathbb{R}^d$, characteristic $\Rightarrow$ cc-universal but not vice-versa. However, on general non-compact Hausdorff spaces, continuous kernels that are characteristic need not be cc-universal.

Open questions: The following relationship need to be clarified, which we do in Section 4.

(D) While the relation between universal and characteristic kernels that are translation invariant on $\mathbb{R}^d$ is clear (see the summary above), the characterization of characteristic and c_0-universal kernels that are radial on $\mathbb{R}^d$ is not known and therefore the relation between characteristic and universal kernels that are radial on $\mathbb{R}^d$ is not clear. We address this in Section 4.3.

3.3 Relation of Universal and Characteristic Kernels to Strictly PD, Integrally Strictly PD and Conditionally Strictly PD Kernels

In this section, we relate characteristic kernels and various notions of universal kernels to strictly pd, integrally strictly pd and conditionally strictly pd kernels. Before that, we summarize the relation between strictly pd, integrally strictly pd and conditionally strictly pd kernels. In Sriperumbudur et al. (2010b, Section 3.4), we showed that integrally strictly pd kernels are strictly pd. The converse is not true, which follows from Steinwart and Christmann (2008, Proposition 4.60, Theorem 4.62). However, if X is a finite set, then k being strictly pd also implies it is integrally strictly pd. From the definitions of strictly pd and conditionally strictly pd kernels, it is clear that a strictly pd kernel is conditionally strictly pd but not vice-versa.

Universal kernels vs. Strictly pd kernels: Carmeli et al. (2010, Corollary 4.3) showed that cc-universal kernels are strictly pd, which means c_0-universal kernels are also strictly pd (as c_0-universal $\Rightarrow$ cc-universal from Section 3.1). This means, when X is compact Hausdorff, c-universal kernels are strictly pd, which matches with the result in Steinwart and Christmann (2008, Definition 4.53, Proposition 4.54, Example 4.11).

Conversely, a strictly pd c_0-kernel on an LCH space need not be c_0-universal. This follows from Theorem 4.62 in Steinwart and Christmann (2008) which shows that there exists a bounded strictly pd kernel, k on $X := \mathbb{N} \cup \{0\}$ with $k(\cdot, x) \in C_0(X)$, $\forall x \in X$ such that k is not L_p-universal (which from the summary of Section 3.1 means k is not c_0-universal). Similarly, when X is compact, the converse is not true, that is, continuous strictly pd kernels need not be c-universal which follows from the results due to Dahmen and Micchelli (1987) and Pinkus (2004) for Taylor kernels (Steinwart and Christmann, 2008, Lemma 4.8, Corollary 4.57)—refer to Steinwart and Christmann (2008, Section 4.7, p. 161) for more details.[7] Therefore, it is evident that a continuous strictly pd kernel is in general not cc-universal on an Hausdorff space. However, for translation invariant kernels that are continuous, bounded and integrable on $\mathbb{R}^d$, that is, $k(x,y) = \psi(x-y), x, y \in \mathbb{R}^d$, where $\psi \in$

7. Another example of continuous strictly pd kernels that are not c-universal is as follows. Using the technique in the proof of Theorem 14 of Sriperumbudur et al. (2010b), it can be shown that a continuous translation invariant kernel on $\mathbb{T} \times \mathbb{T}$ is c-universal if and only if $A_\psi(n) > 0, \forall n \in \mathbb{Z}$. Therefore, by Theorem 8 (see Appendix B), a strictly pd kernel on $\mathbb{T}$ need not be c-universal.

$C_b(\mathbb{R}^d) \cap L^1(\mathbb{R}^d)$, strictly pd implies *cc*-universality. This follows from Theorem 6.11 and Corollary 6.12 of Wendland (2005) that if $\psi \in C_b(\mathbb{R}^d) \cap L^1(\mathbb{R}^d)$ is strictly pd, then $(\text{supp}(\Lambda))^\circ \neq \emptyset$, which from the summary of Section 3.1 means k is *cc*-universal. Similarly, when the kernel is radial on $\mathbb{R}^d$, then strictly pd kernels are *cc*-universal. This follows from Theorem 7.14 of Wendland (2005), which shows that a radial kernel on $\mathbb{R}^d$ is strictly pd if and only if $\text{supp}(\nu) \neq \{0\}$, and therefore *cc*-universal (from the summary of Section 3.1). On the other hand, when X is finite, all these notions of universal and strictly pd kernels are equivalent, which follows from the result due to Carmeli et al. (2010, Corollary 4.3) that *cc*-universal and strictly pd kernels are the same when X is finite.

Characteristic kernels vs. Strictly pd kernels: Since characteristic kernels that are c_0- and translation invariant on $\mathbb{R}^d$ are equivalent to c_0-universal kernels (see the summary of Section 3.2), it is clear that they are strictly pd. However, the converse is not true: for example, the sinc-squared kernel, $k(x,y) = \frac{\sin^2(\sigma(x-y))}{(x-y)^2}$ on $\mathbb{R}$, which has $\text{supp}(\Lambda) = [-\sigma,\sigma] \subsetneq \mathbb{R}$ is strictly pd (Wendland, 2005, Theorem 6.11), while it is not characteristic. Based on Example 1, it can be shown that in general, characteristic kernels on a non-compact space (not necessarily $\mathbb{R}^d$) need not be strictly pd: in Example 1, k is characteristic but is not strictly pd because for $(a_1,\ldots,a_n) = (1,0,\ldots,0)$ and $(x_1,\ldots,x_n) = (1,\ldots,n)$, we have $\sum_{l,j=1}^n a_l a_j k(x_l,x_j) = a_1^2 k(1,1) + 2a_1 \sum_{j=2}^n a_j k(j,1) + \sum_{j=2}^n a_j^2 = 0$. Note that Example 1 holds even if X is a compact subset of $\mathbb{N}$. Therefore, when X is compact Hausdorff, a characteristic kernel need not be strictly pd. However, for translation invariant kernels on $\mathbb{T}$, a characteristic kernel is also strictly pd, while the converse is not true: Fukumizu et al. (2009, Theorem 8) and Sriperumbudur et al. (2010b, Theorem 14) have shown that k on $\mathbb{T}\times\mathbb{T}$ is characteristic if and only if $A_\psi(0) \geq 0, A_\psi(n) > 0, \forall\, n \in \mathbb{Z}\backslash\{0\}$, which by Theorem 8 (see Appendix B) is strictly pd, while the converse is clearly not true.

Characteristic kernels vs. Integrally strictly pd kernels: In Sriperumbudur et al. (2009, Theorem 4) and Sriperumbudur et al. (2010b, Theorem 7), we have shown that integrally strictly pd kernels are characteristic, while the converse in general is not true.[8] When k is bounded continuous and translation invariant on $\mathbb{R}^d$, however the converse holds, which is due to the fact that if k is characteristic, then $\text{supp}(\Lambda) = \mathbb{R}^d$ (Sriperumbudur et al., 2008, Theorem 7), which ensures that k is integrally strictly pd.

Summary: The following statements summarize the relation of universal and characteristic kernels to strictly pd, integrally strictly pd and conditionally strictly pd kernels, which are depicted in Figure 1.

- *c*-, *cc*- and c_0-universal kernels are strictly pd and are therefore conditionally strictly pd, while the converse in general is not true. When X is finite, then *c*-, *cc*- and c_0-universal kernels are equivalent to strictly pd kernels.

- Bounded, continuous, integrable, strictly pd translation invariant kernels on $\mathbb{R}^d$ are *cc*-universal. Radial kernels on $\mathbb{R}^d$ are strictly pd if and only if they are *cc*-universal.

- For a general non-compact Hausdorff space, characteristic kernels need not be strictly pd and vice-versa. However, bounded continuous translation invariant kernels on $\mathbb{R}^d$ or $\mathbb{T}$ that are characteristic are strictly pd but the converse is not true.

8. By Example 1, it is clear that for $\mu = \delta_1 \in M_b(X)\backslash\{0\}$, $\int\int_X k(x,y)\,d\mu(x)\,d\mu(y) = k(1,1) = 0$, where δ_1 represents the Dirac measure at 1. Therefore k is not integrally strictly pd but is characteristic.

- Integrally strictly pd kernels are characteristic. Though the converse is not true in general, it holds if the kernel is bounded, continuous and translation invariant on $\mathbb{R}^d$.

Open questions: The following questions need to be clarified, which is done in Section 4.

(E) While the relation of universal kernels to strictly pd and conditionally strictly pd kernels is clear from the above summary, the relation between universal and integrally strictly pd kernels is not known, which we establish in Section 4.2.

(F) When X is a finite set, it is easy to see that characteristic and conditionally strictly pd kernels are equivalent (see Section 4.4). However, their relationship is not clear for a general measurable space, which we clarify in Section 4.4.

(G) As summarized above, radial kernels on $\mathbb{R}^d$ are strictly pd if and only if they are *cc*-universal. However, the relation between all the other notions of pd kernels—c_0-universal, characteristic, strictly pd and integrally strictly pd—is not known, which is addressed in Section 4.3.

4. Relation Between Various Notions of Positive Definite Kernels: New Results

In this section, we address the open questions, (A)–(G) mentioned in Section 3 to understand the complete relationship between various notions of positive definite kernels.

4.1 c_0-universality and RKHS Embedding of Measures

As mentioned in Section 3.1, Micchelli et al. (2006) have established the relation of *c*-universality and *cc*-universality to injective RKHS embedding of finite signed Borel measures—shown in (8) and (11)—through a simple application of the Hahn-Banach theorem (see Theorem 1). The following result (also see Suquet, 2009, Remark 1.1) in Proposition 2 provides a measure embedding characterization—shown in (13)—for c_0-universality, which is also obtained as a simple application of the Hahn-Banach theorem, and therefore addresses the open question in (A). Before we state Proposition 2, we present the Hahn-Banach theorem, which we quote from Rudin (1991, Theorem 3.5 and the remark following Theorem 3.5).

Theorem 1 (Hahn-Banach) *Suppose A is a subspace of a locally convex topological vector space Y. Then A is dense in Y if and only if $A^\perp = \{0\}$, where*

$$A^\perp := \{T \in Y' : \forall x \in A,\ T(x) = 0\}.$$

The following result, which presents a necessary and sufficient condition for k to be c_0-universal hinges on the above theorem, where we choose A to be the RKHS, $\mathcal{H}$ and Y to be $C_0(X)$ for which Y' is known through the Riesz representation theorem (Folland, 1999, Theorem 7.17).

Proposition 2 (c_0-universality and RKHS embedding of measures) *Suppose X is an LCH space with the kernel, k being bounded and $k(\cdot,x) \in C_0(X), \forall x \in X$. Then k is c_0-universal if and only if the embedding,*

$$\mu \mapsto \int_X k(\cdot,x)\, d\mu(x),\ \mu \in M_b(X), \tag{14}$$

is injective.

Proof By definition, k is c_0-universal if $\mathcal{H}$ is dense in $C_0(X)$. We now invoke Theorem 1 to characterize the denseness of $\mathcal{H}$ in $C_0(X)$, which means we need to consider the dual $C_0'(X) := (C_0(X))'$ of $C_0(X)$. By the Riesz representation theorem (Folland, 1999, Theorem 7.17), $C_0'(X) = M_b(X)$ in the sense that there is a bijective linear isometry $\mu \mapsto T_\mu$ from $M_b(X)$ onto $C_0'(X)$, given by the natural mapping, $T_\mu(f) = \int_X f\, d\mu,\ f \in C_0(X)$. Therefore, by Theorem 1, $\mathcal{H}$ is dense in $C_0(X)$ if and only if $\mathcal{H}^\perp := \{\mu \in M_b(X) : \forall f \in \mathcal{H},\ \int_X f\, d\mu = 0\} = \{0\}$. From Lemma 7 (see Appendix B), we have $\mathcal{H}^\perp = \{\mu \in M_b(X) : \int_X k(\cdot,x)\, d\mu(x) = 0\}$ and therefore the result follows from Theorem 1. ■

Remark 3 *(a) When X is compact, $C_0(X)$ coincides with $C(X)$, and therefore the result in (14) matches with the one in (8), derived by Micchelli et al. (2006).*

(b) The characterization of cc-universality, shown in (11) can also be directly obtained as a simple application of Theorem 1, wherein the proof is similar to that of Proposition 2 except that we need to consider the dual of $C(X)$ endowed with the topology of compact convergence (a locally convex topological vector space) to characterize the denseness of $\mathcal{H}$ in $C(X)$. It is known (Hewitt, 1950) that $C'(X) = M_{bc}(X)$ in the sense that there is a bijective linear isometry $\mu \mapsto T_\mu$ from $M_{bc}(X)$ onto $C'(X)$, given by the natural mapping, $T_\mu(f) = \int_X f\, d\mu,\ f \in C(X)$. The rest of the proof is verbatim with $M_b(X)$ replaced by $M_{bc}(X)$.

(c) Comparing (14) and (2), it is clear that c_0-universal kernels are characteristic while the converse is not true, which matches with the result in Section 3.2.

4.2 Relation Between Universal Kernels and Integrally Strictly PD Kernels

In this section, we address the open question (E) through the following result which shows that c_0-kernels are integrally strictly pd if and only if they are c_0-universal.

Proposition 4 (c_0-universal and integrally strictly pd kernels) *Suppose the assumptions in Proposition 2 hold. Then, a c_0-kernel, k is c_0-universal if and only if it is integrally strictly pd, that is,*

$$\int\int_X k(x,y)\, d\mu(x)\, d\mu(y) > 0, \forall \mu \in M_b(X)\backslash\{0\}. \tag{15}$$

Proof ($\Leftarrow$) Suppose k is not c_0-universal. By Proposition 2, there exists $\mu \in M_b(X)\backslash\{0\}$ such that $\int_X k(\cdot,x)\, d\mu(x) = 0$, which implies $\| \int_X k(\cdot,x)\, d\mu(x)\|_{\mathcal{H}} = 0$. This means

$$0 = \left\langle \int_X k(\cdot,x)\, d\mu(x), \int_X k(\cdot,x)\, d\mu(x) \right\rangle_{\mathcal{H}} \overset{(e)}{=} \int\int_X k(x,y)\, d\mu(x)\, d\mu(y),$$

that is, k is not integrally strictly pd, where (e) follows from Lemma 7 (see Appendix B). Therefore, if (15) holds, then k is c_0-universal.

($\Rightarrow$) Suppose there exists $\mu \in M_b(X)\backslash\{0\}$ such that $\int\int_X k(x,y)\, d\mu(x)\, d\mu(y) = 0$, that is,

$$\left\| \int_X k(\cdot,x)\, d\mu(x) \right\|_{\mathcal{H}} = 0 \Rightarrow \int_X k(\cdot,x)\, d\mu(x) = 0.$$

Therefore, the embedding in (14) is not injective, which by Proposition 2 implies that k is not c_0-universal. Therefore, if k is c_0-universal, then k satisfies (15). ■

4.3 Radial Kernels on $\mathbb{R}^d$

In this section, we address the open questions (B), (C), (D) and (G) by showing that all the notions of universality and characteristic kernels are equivalent to strictly pd kernels.

Proposition 5 (All notions are equivalent for radial kernels on $\mathbb{R}^d$) *Suppose k is radial on $\mathbb{R}^d$. Then the following conditions are equivalent.*

(a) $\mathrm{supp}(\nu) \neq \{0\}$.

(b) k is integrally strictly pd.

(c) k is c_0-universal.

(d) k is cc-universal.

(e) k is strictly pd.

(f) k is characteristic.

Proof Note that $(b) \Leftrightarrow (c)$ follows from Proposition 4, $(c) \Rightarrow (d)$ from (11) and (13) and $(d) \Leftrightarrow (e)$ from Micchelli et al. (2006, Proposition 14) and Wendland (2005, Theorem 7.14). Theorem 7.14 in Wendland (2005) also ensures that $(e) \Rightarrow (a)$. Now, we show $(a) \Rightarrow (b)$. To do this, we first derive an intermediate result. Suppose $\hat{\mu}$ is the Fourier transform of μ defined as $\hat{\mu}(\omega) = \int_{\mathbb{R}^d} e^{\sqrt{-1}\omega^T x} d\mu(x)$, then for any ψ defined as in (5), we have

$$\begin{aligned}
\int\int_{\mathbb{R}^d} \psi(x-y)\, d\mu(x)\, d\mu(y) &= \int\int\int_{\mathbb{R}^d} e^{-\sqrt{-1}(x-y)^T\omega}\, d\Lambda(\omega)\, d\mu(x)\, d\mu(y) \\
&= \int\int_{\mathbb{R}^d} e^{-\sqrt{-1}x^T\omega}\, d\mu(x) \int_{\mathbb{R}^d} e^{\sqrt{-1}y^T\omega}\, d\mu(y)\, d\Lambda(\omega) \\
&= \int_{\mathbb{R}^d} \hat{\mu}(\omega)\overline{\hat{\mu}(\omega)}\, d\Lambda(\omega) \\
&= \int_{\mathbb{R}^d} |\hat{\mu}(\omega)|^2\, d\Lambda(\omega). \qquad (16)
\end{aligned}$$

Consider $\int\int_{\mathbb{R}^d} k(x,y)\, d\mu(x)\, d\mu(y)$ with k as in (6), given by

$$\begin{aligned}
B := \int\int_{\mathbb{R}^d} k(x,y)\, d\mu(x)\, d\mu(y) &= \int\int_{\mathbb{R}^d} \int_0^\infty e^{-t\|x-y\|_2^2}\, d\nu(t)\, d\mu(x)\, d\mu(y) \\
&\overset{(\star)}{=} \int_0^\infty \left[\int\int_{\mathbb{R}^d} e^{-t\|x-y\|_2^2}\, d\mu(x)\, d\mu(y) \right] d\nu(t) \\
&\overset{(\clubsuit)}{=} \int_0^\infty \frac{1}{(4\pi t)^{d/2}} \left[\int_{\mathbb{R}^d} |\hat{\mu}(\omega)|^2 e^{-\frac{\|\omega\|_2^2}{4t}}\, d\omega \right] d\nu(t) \\
&\overset{(\spadesuit)}{=} \int_{\mathbb{R}^d} |\hat{\mu}(\omega)|^2 \left[\int_0^\infty \frac{1}{(4\pi t)^{d/2}} e^{-\frac{\|\omega\|_2^2}{4t}}\, d\nu(t) \right] d\omega, \qquad (17)
\end{aligned}$$

where Fubini's theorem is invoked in $(\star)$ and $(\spadesuit)$, while we used (16) in $(\clubsuit)$, where we set $\psi(x) = e^{-t\|x\|_2^2}$ with $d\Lambda(\omega) = (4\pi t)^{-d/2} e^{-\|\omega\|_2^2/4t}\, d\omega$. Since $\mathrm{supp}(\nu) \neq \{0\}$, the inner integral in (17) is positive for every $\omega \in \mathbb{R}^d$ and so $B > 0$, which means k is integrally strictly pd.

We now prove that $(c) \Leftrightarrow (f)$. $(c) \Rightarrow (f)$ follows from Section 3.2. To prove the converse, we need to prove that if k is not c_0-universal, then it is not characteristic. If k is not c_0-universal, then we have $\text{supp}(\nu) = \{0\}$, which means the kernel is a constant function on $\mathbb{R}^d \times \mathbb{R}^d$ and therefore not characteristic. ■

4.4 Relation Between Characteristic and Conditionally Strictly PD Kernels

In this section we address the open question (F) which is about the relation of characteristic kernels to conditionally strictly pd kernels.

As shown in Section 3.3, although the relation between universal and conditionally strictly pd kernels straightforwardly follows from universal kernels being strictly pd, which in turn are conditionally strictly pd, such an implication is not possible in the case of characteristic kernels as they are not in general strictly pd (see Example 1). However, the following result establishes the relation between characteristic and conditionally strictly pd kernels.

Proposition 6 *If k is characteristic, then it is conditionally strictly pd.*

Proof Suppose k is not conditionally strictly pd. This means for some $n \geq 2$ and for mutually distinct $x_1, \ldots, x_n \in X$, there exists $\{\alpha_j\}_{j=1}^n \neq 0$ with $\sum_{j=1}^n \alpha_j = 0$ such that $\sum_{l,j=1}^n \alpha_l \alpha_j k(x_l, x_j) = 0$. Define $I := \{j : \alpha_j > 0\}$, $\mathbb{P} := \beta^{-1} \sum_{j \in I} \alpha_j \delta_{x_j}$ and $\mathbb{Q} := -\beta^{-1} \sum_{j \notin I} \alpha_j \delta_{x_j}$, where $\beta := \sum_{j \in I} \alpha_j$. It is easy to see that $\mathbb{P}$ and $\mathbb{Q}$ are distinct Borel probability measures on X. Then, we have

$$\left\| \int_X k(\cdot, x)\, d(\mathbb{P} - \mathbb{Q})(x) \right\|_{\mathcal{H}}^2 = \beta^{-2} \left\| \sum_{j=1}^n \alpha_j k(\cdot, x_j) \right\|_{\mathcal{H}} = \beta^{-2} \sum_{l,j=1}^n \alpha_l \alpha_j k(x_l, x_j) = 0.$$

So, there exist $\mathbb{P} \neq \mathbb{Q}$ such that $\int_X k(\cdot, x)\, d(\mathbb{P} - \mathbb{Q})(x) = 0$, that is, k is not characteristic. ■

The converse to Proposition 6 in general is however not true: we showed in Section 3.3 that strictly pd kernels are conditionally strictly pd but need not be characteristic and so conditionally strictly pd kernels need not have to be characteristic. In the following, we present a concrete example to show the same—a similar example is used to prove Theorem 4.62 in Steinwart and Christmann (2008), which shows that c_0-kernels that are strictly pd need not be c_0-universal.

Example 2 *Let $X = \mathbb{N} \cup \{0\}$. Define $k(0,0) = \sum_{n \in \mathbb{N}} b_n^2$, $k(m,n) = \delta_{mn}$ and $k(n,0) = b_n$ for $m, n \geq 1$, where $\{b_n\}_{n \geq 1} \subset (0,1)$ and $\sum_{n \in \mathbb{N}} b_n = 1$. Let $n \geq 2$ and $\alpha := (\alpha_0, \ldots, \alpha_n) \in \mathbb{R}^{n+1}$ be a vector with $\alpha \neq 0$ such that $\sum_{j=0}^n \alpha_j = 0$. Consider*

$$\begin{aligned} B := \sum_{l,j=0}^n \alpha_l \alpha_j k(l,j) &= \alpha_0^2 k(0,0) + 2 \sum_{j=1}^n \alpha_j \alpha_0 k(j,0) + \sum_{l,j=1}^n \alpha_l \alpha_j k(l,j) \\ &= \alpha_0^2 \sum_{j \in \mathbb{N}} b_j^2 + 2\alpha_0 \sum_{j=1}^n \alpha_j b_j + \sum_{j=1}^n \alpha_j^2 = \alpha_0^2 \sum_{j \in \mathbb{N}} b_j^2 + \sum_{j=1}^n \alpha_j (2\alpha_0 b_j + \alpha_j). \end{aligned}$$

If $\alpha_0 = 0$, then $B = \sum_{j=1}^n \alpha_j^2 > 0$ since we assumed $\alpha \neq 0$. Suppose $\alpha_0 \neq 0$. Then

$$B \geq \alpha_0^2 \sum_{j \in \mathbb{N}} b_j^2 + \sum_{j=1}^n \alpha_j^* (2\alpha_0 b_j + \alpha_j^*), \tag{18}$$

where

$$(\alpha_1^*,\ldots,\alpha_n^*) = \arg\min\left\{\sum_{j=1}^n \alpha_j(2\alpha_0 b_j+\alpha_j) : \sum_{j=1}^n \alpha_j = -\alpha_0\right\}. \tag{19}$$

Note that $(\alpha_1^*,\ldots,\alpha_n^*)$ *is unique as the objective in (19) is strictly convex, which is minimized over a convex set. To solve (19), let us consider the Lagrangian, given as*

$$L(\alpha_1,\ldots,\alpha_n,\lambda) = \sum_{j=1}^n \alpha_j(2\alpha_0 b_j+\alpha_j) - \lambda\Big(\sum_{j=1}^n \alpha_j+\alpha_0\Big),$$

where $\lambda \geq 0$. *Differentiating L w.r.t.* α_j *and setting it to zero yields* $\alpha_j^* = (\lambda - 2\alpha_0 b_j)/2$. *Since* $\sum_{j=1}^n \alpha_j^* = -\alpha_0$, *we have* $\lambda = \frac{2\alpha_0(a-1)}{n}$, *where* $a := \sum_{j=1}^n b_j$. *Substituting for* λ *in* α_j^*, *we have*

$$\alpha_j^* = \frac{\alpha_0(a-1-nb_j)}{n},\ j \in \mathbb{N}_n.$$

Substituting for α_j^* *in (18) gives*

$$B \geq \alpha_0^2 \sum_{j\in\mathbb{N}} b_j^2 + \frac{\alpha_0^2(a-1)^2}{n} - \alpha_0^2 \sum_{j=1}^n b_j^2 = \alpha_0^2 \sum_{j=n+1}^{\infty} b_j^2 + \frac{\alpha_0^2(\sum_{j=1}^n b_j - 1)^2}{n} > 0.$$

Consequently, we have $B > 0$ *in any case, and therefore k is conditionally strictly pd. In the following, we however show that k is not characteristic.*

Let $\mathbb{P} = \delta_0$ *and* $\mathbb{Q} = \sum_{j=1}^n b_j \delta_j$. *Clearly* $\mathbb{P} \neq \mathbb{Q}$. *Consider*

$$\begin{aligned}\Big\|\int_X k(\cdot,x)\,d(\mathbb{P}-\mathbb{Q})(x)\Big\|_{\mathcal{H}}^2 &= \Big\|k(\cdot,0) - \sum_{j\in\mathbb{N}} k(\cdot,j)b_j\Big\|_{\mathcal{H}}^2 \\ &= k(0,0) - 2\sum_{j\in\mathbb{N}} k(j,0)b_j + \sum_{l,j\in\mathbb{N}} k(l,j)b_l b_j \\ &= \sum_{j\in\mathbb{N}} b_j^2 - 2\sum_{j\in\mathbb{N}} b_j^2 + \sum_{j\in\mathbb{N}} b_j^2 = 0.\end{aligned}$$

This implies the embedding in (2) is not injective and therefore k is not characteristic.

When X is finite, then the converse to Proposition 6 holds, that is, conditionally strictly pd kernels are characteristic, which is shown as follows. Let $X = \mathbb{N}_n$. Suppose k is conditionally strictly pd, that is, for any $n \geq 2$, $(\alpha_1,\ldots,\alpha_n) \neq (0,\ldots,0)$ with $\sum_{j=1}^n \alpha_j = 0$, and all distinct $x_1,\ldots,x_n \in X$, we have $\sum_{l,j=1}^n \alpha_l\alpha_j k(x_l,x_j) > 0$. Let $I := \{j : \alpha_j > 0\}$. Define $\mathbb{P} := \beta^{-1}\sum_{j\in I}\alpha_j\delta_j$ and $\mathbb{Q} := -\beta^{-1}\sum_{j\notin I}\alpha_j\delta_j$, where $\beta := \sum_{j\in I}\alpha_j$ and $\mathbb{P} \neq \mathbb{Q}$. Then

$$\Big\|\int k(\cdot,x)\,d(\mathbb{P}-\mathbb{Q})(x)\Big\|_{\mathcal{H}}^2 = \beta^{-2}\sum_{l,j=1}^n \alpha_l\alpha_j k(l,j) > 0$$

and therefore k is characteristic.

5. Conclusions

In this work, we have presented a unified study to explain the relation between universal kernels, characteristic kernels and RKHS embedding of measures: while characteristic kernels are related to the injective RKHS embedding of Borel probability measures, the universal kernels are related to the injective RKHS embedding of finite signed Borel measures. We showed that for all practical purposes (e.g., Gaussian kernel, Laplacian kernel, etc.), the notions of characteristic and universal kernels are equivalent. In addition, we also explored their relation to various other notions of positive definite (pd) kernels: strictly pd, integrally strictly pd and conditionally strictly pd. As an example, we showed all these notions to be equivalent (except for conditionally strictly pd) in the case of radial kernels on $\mathbb{R}^d$. We would like to note that while this study assumes the kernel to be real-valued, all the results extend verbatim to the case of complex-valued kernels as well.

This unified study shows that certain families of kernels, for example, bounded continuous translation invariant kernels on $\mathbb{R}^d$ and radial kernels on $\mathbb{R}^d$, are interesting for practical use, since the disparate notions of universal and characteristic kernels seem to coincide for these families. On the other hand, it may not give a guide regarding which kernel should be used given a problem.

Acknowledgments

The authors thank anonymous reviewers for their constructive comments that greatly improved the manuscript and also for pointing out to Suquet (2009). B. K. S. and G. R. G. L. wish to acknowledge support from the Institute of Statistical Mathematics (ISM), Tokyo, the National Science Foundation (grant DMS-MSPA 0625409), the Fair Isaac Corporation and the University of California MICRO program. Most of this work was done when B. K. S. was affiliated with the University of California, San Diego, of which a part was carried out while B. K. S. was visiting ISM. K. F. was supported by JSPS KAKENHI 19500249 and 22300098.

Appendix A. Radial Kernels are Translation Invariant on $\mathbb{R}^d$

Let k be radial on $\mathbb{R}^d \times \mathbb{R}^d$. Define $k(x,y) = \psi(x-y) := \int_{[0,\infty)} e^{-t\|x-y\|_2^2}\, d\nu(t)$, $x, y \in \mathbb{R}^d$, where $\nu \in M_b^+([0,\infty))$. Since

$$e^{-t\|x-y\|_2^2} = \int_{\mathbb{R}^d} e^{-\sqrt{-1}(x-y)^T\omega}(4\pi t)^{-d/2} e^{-\|\omega\|_2^2/4t}\, d\omega,$$

we have $\psi(x) = \int_{\mathbb{R}^d} e^{-\sqrt{-1}x^T\omega}\phi(\omega)\, d\omega$, where

$$\phi(\omega) = \int_{[0,\infty)} (4\pi t)^{-d/2} e^{-\|\omega\|_2^2/4t}\, d\nu(t).$$

It is easy to check that $\phi(\omega) \geq 0$, $\forall\, \omega \in \mathbb{R}^d$ and $\phi \in L^1(\mathbb{R}^d)$. Therefore ψ satisfies (5), which means k is a bounded continuous translation invariant kernel on $\mathbb{R}^d$.

Appendix B. Supplementary Results

For completeness, we present the following supplementary result, which is a simple generalization of the technique used in the proof of Theorem 3 in Sriperumbudur et al. (2008).

Lemma 7 *Let k be a measurable and bounded kernel on a measurable space, X and let $\mathcal{H}$ be its associated RKHS. Then, for any $f \in \mathcal{H}$ and for any finite signed Borel measure, μ,*

$$\int_X f(x)\,d\mu(x) = \int_X \langle f, k(\cdot,x)\rangle_{\mathcal{H}}\,d\mu(x) = \left\langle f, \int_X k(\cdot,x)\,d\mu(x)\right\rangle_{\mathcal{H}}.$$

Proof Let $T_\mu : \mathcal{H} \to \mathbb{R}$ be a linear functional defined as $T_\mu[f] := \int_X f(x)\,d\mu(x)$. It is easy to show that

$$\|T_\mu\| := \sup_{f\in\mathcal{H}} \frac{|T_\mu[f]|}{\|f\|_{\mathcal{H}}} \leq \sqrt{\sup_{x\in X} k(x,x)}\|\mu\| < \infty.$$

Therefore, T_μ is a bounded linear functional on $\mathcal{H}$. By the Riesz representation theorem (Folland, 1999, Theorem 5.25), there exists a unique $\lambda_\mu \in \mathcal{H}$ such that $T_\mu[f] = \langle f, \lambda_\mu\rangle_{\mathcal{H}}$ for all $f \in \mathcal{H}$. Set $f = k(\cdot,u)$ for some $u \in X$, which implies $\lambda_\mu = \int_X k(\cdot,x)\,d\mu(x)$ and the result follows. ■

The following result in Theorem 8 characterizes strictly pd kernels on $\mathbb{T}$, which we quote from Menegatto (1995). Before we state the result, we introduce some notation. For natural numbers m and n and a set A of integers, $m+nA := \{j \in \mathbb{Z} \,|\, j = m+na,\, a \in A\}$. An increasing sequence $\{c_l\}$ of nonnegative integers is said to be *prime* if it is not contained in any set of the form $p_1\mathbb{N} \cup p_2\mathbb{N} \cup \cdots \cup p_n\mathbb{N}$, where $p_1, p_2, \ldots, p_n$ are prime numbers. Any infinite increasing sequence of prime numbers is a trivial example of a prime sequence. We write $\mathbb{N}_n^0 := \{0, 1, \ldots, n\}$.

Theorem 8 (Menegatto 1995) *Let ψ be a pd function on $\mathbb{T}$ of the form in (7). Let $\overline{N} := \{|n| : A_\psi(n) > 0,\, n \in \mathbb{Z}\} \subset \mathbb{N} \cup \{0\}$. Then ψ is strictly pd if $\overline{N}$ has a subset of the form $\cup_{l=0}^{\infty}(b_l + c_l\mathbb{N}_l^0)$, in which $\{b_l\} \cup \{c_l\} \subset \mathbb{N}$ and $\{c_l\}$ is a prime sequence.*

Appendix C. c_b-universality

As mentioned in Section 2, the definition of c_0-universality deals with $\mathcal{H}$ being dense in $C_0(X)$ w.r.t. the uniform norm, where X is an LCH space. Although the notion of c_0-universality addresses limitations associated with both c- and cc-universality, it only approximates a subset of $C(X)$, that is, it cannot deal with functions in $C(X)\backslash C_0(X)$. This limitation can be addressed by considering a larger class of functions to be approximated.

To this end, one can consider a notion of universality that is stronger than c_0-universality: a bounded continuous kernel, k is said to be c_b-universal if its corresponding RKHS, $\mathcal{H}$ is dense in $C_b(X)$, the space of bounded continuous functions on a topological space, X (note that $C_0(X) \subset C_b(X)$). This notion of c_b-universality may be more applicable in learning theory than c_0-universality as the target function, $f^\star$ can belong to $C_b(X)$ (which is a more natural assumption) instead of it being restrained to $C_0(X)$ (note that $C_0(X)$ only contains functions that vanish at infinity). Similar to Proposition 2, the following theorem provides a necessary and sufficient condition for k to be c_b-universal. Before we state the result, we need some definitions.

A *set function* is a function defined on a family of sets, and has values in $[-\infty, +\infty]$. A set function μ defined on a family τ of sets is said to be *finitely additive* if $\emptyset \in \tau$, $\mu(\emptyset) = 0$ and $\mu(\cup_{l=1}^n A_l) = \sum_{l=1}^n \mu(A_l)$, for every finite family $\{A_1, \ldots, A_n\}$ of disjoint subsets of τ such that $\cup_{l=1}^n A_l \in \tau$. A *field of subsets* of a set X is a non-empty family, Σ, of subsets of X such that $\emptyset \in \Sigma$, $X \in \Sigma$, and for all $A, B \in \Sigma$, we have $A \cup B \in \Sigma$ and $B\backslash A \in \Sigma$. An additive set function μ defined on a field Σ of subsets of a topological space X is said to be *regular* if for each $A \in \Sigma$ and $\varepsilon > 0$, there exists $B \in \Sigma$ whose closure is contained in A and there exists $C \in \Sigma$ whose interior contains A such that $|\mu(D)| < \varepsilon$ for every $D \in \Sigma$ with $D := C\backslash B$.

Proposition 9 (c_b-universality and RKHS embedding of set functions) *Suppose X is a normal topological space and $M_{rba}(X)$ is the space of all finitely additive, regular, bounded set functions defined on the field generated by the closed sets of X. Then, a bounded continuous kernel, k is c_b-universal if and only if the embedding,*

$$\mu \mapsto \int_X k(\cdot,x)\, d\mu, \mu \in M_{rba}(X), \tag{20}$$

is injective.

Proof The proof is very similar to that of Proposition 2, wherein we identify $(C_b(X))' \cong M_{rba}(X)$ such that $T \in (C_b(X))'$ and $\mu \in M_{rba}(X)$ satisfy $T(f) = \int_X f\, d\mu$, $f \in C_b(X)$ (Dunford and Schwartz, 1958, p. 262). Here, $\cong$ represents the isometric isomorphism. The rest of the proof is verbatim with $M_b(X)$ replaced by $M_{rba}(X)$. ■

Note that $M_{rba}(X)$ does not contain any measure—though a set function in $M_{rba}(X)$ can be extended to a measure—as measures are countably additive and defined on a σ-field. Since μ in Proposition 9 is not a measure but a finitely additive set function defined on a field, it is not clear how to deal with the integral in (20). Due to the technicalities involved in dealing with set functions, the analysis of c_b-universality and its relation to other notions considered in Section 3 is not clear, although it is an interesting problem to be resolved because of its applicability in learning theory.

References

N. Aronszajn. Theory of reproducing kernels. *Trans. Amer. Math. Soc.*, 68:337–404, 1950.

C. Berg, J. P. R. Christensen, and P. Ressel. *Harmonic Analysis on Semigroups*. Spring Verlag, New York, 1984.

A. Caponnetto, M. Pontil, C. Micchelli, and Y. Ying. Universal multi-task kernels. *Journal of Machine Learning Research*, 9:1615–1646, 2008.

C. Carmeli, E. De Vito, A. Toigo, and V. Umanità. Vector valued reproducing kernel Hilbert spaces and universality. *Analysis and Applications*, 8:19–61, 2010.

W. Dahmen and C. A. Micchelli. Some remarks on ridge functions. *Approx. Theory Appl.*, 3: 139–143, 1987.

R. M. Dudley. *Real Analysis and Probability*. Cambridge University Press, Cambridge, UK, 2002.

N. Dunford and J. T. Schwartz. *Linear Operators. I: General Theory*. Wiley-Interscience, New York, 1958.

T. Evgeniou, M. Pontil, and T. Poggio. Regularization networks and support vector machines. *Advances in Computational Mathematics*, 13(1):1–50, 2000.

G. B. Folland. *Real Analysis: Modern Techniques and Their Applications*. Wiley-Interscience, New York, 1999.

K. Fukumizu, F. Bach, and M. Jordan. Dimensionality reduction for supervised learning with reproducing kernel Hilbert spaces. *Journal of Machine Learning Research*, 5:73–99, 2004.

K. Fukumizu, A. Gretton, X. Sun, and B. Schölkopf. Kernel measures of conditional dependence. In J.C. Platt, D. Koller, Y. Singer, and S. Roweis, editors, *Advances in Neural Information Processing Systems 20*, pages 489–496, Cambridge, MA, 2008. MIT Press.

K. Fukumizu, B. K. Sriperumbudur, A. Gretton, and B. Schölkopf. Characteristic kernels on groups and semigroups. In D. Koller, D. Schuurmans, Y. Bengio, and L. Bottou, editors, *Advances in Neural Information Processing Systems 21*, pages 473–480, 2009.

A. Gretton, K. M. Borgwardt, M. Rasch, B. Schölkopf, and A. Smola. A kernel method for the two sample problem. In B. Schölkopf, J. Platt, and T. Hoffman, editors, *Advances in Neural Information Processing Systems 19*, pages 513–520. MIT Press, 2007.

A. Gretton, K. Fukumizu, C.-H. Teo, L. Song, B. Schölkopf, and A. Smola. A kernel statistical test of independence. In *Advances in Neural Information Processing Systems 20*, pages 585–592. MIT Press, 2008.

E. Hewitt. Linear functionals on spaces of continuous functions. *Fundamenta Mathematicae*, 37: 161–189, 1950.

G. S. Kimeldorf and G. Wahba. A correspondence between bayesian estimation on stochastic processes and smoothing by splines. *Annals of Mathematical Statistics*, 41(2):495–502, 1970.

V. A. Menegatto. Strictly positive definite kernels on the circle. *Rocky Mountain Journal of Mathematics*, 25(3):1149–1163, 1995.

C. A. Micchelli, Y. Xu, and H. Zhang. Universal kernels. *Journal of Machine Learning Research*, 7:2651–2667, 2006.

A. Pinkus. Strictly positive definite functions on a real inner product space. *Adv. Comput. Math.*, 20:263–271, 2004.

W. Rudin. *Functional Analysis*. McGraw-Hill, USA, 1991.

B. Schölkopf and A. J. Smola. *Learning with Kernels*. MIT Press, Cambridge, MA, 2002.

B. Schölkopf, R. Herbrich, and A. J. Smola. A generalized representer theorem. In *Proc. of the 14th Annual Conference on Learning Theory*, pages 416–426, 2001.

J. Shawe-Taylor and N. Cristianini. *Kernel Methods for Pattern Analysis*. Cambridge University Press, UK, 2004.

B. K. Sriperumbudur, A. Gretton, K. Fukumizu, G. R. G. Lanckriet, and B. Schölkopf. Injective Hilbert space embeddings of probability measures. In R. Servedio and T. Zhang, editors, *Proc. of the 21st Annual Conference on Learning Theory*, pages 111–122, 2008.

B. K. Sriperumbudur, K. Fukumizu, A. Gretton, G. R. G. Lanckriet, and B. Schölkopf. Kernel choice and classifiability for RKHS embeddings of probability distributions. In Y. Bengio, D. Schuurmans, J. Lafferty, C. K. I. Williams, and A. Culotta, editors, *Advances in Neural Information Processing Systems 22*, pages 1750–1758. MIT Press, 2009.

B. K. Sriperumbudur, K. Fukumizu, and G. R. G. Lanckriet. On the relation between universality, characteristic kernels and RKHS embedding of measures. In *JMLR Workshop and Conference Proceedings*, volume 9, pages 781–788. AISTATS, 2010a.

B. K. Sriperumbudur, A. Gretton, K. Fukumizu, B. Schölkopf, and G. R. G. Lanckriet. Hilbert space embeddings and metrics on probability measures. *Journal of Machine Learning Research*, 11:1517–1561, 2010b.

I. Steinwart. On the influence of the kernel on the consistency of support vector machines. *Journal of Machine Learning Research*, 2:67–93, 2001.

I. Steinwart and A. Christmann. *Support Vector Machines*. Springer, 2008.

J. Stewart. Positive definite functions and generalizations, an historical survey. *Rocky Mountain Journal of Mathematics*, 6(3):409–433, 1976.

Ch. Suquet. Reproducing kernel Hilbert spaces and random measures. In H. G. W. Begehr and F. Nicolosi, editors, *Proc. of the 5th International ISAAC Congress, Catania, Italy, 25-30 July 2005*, pages 143–152. World Scientific, 2009.

H. Wendland. *Scattered Data Approximation*. Cambridge University Press, Cambridge, UK, 2005.

Journal of Machine Learning Research 12 (2011) 2411-2414 Submitted 8/09; Revised 7/10; Published 7/11

MULAN: A Java Library for Multi-Label Learning

Grigorios Tsoumakas GREG@CSD.AUTH.GR
Eleftherios Spyromitros-Xioufis ESPYROMI@CSD.AUTH.GR
Jozef Vilcek JOJOVILCO@GMAIL.COM
Ioannis Vlahavas VLAHAVAS@CSD.AUTH.GR
Department of Informatics
Aristotle University of Thessaloniki
Thessaloniki 54124, Greece

Editor: Cheng Soon Ong

Abstract

MULAN is a Java library for learning from multi-label data. It offers a variety of classification, ranking, thresholding and dimensionality reduction algorithms, as well as algorithms for learning from hierarchically structured labels. In addition, it contains an evaluation framework that calculates a rich variety of performance measures.

Keywords: multi-label data, classification, ranking, thresholding, dimensionality reduction, hierarchical classification, evaluation

1. Multi-Label Learning

A multi-label data set consists of training examples that are associated with a subset of a finite set of labels. Nowadays, multi-label data are becoming ubiquitous. They arise in an increasing number and diversity of applications, such as semantic annotation of images and video, web page categorization, direct marketing, functional genomics and music categorization into genres and emotions.

There exist two major multi-label learning tasks (Tsoumakas et al., 2010): *multi-label classification* and *label ranking*. The former is concerned with learning a model that outputs a bipartition of the set of labels into relevant and irrelevant with respect to a query instance. The latter is concerned with learning a model that outputs a ranking of the labels according to their relevance to a query instance. Some algorithms learn models that serve both tasks. Several algorithms learn models that primarily output a vector of numerical scores, one for each label. This vector is then converted to a ranking after solving ties, or to a bipartition, after *thresholding* (Ioannou et al., 2010).

Multi-label learning methods addressing these tasks can be grouped into two categories (Tsoumakas et al., 2010): *problem transformation* and *algorithm adaptation*. The first group of methods are algorithm independent. They transform the learning task into one or more single-label classification tasks, for which a large body of learning algorithms exists. The second group of methods extend specific learning algorithms in order to handle multi-label data directly. There exist extensions of decision tree learners, nearest neighbor classifiers, neural networks, ensemble methods, support vector machines, kernel methods, genetic algorithms and others.

Multi-label learning stretches across several other tasks. When labels are structured as a tree-shaped hierarchy or a directed acyclic graph, then we have the interesting task of *hierarchical multi-label learning*. *Dimensionality reduction* is another important task for multi-label data, as it is for

any kind of data. When bags of instances are used to represent a training object, then *multi-instance multi-label* learning algorithms are required. There also exist *semi-supervised learning* and *active learning* algorithms for multi-label data.

2. The MULAN Library

The main goal of MULAN is to bring the benefits of machine learning open source software (MLOSS) (Sonnenburg et al., 2007) to people working with multi-label data. The availability of MLOSS is especially important in emerging areas like multi-label learning, because it removes the burden of implementing related work and speeds up the scientific progress. In multi-label learning, an extra burden is implementing appropriate evaluation measures, since these are different compared to traditional supervised learning tasks. Evaluating multi-label algorithms with a variety of measures, is considered important by the community, due to the different types of output (bipartition, ranking) and diverse applications.

Towards this goal, MULAN offers a plethora of state-of-the-art algorithms for multi-label classification and label ranking and an evaluation framework that computes a large variety of multi-label evaluation measures through hold-out evaluation and cross-validation. In addition, the library offers a number of thresholding strategies that produce bipartitions from score vectors, simple baseline methods for multi-label dimensionality reduction and support for hierarchical multi-label classification, including an implemented algorithm.

MULAN is a library. As such, it offers only programmatic API to the library users. There is no graphical user interface (GUI) available. The possibility to use the library via command line, is also currently not supported. Another drawback of MULAN is that it runs everything in main memory so there exist limitations with very large data sets.

MULAN is written in Java and is built on top of Weka (Witten and Frank, 2005). This choice was made in order to take advantage of the vast resources of Weka on supervised learning algorithms, since many state-of-the-art multi-label learning algorithms are based on problem transformation. The fact that several machine learning researchers and practitioners are familiar with Weka was another reason for this choice. However, many aspects of the library are independent of Weka and there are interfaces for most of the core classes.

MULAN is an advocate of open science in general. One of the unique features of the library is a recently introduced experiments package, whose goal is to host code that reproduces experimental results reported on published papers on multi-label learning.

To the best of our knowledge, most of the general learning platforms, like Weka, don't support multi-label data. There are currently only a number of implementations of specific multi-label learning algorithms, but not a general library like MULAN.

3. Using MULAN

This section presents an example of how to setup an experiment for empirically evaluating two multi-label algorithms on a multi-label data set using cross-validation. We create a new Java class for this experiment, which we call `MulanExp1.java`.

The first thing to do is load the multi-label data set that will be used for the empirical evaluation. MULAN requires two text files for the specification of a data set. The first one is in the ARFF format of Weka. The labels should be specified as nominal attributes with values "0" and "1" indicating

absence and presence of the label respectively. The second file is in XML format. It specifies the labels and any hierarchical relationships among them. Hierarchies of labels can be expressed in the XML file by nesting the `label` tag.

In our example, the two filenames are given to the experiment class through command-line parameters.

```
String arffFile = Utils.getOption("arff", args);
String xmlFile = Utils.getOption("xml", args);
```

Loading the data can then be done using the following code.

```
MultiLabelInstances data = new MultiLabelInstances(arffFile, xmlFile);
```

The next step is to create an instance from each of the two learners that we want to evaluate. We will create an instance of the RA*k*EL and ML*k*NN algorithms. RA*k*EL is actually a meta algorithm and can accept any multi-label learner as a parameter, but is typically used in conjunction with the Label Powerset (LP) algorithm. In turn LP is a transformation-based algorithm and it accepts a single-label classifier as a parameter. We will use Weka's J48 algorithm for this purpose. ML*k*NN is an algorithm adaptation method that is based on *k*NN.

```
RAkEL learner1 = new RAkEL(new LabelPowerset(new J48()));
MLkNN learner2 = new MLkNN();
```

We then declare an `Evaluator` object that handles empirical evaluations and an object of the `MultipleEvaluation` class that stores cross-validation results.

```
Evaluator eval = new Evaluator();
MultipleEvaluation results;
```

To actually perform the evaluations we use the `crossValidate` method of the `Evaluator` class. This returns a `MultipleEvaluation` object, which we can print to see the results in terms of all applicable evaluation measures available in MULAN.

```
int numFolds = 10;
results = eval.crossValidate(learner1, data, numFolds);
System.out.println(results);
results = eval.crossValidate(learner2, data, numFolds);
System.out.println(results);
```

For running the experiment, we can use the emotions data (emotions.xml and emotions.arff) that are available together with the MULAN distribution. Other open access multi-label data sets can be found at `http://mulan.sourceforge.net/datasets.html`. Assuming the experiment's source file is in the same directory with emotions.arff, emotions.xml, weka.jar and mulan.jar from the distribution package, then to run this experiment we type the following commands (under Linux use : instead of ; as path separator).

```
javac -cp mulan.jar;weka.jar MulanExp1.java
java -cp mulan.jar;weka.jar;. MulanExp1 -arff emotions.arff -xml emotions.xml
```

The `mulan.examples` package includes additional examples of usage of the MULAN API, such as how to do hold-out and cross-validation learning experiments, how to store/load learned models, perform dimensionality reduction, estimate data set statistics and obtain predictions on test sets with unknown label values.

4. Documentation, Requirements and Availability

MULAN's online documentation[1] contains user oriented sections, such as *getting started with* MULAN and *the data set format of* MULAN, as well as developer-oriented sections, such as *extending* MULAN, *API reference* and *running tests*. There is also a mailing list for requesting support on using or extending MULAN.

MULAN is available under the GNU GPL licence. The current version of the library[2] is 1.3.0. It requires Java version 1.6, Weka version 3.7.3 and JUnit version 4.5 (only for running tests).

Acknowledgments

We would like to thank several people that have contributed pieces of code to the library. First and most importantly Robert Friberg for his help on the first steps towards MULAN. Then, the following people in alphabetical order: S. Bakirtzoglou, W. Cheng, M. Ioannou, I. Katakis, S.-H. Park, E. Rairat, G. Sakkas, G. Saridis, K. Sechidis, E. Stachtiari and G. Traianos.

References

Marios Ioannou, George Sakkas, Grigorios Tsoumakas, and Ioannis Vlahavas. Obtaining bipartitions from score vectors for multi-label classification. *IEEE International Conference on Tools with Artificial Intelligence*, 1:409–416, 2010. ISSN 1082-3409.

Sören Sonnenburg, Mikio L. Braun, Cheng Soon Ong, Samy Bengio, Leon Bottou, Geoffrey Holmes, Yann LeCun, Klaus-Robert Müller, Fernando Pereira, Carl Edward Rasmussen, Gunnar Rätsch, Bernhard Schölkopf, Alexander Smola, Pascal Vincent, Jason Weston, and Robert Williamson. The need for open source software in machine learning. *JMLR*, 8:2443–2466, 2007.

Grigorios Tsoumakas, Ioannis Katakis, and Ioannis Vlahavas. Mining multi-label data. In Oded Maimon and Lior Rokach, editors, *Data Mining and Knowledge Discovery Handbook*, chapter 34, pages 667–685. Springer, 2nd edition, 2010.

Ian H. Witten and Eibe Frank. *Data Mining: Practical Machine Learning Tools and Techniques*. Morgan Kaufmann, 2005.

1. Available at `http://mulan.sourceforge.net/documentation.html`.
2. Available for download at `http://sourceforge.net/projects/mulan/`.

Journal of Machine Learning Research 12 (2011) 2415-2435 Submitted 8/10; Revised 2/11; Published 7/11

Union Support Recovery in Multi-task Learning

Mladen Kolar MLADENK@CS.CMU.EDU
John Lafferty LAFFERTY@CS.CMU.EDU
Larry Wasserman LARRY@STAT.CMU.EDU
School of Computer Science
Carnegie Mellon University
5000 Forbes Avenue
Pittsburgh, PA 15213, USA

Editor: Hui Zou

Abstract

We sharply characterize the performance of different penalization schemes for the problem of selecting the relevant variables in the multi-task setting. Previous work focuses on the regression problem where conditions on the design matrix complicate the analysis. A clearer and simpler picture emerges by studying the Normal means model. This model, often used in the field of statistics, is a simplified model that provides a laboratory for studying complex procedures.

Keywords: high-dimensional inference, multi-task learning, sparsity, normal means, minimax estimation

1. Introduction

We consider the problem of estimating a sparse signal in the presence of noise. It has been empirically observed, on various data sets ranging from cognitive neuroscience (Liu et al., 2009) to genome-wide association mapping studies (Kim et al., 2009), that considering related estimation tasks jointly can improve estimation performance. Because of this, joint estimation from related tasks or *multi-task learning* has received much attention in the machine learning and statistics community (see for example Turlach et al., 2005; Zou and Yuan, 2008; Zhang, 2006; Negahban and Wainwright, 2009; Obozinski et al., 2011; Lounici et al., 2009; Liu et al., 2009; Lounici et al., 2010; Argyriou et al., 2008; Kim et al., 2009, and references therein). However, the theory behind multi-task learning is not yet settled.

An example of multi-task learning is the problem of estimating the coefficients of several multiple regressions

$$\mathbf{y}_j = \mathbf{X}_j \boldsymbol{\beta}_j + \boldsymbol{\epsilon}_j, \qquad j \in [k] \tag{1}$$

where $\mathbf{X}_j \in \mathbb{R}^{n \times p}$ is the design matrix, $\mathbf{y}_j \in \mathbb{R}^n$ is the vector of observations, $\boldsymbol{\epsilon}_j \in \mathbb{R}^n$ is the noise vector and $\boldsymbol{\beta}_j \in \mathbb{R}^p$ is the unknown vector of regression coefficients for the j-th task, with $[k] = \{1, \ldots, k\}$.

When the number of variables p is much larger than the sample size n, it is commonly assumed that the regression coefficients are jointly sparse, that is, there exists a small subset $S \subset [p]$ of the regression coefficients, with $s := |S| \ll n$, that are non-zero for all or most of the tasks.

The model in (1) under the joint sparsity assumption was analyzed in, for example, Obozinski et al. (2011), Lounici et al. (2009), Negahban and Wainwright (2009), Lounici et al. (2010) and

Kolar and Xing (2010). Obozinski et al. (2011) propose to minimize the penalized least squares objective with a mixed $(2,1)$-norm on the coefficients as the penalty term. The authors focus on consistent estimation of the support set S, albeit under the assumption that the number of tasks k is fixed. Negahban and Wainwright (2009) use the mixed $(\infty,1)$-norm to penalize the coefficients and focus on the exact recovery of the non-zero pattern of the regression coefficients, rather than the support set S. For a rather limited case of $k=2$, the authors show that when the regression do not share a common support, it may be harmful to consider the regression problems jointly using the mixed $(\infty,1)$-norm penalty. Kolar and Xing (2010) address the feature selection properties of simultaneous greedy forward selection. However, it is not clear what the benefits are compared to the ordinary forward selection done on each task separately. In Lounici et al. (2009) and Lounici et al. (2010), the focus is shifted from the consistent selection to benefits of the joint estimation for the prediction accuracy and consistent estimation. The number of tasks k is allowed to increase with the sample size. However, it is assumed that all tasks share the same features; that is, a relevant coefficient is non-zero for all tasks.

Despite these previous investigations, the theory is far from settled. A simple clear picture of when sharing between tasks actually improves performance has not emerged. In particular, to the best of our knowledge, there has been no previous work that sharply characterizes the performance of different penalization schemes on the problem of selecting the relevant variables in the multi-task setting.

In this paper we study multi-task learning in the context of the *many Normal means model*. This is a simplified model that is often useful for studying the theoretical properties of statistical procedures. The use of the many Normal means model is fairly common in statistics but appears to be less common in machine learning. Our results provide a sharp characterization of the sparsity patterns under which the Lasso procedure performs better than the group Lasso. Similarly, our results characterize how the group Lasso (with the mixed $(2,1)$ norm) can perform better when each non-zero row is dense.

1.1 The Normal Means Model

The simplest Normal means model has the form

$$Y_i = \mu_i + \sigma\varepsilon_i, \quad i = 1,\ldots,p \tag{2}$$

where $\mu_1,\ldots,\mu_p$ are unknown parameters and $\varepsilon_1,\ldots,\varepsilon_p$ are independent, identically distributed Normal random variables with mean 0 and variance 1. There are a variety of results (Brown and Low, 1996; Nussbaum, 1996) showing that many learning problems can be converted into a Normal means problem. This implies that results obtained in the Normal means setting can be transferred to many other settings. As a simple example, consider the nonparametric regression model $Z_i = m(i/n) + \delta_i$ where m is a smooth function on $[0,1]$ and $\delta_i \sim N(0,1)$. Let $\phi_1,\phi_2,\ldots,$ be an orthonormal basis on [0,1] and write $m(x) = \sum_{j=1}^{\infty} \mu_j \phi_j(x)$ where $\mu_j = \int_0^1 m(x)\phi_j(x)dx$. To estimate the regression function m we need only estimate $\mu_1,\mu_2,\ldots,$. Let $Y_j = n^{-1}\sum_{i=1}^n Z_i \phi_j(i/n)$. Then $Y_j \approx N(\mu_j,\sigma^2)$ where $\sigma^2 = 1/n$. This has the form of (2) with $\sigma = 1/\sqrt{n}$. Hence this regression problem can be converted into a Normal means model.

However, the most important aspect of the Normal means model is that it allows a clean setting for studying complex problems. In this paper, we consider the following Normal means model. Let

$$Y_{ij} \sim \begin{cases} (1-\varepsilon)\mathcal{N}(0,\sigma^2)+\varepsilon\mathcal{N}(\mu_{ij},\sigma^2) & j\in[k], \quad i\in S \\ N(0,\sigma^2) & j\in[k], \quad i\in S^c \end{cases} \tag{3}$$

where $(\mu_{ij})_{i,j}$ are unknown real numbers, $\sigma=\sigma_0/\sqrt{n}$ is the variance with $\sigma_0>0$ known, $(Y_{ij})_{i,j}$ are random observations, $\varepsilon\in[0,1]$ is the parameter that controls the sparsity of features across tasks and $S\subset[p]$ is the set of relevant features. Let $s=|S|$ denote the number of relevant features. Denote the matrix $M\in\mathbb{R}^{p\times k}$ of means

	Tasks			
	1	2	…	k
1	μ_{11}	μ_{12}	…	μ_{1k}
2	μ_{21}	μ_{22}	…	μ_{2k}
⋮	⋮	⋮	⋱	⋮
p	μ_{p1}	μ_{p2}	…	μ_{pk}

and let $\boldsymbol{\theta}_i=(\mu_{ij})_{j\in[k]}$ denote the i-th row of the matrix M. The set $S^c=[p]\backslash S$ indexes the zero rows of the matrix M and the associated observations are distributed according to the Normal distribution with zero mean and variance σ^2. The rows indexed by S are non-zero and the corresponding observation are coming from a mixture of two Normal distributions. The parameter ε determines the proportion of observations coming from a Normal distribution with non-zero mean. The reader should regard each column as one vector of parameters that we want to estimate. The question is whether sharing across columns improves the estimation performance.

It is known from the work on the Lasso that in regression problems, the design matrix needs to satisfy certain conditions in order for the Lasso to correctly identify the support S (see van de Geer and Bühlmann, 2009, for an extensive discussion on the different conditions). These regularity conditions are essentially unavoidable. However, the Normal means model (3) allows us to analyze the estimation procedure in (4) and focus on the scaling of the important parameters $(n,k,p,s,\varepsilon,\mu_{\min})$ for the success of the support recovery. Using the model (3) and the estimation procedure in (4), we are able to identify regimes in which estimating the support is more efficient using the ordinary Lasso than with the multi-task Lasso and vice versa. Our results suggest that the multi-task Lasso does not outperform the ordinary Lasso when the features are not considerably shared across tasks; thus, practitioners should be careful when applying the multi-task Lasso without knowledge of the task structure.

An alternative representation of the model is

$$Y_{ij}=\begin{cases} \mathcal{N}(\xi_{ij}\mu_{ij},\sigma^2) & j\in[k], \quad i\in S \\ N(0,\sigma^2) & j\in[k], \quad i\in S^c \end{cases}$$

where ξ_{ij} is a Bernoulli random variable with success probability ε. Throughout the paper, we will set $\varepsilon=k^{-\beta}$ for some parameter $\beta\in[0,1)$; $\beta<1/2$ corresponds to dense rows and $\beta>1/2$ corresponds to sparse rows. Let $\mu_{\min}$ denote the following quantity $\mu_{\min}=\min|\mu_{ij}|$.

Under the model (3), we analyze penalized least squares procedures of the form

$$\widehat{\boldsymbol{\mu}}=\underset{\boldsymbol{\mu}\in\mathbb{R}^{p\times k}}{\operatorname{argmin}}\ \frac{1}{2}||\mathbf{Y}-\boldsymbol{\mu}||_F^2+\operatorname{pen}(\boldsymbol{\mu}) \tag{4}$$

where $||A||_F = \sum_{jk} A_{jk}^2$ is the Frobenious norm, $\text{pen}(\cdot)$ is a penalty function and $\boldsymbol{\mu}$ is a $p \times k$ matrix of means. We consider the following penalties:

1. the ℓ_1 penalty
$$\text{pen}(\boldsymbol{\mu}) = \lambda \sum_{i \in [p]} \sum_{j \in [k]} |\mu_{ij}|,$$
which corresponds to the Lasso procedure applied on each task independently, and denote the resulting estimate as $\widehat{\boldsymbol{\mu}}^{\ell_1}$

2. the mixed $(2,1)$-norm penalty
$$\text{pen}(\boldsymbol{\mu}) = \lambda \sum_{i \in [p]} ||\boldsymbol{\theta}_i||_2,$$
which corresponds to the multi-task Lasso formulation in Obozinski et al. (2011) and Lounici et al. (2009), and denote the resulting estimate as $\widehat{\boldsymbol{\mu}}^{\ell_1/\ell_2}$

3. the mixed $(\infty,1)$-norm penalty
$$\text{pen}(\boldsymbol{\mu}) = \lambda \sum_{i \in [p]} ||\boldsymbol{\theta}_i||_\infty,$$
which correspond to the multi-task Lasso formulation in Negahban and Wainwright (2009), and denote the resulting estimate as $\widehat{\boldsymbol{\mu}}^{\ell_1/\ell_\infty}$.

For any solution $\widehat{\boldsymbol{\mu}}$ of (4), let $S(\widehat{\boldsymbol{\mu}})$ denote the set of estimated non-zero rows
$$S(\widehat{\boldsymbol{\mu}}) = \{i \in [p] \ : \ ||\widehat{\boldsymbol{\theta}}_i||_2 \neq 0\}.$$
We establish sufficient conditions under which $\mathbb{P}[S(\widehat{\boldsymbol{\mu}}) \neq S] \leq \alpha$ for different methods. These results are complemented with necessary conditions for the recovery of the support set S.

In this paper, we focus our attention on the three penalties outlined above. There is a large literature on the penalized least squares estimation using concave penalties as introduced in Fan and Li (2001). These penalization methods have better theoretical properties in the presence of the design matrix, especially when the design matrix is far from satisfying the irrepresentable condition (Zhao and Yu, 2006). In the Normal means model, due to the lack of the design matrix, there is no advantage to concave penalties in terms of variable selection.

1.2 Overview of the Main Results

The main contributions of the paper can be summarized as follows.

1. We establish a lower bound on the parameter $\mu_{\min}$ as a function of the parameters (n,k,p,s,β). Our result can be interpreted as follows: for any estimation procedure there exists a model given by (3) with non-zero elements equal to $\mu_{\min}$ such that the estimation procedure will make an error when identifying the set S with probability bounded away from zero.

2. We establish the sufficient conditions on the signal strength $\mu_{\min}$ for the Lasso and both variants of the group Lasso under which these procedures can correctly identify the set of non-zero rows S.

By comparing the lower bounds with the sufficient conditions, we are able to identify regimes in which each procedure is optimal for the problem of identifying the set of non-zero rows S. Furthermore, we point out that the usage of the popular group Lasso with the mixed $(\infty,1)$ norm can be disastrous when features are not perfectly shared among tasks. This is further demonstrated through an empirical study.

1.3 Organization of the Paper

The paper is organized as follows. We start by analyzing the lower bound for any procedure for the problem of identifying the set of non-zero rows in Section 2. In Section 3 we provide sufficient conditions on the signal strength $\mu_{\min}$ for the Lasso and the group Lasso to be able to detect the set of non-zero rows S. In the following section, we propose an improved approach to the problem of estimating the set S. Results of a small empirical study are reported in Section 4. We close the paper by a discussion of our findings.

2. Lower Bound on the Support Recovery

In this section, we derive a lower bound for the problem of identifying the correct variables. In particular, we derive conditions on $(n,k,p,s,\varepsilon,\mu_{\min})$ under which any method is going to make an error when estimating the correct variables. Intuitively, if $\mu_{\min}$ is very small, a non-zero row may be hard to distinguish from a zero row. Similarly, if ε is very small, many elements in a row will be zero and, again, as a result it may be difficult to identify a non-zero row. Before, we give the main result of the section, we introduce the class of models that are going to be considered.

Let

$$\mathcal{F}[\mu] := \{\boldsymbol{\theta} \in \mathbb{R}^k \,:\, \min_j |\theta_j| \geq \mu\}$$

denote the set of feasible non-zero rows. For each $j \in \{0,1,\ldots,k\}$, let $\mathcal{M}(j,k)$ be the class of all the subsets of $\{1,\ldots,k\}$ of cardinality j. Let

$$\mathbb{M}[\mu,s] = \bigcup_{\omega \in \mathcal{M}(s,p)} \Big\{(\boldsymbol{\theta}_1,\ldots,\boldsymbol{\theta}_p)' \in \mathbb{R}^{p\times k} \,:\, \boldsymbol{\theta}_i \in \mathcal{F}[\mu] \text{ if } i \in \omega,\ \boldsymbol{\theta}_i = \mathbf{0} \text{ if } i \notin \omega\Big\} \tag{5}$$

be the class of all feasible matrix means. For a matrix $M \in \mathbb{M}[\mu,s]$, let $\mathbb{P}_M$ denote the joint law of $\{Y_{ij}\}_{i\in[p],j\in[k]}$. Since $\mathbb{P}_M$ is a product measure, we can write $\mathbb{P}_M = \otimes_{i\in[p]}\mathbb{P}_{\boldsymbol{\theta}_i}$. For a non-zero row $\boldsymbol{\theta}_i$, we set

$$\mathbb{P}_{\boldsymbol{\theta}_i}(A) = \int \mathcal{N}(A;\widehat{\boldsymbol{\theta}},\sigma^2\mathbf{I}_k)d\nu(\widehat{\boldsymbol{\theta}}), \qquad A \in \mathcal{B}(\mathbb{R}^k),$$

where ν is the distribution of the random variable $\sum_{j\in[k]}\mu_{ij}\xi_j e_j$ with $\xi_j \sim \text{Bernoulli}(k^{-\beta})$ and $\{e_j\}_{j\in[k]}$ denoting the canonical basis of $\mathbb{R}^k$. For a zero row $\boldsymbol{\theta}_i = \mathbf{0}$, we set

$$\mathbb{P}_{\mathbf{0}}(A) = \mathcal{N}(A;\mathbf{0},\sigma^2\mathbf{I}_k), \qquad A \in \mathcal{B}(\mathbb{R}^k).$$

With this notation, we have the following result.

Theorem 1 *Let*

$$\mu^2_{\min} = \mu^2_{\min}(n,k,p,s,\varepsilon,\beta) = \ln\left(1+u+\sqrt{2u+u^2}\right)\sigma^2$$

where

$$u = \frac{\ln\left(1+\frac{\alpha^2(p-s+1)}{2}\right)}{2k^{1-2\beta}}.$$

If $\alpha \in (0, \frac{1}{2})$ *and* $k^{-\beta}u < 1$*, then for all* $\mu \leq \mu_{\min}$*,*

$$\inf_{\widehat{\mu}} \sup_{M \in \mathbb{M}[\mu,s]} \mathbb{P}_M[S(\widehat{\mu}) \neq S(M)] \geq \frac{1}{2}(1-\alpha)$$

where $\mathbb{M}[\mu,s]$ *is given by* (5).

The result can be interpreted in words in the following way: whatever the estimation procedure $\widehat{\mu}$, there exists some matrix $M \in \mathbb{M}[\mu_{\min}, s]$ such that the probability of incorrectly identifying the support $S(M)$ is bounded away from zero. In the next section, we will see that some estimation procedures achieve the lower bound given in Theorem 1.

3. Upper Bounds on the Support Recovery

In this section, we present sufficient conditions on $(n, p, k, \varepsilon, \mu_{\min})$ for different estimation procedures, so that

$$\mathbb{P}[S(\widehat{\boldsymbol{\mu}}) \neq S] \leq \alpha.$$

Let $\alpha', \delta' > 0$ be two parameters such that $\alpha' + \delta' = \alpha$. The parameter α' controls the probability of making a type one error

$$\mathbb{P}[\exists i \in [p] \ : \ i \in S(\widehat{\boldsymbol{\mu}}) \text{ and } i \notin S] \leq \alpha',$$

that is, the parameter α' upper bounds the probability that there is a zero row of the matrix M that is estimated as a non-zero row. Likewise, the parameter δ' controls the probability of making a type two error

$$\mathbb{P}[\exists i \in [p] \ : \ i \notin S(\widehat{\boldsymbol{\mu}}) \text{ and } i \in S] \leq \delta',$$

that is, the parameter δ' upper bounds the probability that there is a non-zero row of the matrix M that is estimated as a zero row.

The control of the type one and type two errors is established through the tuning parameter λ. It can be seen that if the parameter λ is chosen such that, for all $i \in S$, it holds that $\mathbb{P}[i \notin S(\widehat{\boldsymbol{\mu}})] \leq \delta'/s$ and, for all $i \in S^c$, it hold that $\mathbb{P}[i \in S(\widehat{\boldsymbol{\mu}})] \leq \alpha'/(p-s)$, then using the union bound we have that $\mathbb{P}[S(\widehat{\boldsymbol{\mu}}) \neq S] \leq \alpha$. In the following subsections, we will use the outlined strategy to choose λ for different estimation procedures.

3.1 Upper Bounds for the Lasso

Recall that the Lasso estimator is given as

$$\widehat{\boldsymbol{\mu}}^{\ell_1} = \underset{\boldsymbol{\mu} \in \mathbb{R}^{p \times k}}{\operatorname{argmin}} \ \frac{1}{2}||\mathbf{Y} - \boldsymbol{\mu}||_F^2 + \lambda||\boldsymbol{\mu}||_1.$$

It is easy to see that the solution of the above estimation problem is given as the following soft-thresholding operation

$$\widehat{\mu}_{ij}^{\ell_1} = \left(1 - \frac{\lambda}{|Y_{ij}|}\right)_+ Y_{ij}, \tag{6}$$

where $(x)_+ := \max(0,x)$. From (6), it is obvious that $i \in S(\widehat{\boldsymbol{\mu}}^{\ell_1})$ if and only if the maximum statistic, defined as

$$M_k(i) = \max_j |Y_{ij}|,$$

satisfies $M_k(i) \geq \lambda$. Therefore it is crucial to find the critical value of the parameter λ such that

$$\left\{ \begin{array}{rcll} \mathbb{P}[M_k(i) < \lambda] & < & \delta'/s & i \in S \\ \mathbb{P}[M_k(i) \geq \lambda] & < & \alpha'/(p-s) & i \in S^c. \end{array} \right.$$

We start by controlling the type one error. For $i \in S^c$ it holds that

$$\mathbb{P}[M_k(i) \geq \lambda] \leq k\mathbb{P}[|\mathcal{N}(0,\sigma^2)| \geq \lambda] \leq \frac{2k\sigma}{\sqrt{2\pi}\lambda} \exp\big(-\frac{\lambda^2}{2\sigma^2}\big) \tag{7}$$

using a standard tail bound for the Normal distribution. Setting the right hand side to $\alpha'/(p-s)$ in the above display, we obtain that λ can be set as

$$\lambda = \sigma\sqrt{2\ln\frac{2k(p-s)}{\sqrt{2\pi}\alpha'}} \tag{8}$$

and (7) holds as soon as $2\ln\frac{2k(p-s)}{\sqrt{2\pi}\alpha'} \geq 1$. Next, we deal with the type two error. Let

$$\pi_k = \mathbb{P}[|(1-\varepsilon)\mathcal{N}(0,\sigma^2) + \varepsilon\mathcal{N}(\mu_{\min},\sigma^2)| > \lambda]. \tag{9}$$

Then for $i \in S$, $\mathbb{P}[M_k(i) < \lambda] \leq \mathbb{P}[\mathrm{Bin}(k,\pi_k) = 0]$, where $\mathrm{Bin}(k,\pi_k)$ denotes the binomial random variable with parameters (k,π_k). Control of the type two error is going to be established through careful analysis of π_k for various regimes of problem parameters.

Theorem 2 *Let λ be defined by* (8). *Suppose $\mu_{\min}$ satisfies one of the following two cases:*

(i) $\mu_{\min} = \sigma\sqrt{2r\ln k}$ *where*

$$r > \left(\sqrt{1+C_{k,p,s}} - \sqrt{1-\beta}\right)^2$$

with

$$C_{k,p,s} = \frac{\ln\frac{2(p-s)}{\sqrt{2\pi}\alpha'}}{\ln k}$$

and $\lim_{n\to\infty} C_{k,p,s} \in [0,\infty)$*;*

(ii) $\mu_{\min} \geq \lambda$ *when*

$$\lim_{n\to\infty} \frac{\ln k}{\ln(p-s)} = 0$$

and $k^{1-\beta}/2 \geq \ln(s/\delta')$.

Then

$$\mathbb{P}[S(\widehat{\boldsymbol{\mu}}^{\ell_1}) \neq S] \leq \alpha.$$

The proof is given in Section 6.2. The two different cases describe two different regimes characterized by the ratio of $\ln k$ and $\ln(p-s)$.

Now we can compare the lower bound on $\mu_{\min}^2$ from Theorem 1 and the upper bound from Theorem 2. Without loss of generality we assume that $\sigma = 1$. We have that when $\beta < 1/2$ the lower bound is of the order $O\left(\ln\left(k^{\beta-1/2}\ln(p-s)\right)\right)$ and the upper bound is of the order $\ln(k(p-s))$. Ignoring the logarithmic terms in p and s, we have that the lower bound is of the order $\widetilde{O}(k^{\beta-1/2})$ and the upper bound is of the order $\widetilde{O}(\ln k)$, which implies that the Lasso does not achieve the lower bound when the non-zero rows are dense. When the non-zero rows are sparse, $\beta > 1/2$, we have that both the lower and upper bound are of the order $\widetilde{O}(\ln k)$ (ignoring the terms depending on p and s).

3.2 Upper Bounds for the Group Lasso

Recall that the group Lasso estimator is given as

$$\widehat{\boldsymbol{\mu}}^{\ell_1/\ell_2} = \underset{\boldsymbol{\mu}\in\mathbb{R}^{p\times k}}{\operatorname{argmin}} \frac{1}{2}||\mathbf{Y}-\boldsymbol{\mu}||_F^2 + \lambda\sum_{i\in[p]}||\boldsymbol{\theta}_i||_2,$$

where $\boldsymbol{\theta}_i = (\mu_{ij})_{j\in[k]}$. The group Lasso estimator can be obtained in a closed form as a result of the following thresholding operation (see, for example, Friedman et al., 2010)

$$\widehat{\boldsymbol{\theta}}_i^{\ell_1/\ell_2} = \left(1 - \frac{\lambda}{||Y_{i\cdot}||^2}\right)_+ Y_{i\cdot} \tag{10}$$

where $Y_{i\cdot}$ is the i^{th} row of the data. From (10), it is obvious that $i \in S(\widehat{\boldsymbol{\mu}}^{\ell_1/\ell_2})$ if and only if the statistic defined as

$$S_k(i) = \sum_j Y_{ij}^2,$$

satisfies $S_k(i) \geq \lambda$. The choice of λ is crucial for the control of type one and type two errors. We use the following result, which directly follows from Theorem 2 in Baraud (2002).

Lemma 3 *Let $\{Y_i = f_i + \sigma\xi_i\}_{i\in[n]}$ be a sequence of independent observations, where $f = \{f_i\}_{i\in[n]}$ is a sequence of numbers, $\xi_i \overset{iid}{\sim} \mathcal{N}(0,1)$ and σ is a known positive constant. Suppose that $t_{n,\alpha} \in \mathbb{R}$ satisfies $\mathbb{P}[\chi_n^2 > t_{n,\alpha}] \leq \alpha$. Let*

$$\phi_\alpha = I\{\sum_{i\in[n]} Y_i^2 \geq t_{n,\alpha}\sigma^2\}$$

be a test for $f = 0$ versus $f \neq 0$. Then the test ϕ_α satisfies

$$\mathbb{P}[\phi_\alpha = 1] \leq \alpha$$

when $f = 0$ and

$$\mathbb{P}[\phi_\alpha = 0] \leq \delta$$

for all f such that

$$||f||_2^2 \geq 2(\sqrt{5}+4)\sigma^2 \ln\left(\frac{2e}{\alpha\delta}\right)\sqrt{n}.$$

Proof This follows immediately from Theorem 2 in Baraud (2002). ■

It follows directly from lemma 3 that setting

$$\lambda = t_{n,\alpha'/(p-s)}\sigma^2 \tag{11}$$

will control the probability of type one error at the desired level, that is,

$$\mathbb{P}[S_k(i) \geq \lambda] \leq \alpha'/(p-s), \qquad \forall i \in S^c.$$

The following theorem gives us the control of the type two error.

Theorem 4 *Let* $\lambda = t_{n,\alpha'/(p-s)}\sigma^2$. *Then*

$$\mathbb{P}[S(\widehat{\boldsymbol{\mu}}^{\ell_1/\ell_2}) \neq S] \leq \alpha$$

if

$$\mu_{\min} \geq \sigma\sqrt{2(\sqrt{5}+4)}\sqrt{\frac{k^{-1/2+\beta}}{1-c}}\sqrt{\ln\frac{2e(2s-\delta')(p-s)}{\alpha'\delta'}}$$

where $c = \sqrt{2\ln(2s/\delta')/k^{1-\beta}}$.

The proof is given in Section 6.3.

Using Theorem 1 and Theorem 4 we can compare the lower bound on $\mu_{\min}^2$ and the upper bound. Without loss of generality we assume that $\sigma = 1$. When each non-zero row is dense, that is, when $\beta < 1/2$, we have that both lower and upper bounds are of the order $\widetilde{O}(k^{\beta-1/2})$ (ignoring the logarithmic terms in p and s). This suggest that the group Lasso performs better than the Lasso for the case where there is a lot of feature sharing between different tasks. Recall from previous section that the Lasso in this setting does not have the optimal dependence on k. However, when $\beta > 1/2$, that is, in the sparse non-zero row regime, we see that the lower bound is of the order $\widetilde{O}(\ln(k))$ whereas the upper bound is of the order $\widetilde{O}(k^{\beta-1/2})$. This implies that the group Lasso does not have optimal dependence on k in the sparse non-zero row setting.

3.3 Upper Bounds for the Group Lasso with the Mixed $(\infty,1)$ Norm

In this section, we analyze the group Lasso estimator with the mixed $(\infty,1)$ norm, defined as

$$\widehat{\boldsymbol{\mu}}^{\ell_1/\ell_\infty} = \operatorname*{argmin}_{\boldsymbol{\mu}\in\mathbb{R}^{p\times k}} \frac{1}{2}||\mathbf{Y}-\boldsymbol{\mu}||_F^2 + \lambda\sum_{i\in[p]}||\boldsymbol{\theta}_i||_\infty,$$

where $\boldsymbol{\theta}_i = (\mu_{ij})_{j\in[k]}$. The closed form solution for $\widehat{\boldsymbol{\mu}}^{\ell_1/\ell_\infty}$ can be obtained (see Liu et al., 2009), however, we are only going to use the following lemma.

Lemma 5 *(Liu et al., 2009)* $\widehat{\boldsymbol{\theta}}_i^{\ell_1/\ell_\infty} = \mathbf{0}$ *if and only if* $\sum_j |Y_{ij}| \leq \lambda$.

Proof See the proof of Proposition 5 in Liu et al. (2009). ■

Suppose that the penalty parameter λ is set as

$$\lambda = k\sigma\sqrt{2\ln\frac{k(p-s)}{\alpha'}}. \tag{12}$$

It follows immediately using a tail bound for the Normal distribution that

$$\mathbb{P}[\sum_j |Y_{ij}| \geq \lambda] \leq k \max_j \mathbb{P}[|Y_{ij}| \geq \lambda/k] \leq \alpha'/(p-s), \qquad \forall i \in S^c,$$

which implies that the probability of the type one error is controlled at the desired level.

Theorem 6 *Let the penalty parameter* λ *be defined by* (12). *Then*

$$\mathbb{P}[S(\widehat{\boldsymbol{\mu}}^{\ell_1/\ell_\infty}) \neq S] \leq \alpha$$

if

$$\mu_{\min} \;\geq\; \frac{1+\tau}{1-c} k^{-1+\beta} \lambda$$

where $c = \sqrt{2\ln(2s/\delta')/k^{1-\beta}}$ *and* $\tau = \sigma\sqrt{2k\ln\frac{2s-\delta'}{\delta'}}/\lambda$.

The proof is given in Section 6.4.

Comparing upper bounds for the Lasso and the group Lasso with the mixed $(2,1)$ norm with the result of Theorem 6, we can see that both the Lasso and the group Lasso have better dependence on k than the group Lasso with the mixed $(\infty,1)$ norm. The difference becomes more pronounced as β increases. This suggest that we should be very cautious when using the group Lasso with the mixed $(\infty,1)$ norm, since as soon as the tasks do not share exactly the same features, the other two procedures have much better performance on identifying the set of non-zero rows.

4. Simulation Results

We conduct a small-scale empirical study of the performance of the Lasso and the group Lasso (both with the mixed $(2,1)$ norm and with the mixed $(\infty,1)$ norm). Our empirical study shows that the theoretical findings of Section 3 describe sharply the behavior of procedures even for small sample studies. In particular, we demonstrate that as the minimum signal level $\mu_{\min}$ varies in the model (3), our theory sharply determines points at which probability of identifying non-zero rows of matrix M successfully transitions from 0 to 1 for different procedures.

The simulation procedure can be described as follows. Without loss of generality we let $S = [s]$ and draw the samples $\{Y_{ij}\}_{i\in[p],j\in[k]}$ according to the model in (3). The total number of rows p is varied in $\{128, 256, 512, 1024\}$ and the number of columns is set to $k = \lfloor p\log_2(p)\rfloor$. The sparsity of each non-zero row is controlled by changing the parameter β in $\{0, 0.25, 0.5, 0.75\}$ and setting $\varepsilon = k^{-\beta}$. The number of non-zero rows is set to $s = \lfloor \log_2(p)\rfloor$, the sample size is set to $n = 0.1p$ and $\sigma_0 = 1$. The parameters α' and δ' are both set to 0.01. For each setting of the parameters, we report our results averaged over 1000 simulation runs. Simulations with other choices of parameters n, s and k have been tried out, but the results were qualitatively similar and, hence, we do not report them here.

The regularization parameter λ is chosen according to Equations (8), (11) and (12), which assume that the noise level σ_0 is known. In practice, estimating the standard deviation of the noise in high-dimensions is a hard problem and practitioners often use cross-validation as a data-driven way to choose the penalty parameter. For recent work on data-driven tuning of the penalty parameters, we refer the reader to Arlot and Bach (2009).

4.1 Lasso

We investigate the performance on the Lasso for the purpose of estimating the set of non-zero rows, S. Figure 1 plots the probability of success as a function of the signal strength. On the same figure we plot the probability of success for the group Lasso with both $(2,1)$ and $(\infty,1)$-mixed norms. Using theorem 2, we set

$$\mu_{\text{lasso}} = \sqrt{2(r+0.001)\ln k} \tag{13}$$

where r is defined in theorem 2. Next, we generate data according to (3) with all elements $\{\mu_{ij}\}$ set to $\mu = \rho\mu_{\text{lasso}}$, where $\rho \in [0.05, 2]$. The penalty parameter λ is chosen as in (8). Figure 1 plots probability of success as a function of the parameter ρ, which controls the signal strength. This probability transitions very sharply from 0 to 1. A rectangle on a horizontal line represents points at which the probability $\mathbb{P}[\widehat{S} = S]$ is between 0.05 and 0.95. From each subfigure in Figure 1, we can observe that the probability of success for the Lasso transitions from 0 to 1 for the same value of the parameter ρ for different values of p, which indicates that, except for constants, our theory correctly characterizes the scaling of $\mu_{\min}$. In addition, we can see that the Lasso outperforms the group Lasso (with $(2,1)$-mixed norm) when each non-zero row is very sparse (the parameter β is close to one).

4.2 Group Lasso

Next, we focus on the empirical performance of the group Lasso with the mixed $(2,1)$ norm. Figure 2 plots the probability of success as a function of the signal strength. Using theorem 4, we set

$$\mu_{\text{group}} = \sigma\sqrt{2(\sqrt{5}+4)}\sqrt{\frac{k^{-1/2+\beta}}{1-c}}\sqrt{\ln\frac{(2s-\delta')(p-s)}{\alpha'\delta'}} \tag{14}$$

where c is defined in theorem 4. Next, we generate data according to (3) with all elements $\{\mu_{ij}\}$ set to $\mu = \rho\mu_{\text{group}}$, where $\rho \in [0.05, 2]$. The penalty parameter λ is given by (11). Figure 2 plots probability of success as a function of the parameter ρ, which controls the signal strength. A rectangle on a horizontal line represents points at which the probability $\mathbb{P}[\widehat{S} = S]$ is between 0.05 and 0.95. From each subfigure in Figure 2, we can observe that the probability of success for the group Lasso transitions from 0 to 1 for the same value of the parameter ρ for different values of p, which indicated that, except for constants, our theory correctly characterizes the scaling of $\mu_{\min}$. We observe also that the group Lasso outperforms the Lasso when each non-zero row is not too sparse, that is, when there is a considerable overlap of features between different tasks.

4.3 Group Lasso with the Mixed $(\infty,1)$ Norm

Next, we focus on the empirical performance of the group Lasso with the mixed $(\infty,1)$ norm. Figure 3 plots the probability of success as a function of the signal strength. Using theorem 6, we set

$$\mu_{\text{infty}} = \frac{1+\tau}{1-c}k^{-1+\beta}\lambda \tag{15}$$

where τ and c are defined in theorem 6 and λ is given by (12). Next, we generate data according to (3) with all elements $\{\mu_{ij}\}$ set to $\mu = \rho\mu_{\text{infty}}$, where $\rho \in [0.05, 2]$. Figure 3 plots probability of success as a function of the parameter ρ, which controls the signal strength. A rectangle on a horizontal line represents points at which the probability $\mathbb{P}[\widehat{S} = S]$ is between 0.05 and 0.95. From each subfigure

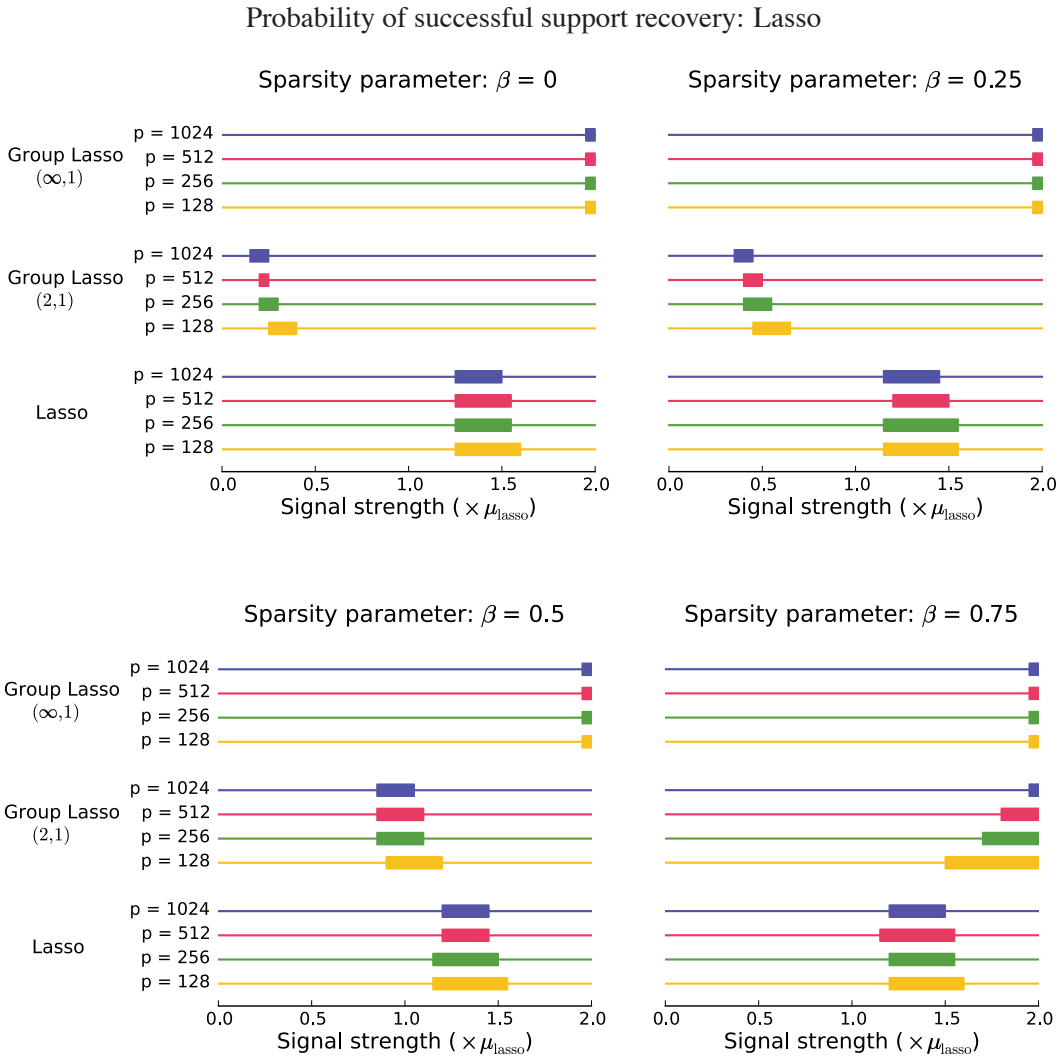


Figure 1: The probability of success for the Lasso for the problem of estimating S plotted against the signal strength, which is varied as a multiple of μ_{lasso} defined in (13). A rectangle on each horizontal line represents points at which the probability $\mathbb{P}[\widehat{S} = S]$ is between 0.05 and 0.95. To the left of the rectangle the probability is smaller than 0.05, while to the right the probability is larger than 0.95. Different subplots represent the probability of success as the sparsity parameter β changes.

in Figure 3, we can observe that the probability of success for the group Lasso transitions from 0 to 1 for the same value of the parameter ρ for different values of p, which indicated that, except for constants, our theory correctly characterizes the scaling of μ_{min}. We also observe that the group Lasso with the mixed $(\infty, 1)$ norm never outperforms the Lasso or the group Lasso with the mixed $(2, 1)$ norm.

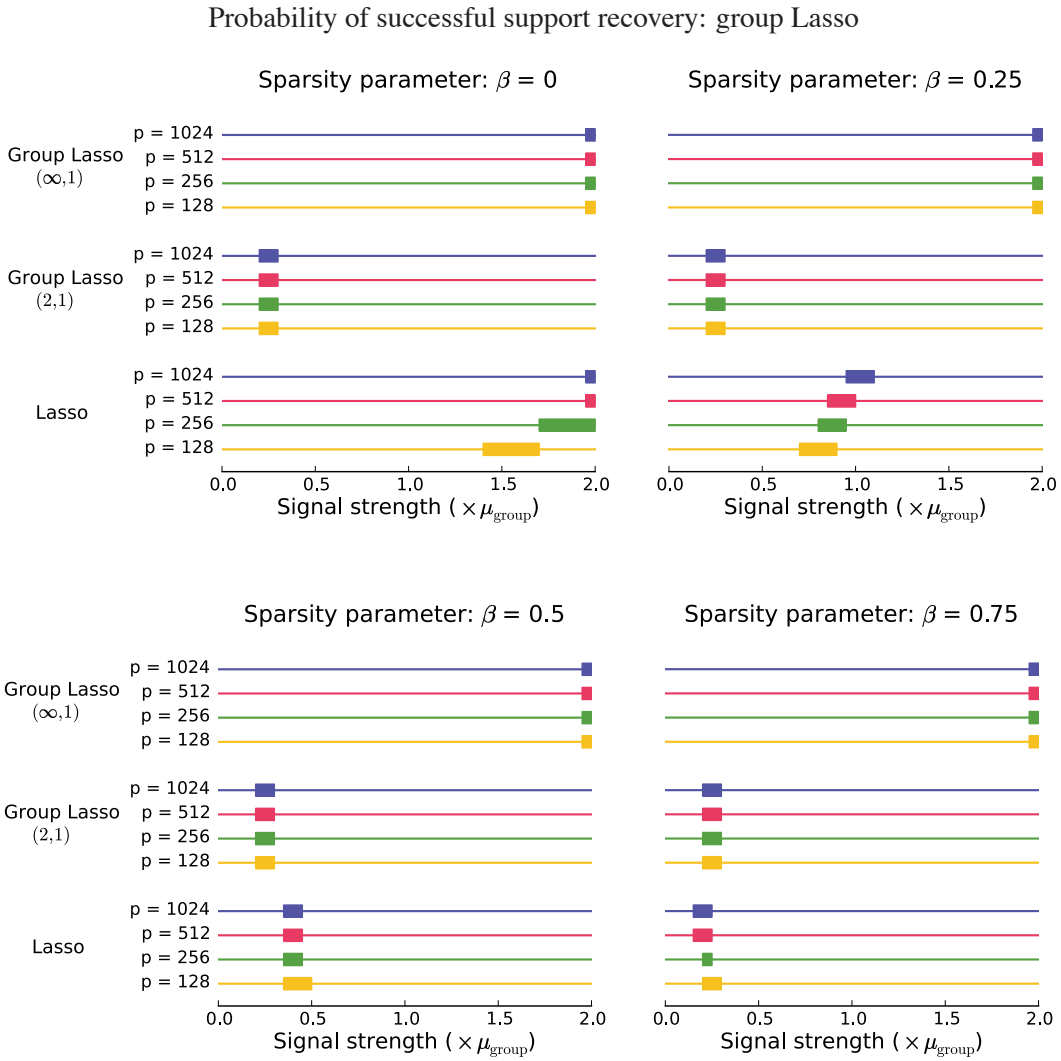


Figure 2: The probability of success for the group Lasso for the problem of estimating S plotted against the signal strength, which is varied as a multiple of μ_{group} defined in (14). A rectangle on each horizontal line represents points at which the probability $\mathbb{P}[\widehat{S} = S]$ is between 0.05 and 0.95. To the left of the rectangle the probability is smaller than 0.05, while to the right the probability is larger than 0.95. Different subplots represent the probability of success as the sparsity parameter β changes.

5. Discussion

We have studied the benefits of task sharing in sparse problems. Under many scenarios, the group lasso outperforms the lasso. The ℓ_1/ℓ_2 penalty seems to be a much better choice for the group lasso than the ℓ_1/ℓ_∞. However, as pointed out to us by Han Liu, for screening, where false discoveries are less important than accurate recovery, it is possible that the ℓ_1/ℓ_∞ penalty could be useful. From the results in Section 3, we can further conclude that the Lasso procedure performs better than the group Lasso when each non-zero row is sparse, while the group Lasso (with the mixed $(2,1)$ norm)

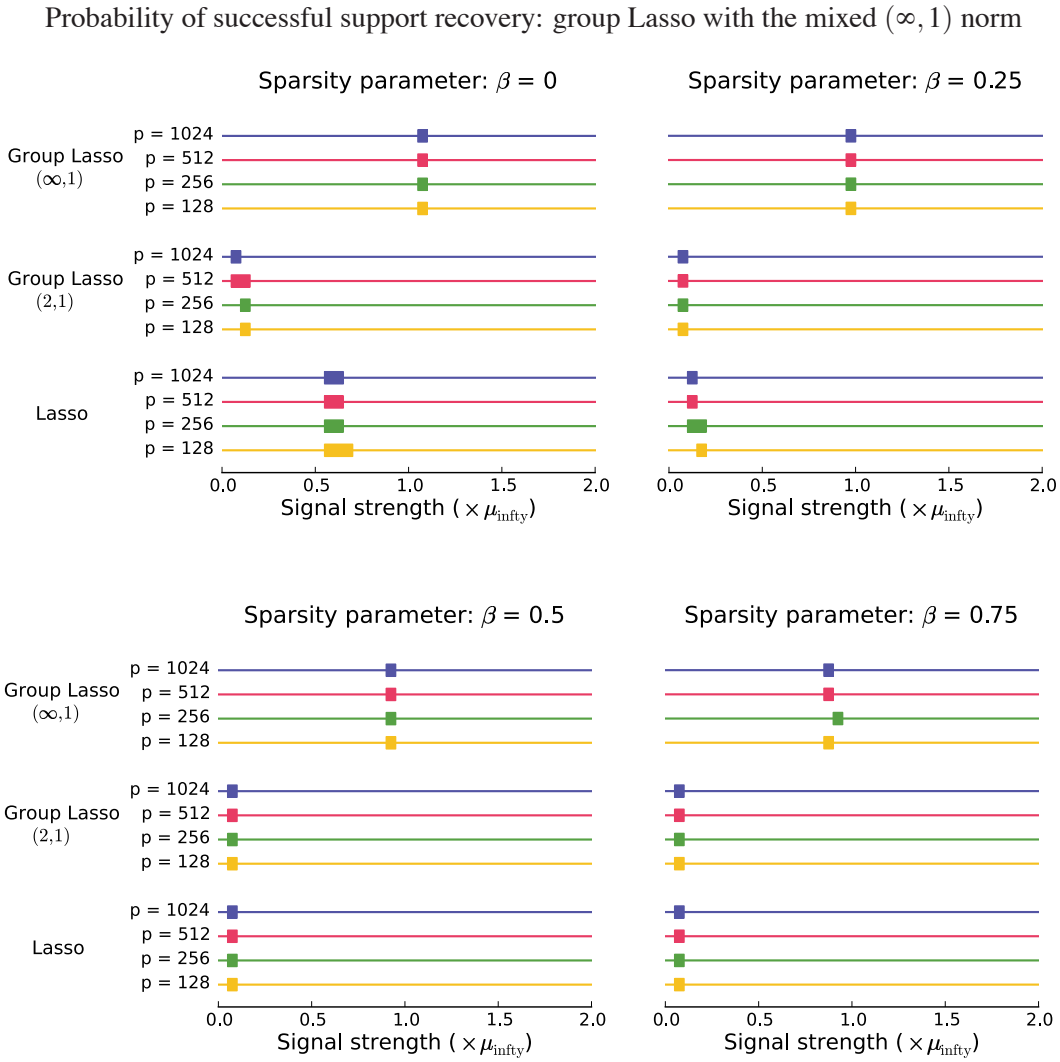


Figure 3: The probability of success for the group Lasso with mixed $(\infty, 1)$ norm for the problem of estimating S plotted against the signal strength, which is varied as a multiple of μ_{infty} defined in (15). A rectangle on each horizontal line represents points at which the probability $\mathbb{P}[\widehat{S} = S]$ is between 0.05 and 0.95. To the left of the rectangle the probability is smaller than 0.05, while to the right the probability is larger than 0.95. Different subplots represent the probability of success as the sparsity parameter β changes.

performs better when each non-zero row is dense. Since in many practical situations one does not know how much overlap there is between different tasks, it would be useful to combine the Lasso and the group Lasso in order to improve the performance. For example, one can take the union of the Lasso and the group Lasso estimate, $\widehat{S} = S(\widehat{\boldsymbol{\mu}}^{\ell_1}) \cup S(\widehat{\boldsymbol{\mu}}^{\ell_1/\ell_2})$. The suggested approach has the advantage that one does not need to know in advance which estimation procedure to use. While such a combination can be justified in the Normal means problem as a way to increase the power to detect the non-zero rows, it is not clear whether the same approach can be justified in the multi-task regression model (1).

The analysis of the Normal means model in (3) provides insights into the theoretical results we could expect in the conventional multi-task learning given in (1). However, there is no direct way to translate our results into valid results for the model in (1); a separate analysis needs to be done in order to establish sharp theoretical results.

6. Proofs

This section collects technical proofs of the results presented in the paper. Throughout the section we use $c_1, c_2, \ldots$ to denote positive constants whose value may change from line to line.

6.1 Proof of Theorem 1

Without loss of generality, we may assume $\sigma = 1$. Let $\phi(u)$ be the density of $\mathcal{N}(0,1)$ and define $\mathbb{P}_{\mathbf{0}}$ and $\mathbb{P}_{\mathbf{1}}$ to be two probability measures on $\mathbb{R}^k$ with the densities with respect to the Lebesgue measure given as

$$f_0(a_1,\ldots,a_k) = \prod_{j\in[k]} \phi(a_j) \tag{16}$$

and

$$f_1(a_1,\ldots,a_k) = \mathbb{E}_Z \mathbb{E}_m \mathbb{E}_\xi \prod_{j\in m} \phi(a_j - \xi_j \mu_{\min}) \prod_{j\notin m} \phi(a_j) \tag{17}$$

where $Z \sim \mathrm{Bin}(k, k^{-\beta})$, m is a random variable uniformly distributed over $\mathcal{M}(Z,k)$ and $\{\xi_j\}_{j\in[k]}$ is a sequence of Rademacher random variables, independent of Z and m. A Rademacher random variable takes values ± 1 with probability $\frac{1}{2}$.

To simplify the discussion, suppose that $p-s+1$ is divisible by 2. Let $T = (p-s+1)/2$. Using $\mathbb{P}_{\mathbf{0}}$ and $\mathbb{P}_{\mathbf{1}}$, we construct the following three measures,

$$\widetilde{\mathbb{Q}} = \mathbb{P}_{\mathbf{1}}^{s-1} \otimes \mathbb{P}_{\mathbf{0}}^{p-s+1},$$

$$\mathbb{Q}_0 = \frac{1}{T} \sum_{\substack{j\in\{s,\ldots,p\} \\ j \text{ odd}}} \mathbb{P}_{\mathbf{1}}^{s-1} \otimes \mathbb{P}_{\mathbf{0}}^{j-s} \otimes \mathbb{P}_{\mathbf{1}} \otimes \mathbb{P}_{\mathbf{0}}^{p-j}$$

and

$$\mathbb{Q}_1 = \frac{1}{T} \sum_{\substack{j\in\{s,\ldots,p\} \\ j \text{ even}}} \mathbb{P}_{\mathbf{1}}^{s-1} \otimes \mathbb{P}_{\mathbf{0}}^{j-s} \otimes \mathbb{P}_{\mathbf{1}} \otimes \mathbb{P}_{\mathbf{0}}^{p-j}.$$

It holds that

$$\begin{aligned} \inf_{\widehat{\mu}} \sup_{M\in\mathbb{M}} \mathbb{P}_M[S(M) \neq S(\widehat{\mu})] &\geq \inf_{\Psi} \max \Big(\mathbb{Q}_0(\Psi = 1), \mathbb{Q}_1(\Psi = 0) \Big) \\ &\geq \frac{1}{2} - \frac{1}{2} ||\mathbb{Q}_0 - \mathbb{Q}_1||_1, \end{aligned}$$

where the infimum is taken over all tests Ψ taking values in $\{0,1\}$ and $||\cdot||_1$ is the total variation distance between probability measures. For a readable introduction on lower bounds on the minimax probability of error, see Section 2 in Tsybakov (2009). In particular, our approach is related to the one described in Section 2.7.4. We proceed by upper bounding the total variation distance between

$\mathbb{Q}_0$ and $\mathbb{Q}_1$. Let $g = d\mathbb{P}_{\mathbf{1}}/d\mathbb{P}_{\mathbf{0}}$ and let $u_i \in \mathbb{R}^k$ for each $i \in [p]$, then

$$\begin{aligned}
&\frac{d\mathbb{Q}_0}{d\widetilde{\mathbb{Q}}}(u_1,\ldots,u_p) \\
&\quad= \frac{1}{T}\sum_{\substack{j\in\{s,\ldots,p\}\\ j \text{ even}}} \prod_{i\in\{1,\ldots,s-1\}} \frac{d\mathbb{P}_{\mathbf{1}}}{d\mathbb{P}_{\mathbf{1}}}(u_i) \prod_{i\in\{s,\ldots,j-1\}} \frac{d\mathbb{P}_{\mathbf{0}}}{d\mathbb{P}_{\mathbf{0}}}(u_i) \frac{d\mathbb{P}_{\mathbf{1}}}{d\mathbb{P}_{\mathbf{0}}}(u_j) \prod_{i\in\{j+1,\ldots,p\}} \frac{d\mathbb{P}_{\mathbf{0}}}{d\mathbb{P}_{\mathbf{0}}}(u_i) \\
&\quad= \frac{1}{T}\sum_{\substack{j\in\{s,\ldots,p\}\\ j \text{ even}}} g(u_j)
\end{aligned}$$

and, similarly, we can compute $d\mathbb{Q}_1/d\widetilde{\mathbb{Q}}$. The following holds

$$\begin{aligned}
&\|\mathbb{Q}_0 - \mathbb{Q}_1\|_1^2 \\
&\quad= \left(\int \Big|\frac{1}{T}\Big(\sum_{\substack{j\in\{s,\ldots,p\}\\ j \text{ even}}} g(u_j) - \sum_{\substack{j\in\{s,\ldots,p\}\\ j \text{ odd}}} g(u_j)\Big)\Big| \prod_{i\in\{s,\ldots,p\}} d\mathbb{P}_{\mathbf{0}}(u_i)\right)^2 \\
&\quad\le \frac{1}{T^2}\int \Big(\sum_{\substack{j\in\{s,\ldots,p\}\\ j \text{ even}}} g(u_j) - \sum_{\substack{j\in\{s,\ldots,p\}\\ j \text{ odd}}} g(u_j)\Big)^2 \prod_{i\in\{s,\ldots,p\}} d\mathbb{P}_{\mathbf{0}}(u_i) \\
&\quad= \frac{2}{T}\big(\mathbb{P}_{\mathbf{0}}(g^2) - 1\big),
\end{aligned} \tag{18}$$

where the last equality follows by observing that

$$\int \sum_{\substack{j\in\{s,\ldots,p\}\\ j \text{ even}}} \sum_{\substack{j'\in\{s,\ldots,p\}\\ j' \text{ even}}} g(u_j)g(u_{j'}) \prod_{\substack{i\in\{s,\ldots,p\}\\ i \text{ even}}} d\mathbb{P}_{\mathbf{0}}(u_i) = T\ \mathbb{P}_{\mathbf{0}}(g^2) + T^2 - T$$

and

$$\int \sum_{\substack{j\in\{s,\ldots,p\}\\ j \text{ even}}} \sum_{\substack{j'\in\{s,\ldots,p\}\\ j' \text{ odd}}} g(u_j)g(u_{j'}) \prod_{i\in\{s,\ldots,p\}} d\mathbb{P}_{\mathbf{0}}(u_i) = T^2.$$

Next, we proceed to upper bound $\mathbb{P}_{\mathbf{0}}(g^2)$, using some ideas presented in the proof of Theorem 1 in Baraud (2002). Recall definitions of f_0 and f_1 in (16) and (17) respectively. Then $g = d\mathbb{P}_{\mathbf{1}}/d\mathbb{P}_{\mathbf{0}} = f_1/f_0$ and we have

$$\begin{aligned}
g(a_1,\ldots,a_k) &= \mathbb{E}_Z\mathbb{E}_m\mathbb{E}_\xi\left[\exp\Big(-\frac{Z\mu_{\min}^2}{2} + \mu_{\min}\sum_{j\in m}\xi_j a_j\Big)\right] \\
&= \mathbb{E}_Z\left[\exp\Big(-\frac{Z\mu_{\min}^2}{2}\Big)\,\mathbb{E}_m\Big[\prod_{j\in m}\cosh(\mu_{\min}a_j)\Big]\right].
\end{aligned}$$

Furthermore, let $Z' \sim \mathrm{Bin}(k, k^{-\beta})$ be independent of Z and m' uniformly distributed over $\mathcal{M}(Z', k)$. The following holds

$$\begin{aligned}
&\mathbb{P}_{\mathbf{0}}(g^2) \\
&= \mathbb{P}_{\mathbf{0}}\Big(\mathbb{E}_{Z',Z}\Big[\exp\Big(-\frac{(Z+Z')\mu_{\min}^2}{2}\Big)\,\mathbb{E}_{m,m'}\prod_{j\in m}\cosh(\mu_{\min}a_j)\prod_{j\in m'}\cosh(\mu_{\min}a_j)\Big]\Big) \\
&= \mathbb{E}_{Z',Z}\Big[\exp\Big(-\frac{(Z+Z')\mu_{\min}^2}{2}\Big) \\
&\qquad\qquad \mathbb{E}_{m,m'}\Big[\prod_{j\in m\cap m'}\int\cosh^2(\mu_{\min}a_j)\phi(a_j)da_j \\
&\qquad\qquad\qquad \prod_{j\in m\triangle m'}\int\cosh(\mu_{\min}a_j)\phi(a_j)da_j\Big]\Big],
\end{aligned}$$

where we use $m\triangle m'$ to denote $(m\cup m')\backslash(m\cap m')$. By direct calculation, we have that

$$\int\cosh^2(\mu_{\min}a_j)\phi(a_j)da_j = \exp(\mu_{\min}^2)\cosh(\mu_{\min}^2)$$

and

$$\int\cosh(\mu_{\min}a_j)\phi(a_j)da_j = \exp(\mu_{\min}^2/2).$$

Since $\frac{1}{2}|m\triangle m'| + |m\cap m'| = (Z+Z')/2$, we have that

$$\begin{aligned}
\mathbb{P}_{\mathbf{0}}(g^2) &= \mathbb{E}_{Z,Z'}\Big[E_{m,m'}\Big[\big(\cosh(\mu_{\min}^2)\big)^{|m\cap m'|}\Big]\Big] \\
&= \mathbb{E}_{Z,Z'}\Big[\sum_{j=0}^{k}p_j\big(\cosh(\mu_{\min}^2)\big)^j\Big] \\
&= \mathbb{E}_{Z,Z'}\Big[\mathbb{E}_X\Big[\cosh(\mu_{\min}^2)^X\Big]\Big],
\end{aligned}$$

where

$$p_j = \begin{cases} 0 & \text{if } j < Z+Z'-k \text{ or } j > \min(Z,Z') \\ \frac{\binom{Z'}{j}\binom{k-Z'}{Z-j}}{\binom{k}{Z}} & \text{otherwise} \end{cases}$$

and $P[X=j] = p_j$. Therefore, X follows a hypergeometric distribution with parameters $k, Z, Z'/k$. [The first parameter denotes the total number of stones in an urn, the second parameter denotes the number of stones we are going to sample without replacement from the urn and the last parameter denotes the fraction of white stones in the urn.] Then following (Aldous, 1985, p. 173; see also Baraud 2002), we know that X has the same distribution as the random variable $\mathbb{E}[\widetilde{X}|\mathcal{T}]$ where $\widetilde{X}$ is a binomial random variable with parameters Z and Z'/k, and $\mathcal{T}$ is a suitable σ-algebra. By convexity, it follows that

$$\begin{aligned}
\mathbb{P}_{\mathbf{0}}(g^2) &\le \mathbb{E}_{Z,Z'}\Big[\mathbb{E}_{\widetilde{X}}\Big[\cosh(\mu_{\min}^2)^{\widetilde{X}}\Big]\Big] \\
&= \mathbb{E}_{Z,Z'}\Big[\exp\Big(Z\ln\Big(1+\frac{Z'}{k}\big(\cosh(\mu_{\min}^2)-1\big)\Big)\Big)\Big] \\
&= \mathbb{E}_{Z'}\mathbb{E}_Z\Big[\exp\Big(Z\ln\Big(1+\frac{Z'}{k}u\Big)\Big)\Big]
\end{aligned}$$

where $\mu_{\min}^2 = \ln(1+u+\sqrt{2u+u^2})$ with

$$u = \frac{\ln\left(1+\frac{\alpha^2 T}{2}\right)}{2k^{1-2\beta}}.$$

Continuing with our calculations, we have that

$$\begin{aligned}
\mathbb{P}_{\mathbf{0}}(g^2) &= \mathbb{E}_{Z'} \exp\left(k \ln\left(1+k^{-(1+\beta)} u Z'\right)\right) \\
&\leq \mathbb{E}_{Z'} \exp\left(k^{-\beta} u Z'\right) \\
&= \exp\left(k \ln\left(1+k^{-\beta}\left(\exp(k^{-\beta}u)-1\right)\right)\right) \\
&\leq \exp\left(k^{1-\beta}\left(\exp\left(k^{-\beta}u\right)-1\right)\right) \\
&\leq \exp\left(2k^{1-2\beta}u\right) \\
&= 1+\frac{\alpha^2 T}{2},
\end{aligned} \tag{19}$$

where the last inequality follows since $k^{-\beta}u < 1$ for all large p. Combining (19) with (18), we have that

$$\|\mathbb{Q}_0 - \mathbb{Q}_1\|_1 \leq \alpha,$$

which implies that

$$\inf_{\widehat{\mu}} \sup_{M \in \mathbb{M}} \mathbb{P}_M[S(M) \neq S(\widehat{\mu})] \geq \frac{1}{2} - \frac{1}{2}\alpha.$$

6.2 Proof of Theorem 2

Without loss of generality, we can assume that $\sigma = 1$ and rescale the final result. For λ given in (8), it holds that $\mathbb{P}[|\mathcal{N}(0,1) \geq \lambda] = o(1)$. For the probability defined in (9), we have the following lower bound

$$\pi_k = (1-\varepsilon)\mathbb{P}[|\mathcal{N}(0,1)| \geq \lambda] + \varepsilon\mathbb{P}[|\mathcal{N}(\mu_{\min},1)| \geq \lambda] \geq \varepsilon\mathbb{P}[\mathcal{N}(\mu_{\min},1) \geq \lambda].$$

We prove the two cases separately.
Case 1: *Large number of tasks*. By direct calculation

$$\pi_k \geq \varepsilon\mathbb{P}[\mathcal{N}(\mu_{\min},1) \geq \lambda] = \frac{1}{\sqrt{4\pi \log k}\left(\sqrt{1+C_{k,p,s}}-\sqrt{r}\right)} k^{-\beta-\left(\sqrt{1+C_{k,p,s}}-\sqrt{r}\right)^2} =: \underline{\pi_k}.$$

Since $1-\beta > \left(\sqrt{1+C_{k,p,s}}-\sqrt{r}\right)^2$, we have that $\mathbb{P}[\mathrm{Bin}(k,\pi_k)=0] \xrightarrow{n\to\infty} 0$. We can conclude that as soon as $k\underline{\pi_k} \geq \ln(s/\delta')$, it holds that $\mathbb{P}[S(\widehat{\mu}^{\ell_1}) \neq S] \leq \alpha$.
Case 2: *Medium number of tasks*. When $\mu_{\min} \geq \lambda$, it holds that

$$\pi_k \geq \varepsilon\mathbb{P}[\mathcal{N}(\mu_{\min},1) \geq \lambda] \geq \frac{k^{-\beta}}{2}.$$

We can conclude that as soon as $k^{1-\beta}/2 \geq \ln(s/\delta')$, it holds that $\mathbb{P}[S(\widehat{\mu}^{\ell_1}) \neq S] \leq \alpha$.

6.3 Proof of Theorem 4

Using a Chernoff bound, $\mathbb{P}[\mathrm{Bin}(k,k^{-\beta}) \leq (1-c)k^{1-\beta}] \leq \delta'/2s$ for $c = \sqrt{2\ln(2s/\delta')/k^{1-\beta}}$. For $i \in S$, we have that

$$\mathbb{P}[S_k(i) \leq \lambda] \leq \frac{\delta'}{2s} + \Big(1 - \frac{\delta'}{2s}\Big)\mathbb{P}\Big[S_k(i) \leq \lambda \;\Big|\; \Big\{||\theta_i||_2^2 \geq (1-c)k^{1-\beta}\mu_{\min}^2\Big\}\Big].$$

Therefore, using lemma 3 with $\delta = \delta'/(2s-\delta')$, if follows that $\mathbb{P}[S_k(i) \leq \lambda] \leq \delta'/(2s)$ for all $i \in S$ when

$$\mu_{\min} \geq \sigma\sqrt{2(\sqrt{5}+4)}\sqrt{\frac{k^{-1/2+\beta}}{1-c}}\sqrt{\ln\frac{2e(2s-\delta')(p-s)}{\alpha'\delta'}}.$$

Since $\lambda = t_{n,\alpha'/(p-s)}\sigma^2$, $\mathbb{P}[S_k(i) \geq \lambda] \leq \alpha'/(p-s)$ for all $i \in S^c$. We can conclude that $\mathbb{P}[S(\widehat{\boldsymbol{\mu}}^{\ell_1/\ell_2}) \neq S] \leq \alpha$.

6.4 Proof of Theorem 6

Without loss of generality, we can assume that $\sigma = 1$. Proceeding as in the proof of theorem 4, $\mathbb{P}[\mathrm{Bin}(k,k^{-\beta}) \leq (1-c)k^{1-\beta}] \leq \delta'/2s$ for $c = \sqrt{2\ln(2s/\delta')/k^{1-\beta}}$. Then for $i \in S$ it holds that

$$\mathbb{P}[\sum_j |Y_{ij}| \leq \lambda] \leq \frac{\delta'}{2s} + \Big(1 - \frac{\delta'}{2s}\Big)\mathbb{P}[(1-c)k^{1-\beta}\mu_{\min} + z_k \leq \lambda],$$

where $z_k \sim \mathcal{N}(0,k)$. Since $(1-c)k^{1-\beta}\mu_{\min} \geq (1+\tau)\lambda$, the right-hand side of the above display can upper bounded as

$$\frac{\delta'}{2s} + \Big(1 - \frac{\delta'}{2s}\Big)\mathbb{P}[\mathcal{N}(0,1) \geq \tau\lambda/\sqrt{k}] \leq \frac{\delta'}{2s} + \Big(1 - \frac{\delta'}{2s}\Big)\frac{\delta'}{2s-\delta'} \leq \frac{\delta'}{s}.$$

The above display gives us the desired control of the type two error, and we can conclude that $\mathbb{P}[S(\widehat{\boldsymbol{\mu}}^{\ell_1/\ell_\infty}) \neq S] \leq \alpha$.

Acknowledgments

We would like to thank Han Liu for useful discussions. We also thank two anonymous reviewers and the associate editor for comments that have helped improve the paper. The research reported here was supported in part by NSF grant CCF-0625879, AFOSR contract FA9550-09-1-0373, a grant from Google, and a graduate fellowship from Facebook.

References

David Aldous. Exchangeability and related topics. In *École d'Été de Probabilités de Saint-Flour XIII 1983*, pages 1–198. Springer, Berlin, 1985.

Andreas Argyriou, Theodoros Evgeniou, and Massimiliano Pontil. Convex multi-task feature learning. *Machine Learning*, 73(3):243–272, 2008.

Sylvain Arlot and Francis Bach. Data-driven calibration of linear estimators with minimal penalties. In *Advances in Neural Information Processing Systems 22*, pages 46–54. 2009.

Yannick Baraud. Non-asymptotic minimax rates of testing in signal detection. *Bernoulli*, 8(5): 577–606, 2002.

Larry Brown and Mark Low. Asymptotic equivalence of nonparametric regression and white noise. *Ann. Statist.*, 24(6):2384–2398, 1996.

Jianqing Fan and Runze Li. Variable selection via nonconcave penalized likelihood and its oracle properties. *J. Am. Statist. Ass.*, 96:1348–1360, 2001.

Jerome Friedman, Trevor Hastie, and Robert Tibshirani. A note on the group lasso and a sparse group lasso. Technical report, Stanford, 2010. Available at arXiv:1001.0736.

Seyoung Kim, Kyung-Ah Sohn, and Eric P. Xing. A multivariate regression approach to association analysis of a quantitative trait network. *Bioinformatics*, 25(12):i204–212, 2009.

Mladen Kolar and Eric P. Xing. Ultra-high dimensional multiple output learning with simultaneous orthogonal matching pursuit: Screening approach. In *AISTATS '10: Proc. 13th Int'l Conf. on Artifical Intelligence and Statistics*, pages 413–420, 2010.

Han Liu, Mark Palatucci, and Jian Zhang. Blockwise coordinate descent procedures for the multi-task lasso, with applications to neural semantic basis discovery. In *ICML '09: Proc. 26th Int'l Conf. on Machine Learning*, pages 649–656, New York, NY, USA, 2009.

Karim Lounici, Massimiliano Pontil, Alexandre Tsybakov, and Sara van de Geer. Taking advantage of sparsity in Multi-Task learning. In *COLT '09: Proc. Conf. on Learning Theory*, 2009.

Karim Lounici, Massimiliano Pontil, Alexandre B Tsybakov, and Sara van de Geer. Oracle inequalities and optimal inference under group sparsity. *Preprint*, 2010. Available at arXiv:1007.1771.

Sahand Negahban and Martin Wainwright. Phase transitions for high-dimensional joint support recovery. In *Advances in Neural Information Processing Systems 21*, pages 1161–1168, 2009.

Michael Nussbaum. Asymptotic equivalence of density estimation and gaussian white noise. *Ann. Statist.*, 24(6):2399–2430, 1996.

Guillaume Obozinski, Martin Wainwright, and Michael Jordan. Support union recovery in high-dimensional multivariate regression. *Ann. Statist.*, 39(1):1–47, 2011.

Alexandre Tsybakov. *Introduction to Nonparametric Estimation*. Springer, 2009.

Berwin Turlach, William Venables, and Stephen Wright. Simultaneous variable selection. *Technometrics*, 47(3):349–363, 2005. ISSN 0040-1706.

Sara van de Geer and Peter Bühlmann. On the conditions used to prove oracle results for the lasso. *Elec. J. Statist.*, 3:1360–1392, 2009.

Jian Zhang. *A Probabilistic Framework for Multitask Learning*. PhD thesis, Carnegie Mellon University, 2006.

Peng Zhao and Bin Yu. On model selection consistency of lasso. *J. Mach. Learn. Res.*, 7:2541–2563, 2006. ISSN 1533-7928.

Hui Zou and Ming Yuan. The $\mathtt{F}_\infty$-norm support vector machine. *Stat. Sin*, 18:379–398, 2008.

Journal of Machine Learning Research 12 (2011) 2437-2459 Submitted 7/10; Revised 2/11; Published 7/11

Parallel Algorithm for Learning Optimal Bayesian Network Structure

Yoshinori Tamada[*] TAMADA@IMS.U-TOKYO.AC.JP
Seiya Imoto IMOTO@IMS.U-TOKYO.AC.JP
Satoru Miyano[†] MIYANO@IMS.U-TOKYO.AC.JP
Human Genome Center
Institute of Medical Science, The University of Tokyo
4-6-1 Shirokanedai, Minato-ku, Tokyo 108-8639, Japan

Editor: Russ Greiner

Abstract

We present a parallel algorithm for the score-based optimal structure search of Bayesian networks. This algorithm is based on a dynamic programming (DP) algorithm having $O(n \cdot 2^n)$ time and space complexity, which is known to be the fastest algorithm for the optimal structure search of networks with n nodes. The bottleneck of the problem is the memory requirement, and therefore, the algorithm is currently applicable for up to a few tens of nodes. While the recently proposed algorithm overcomes this limitation by a space-time trade-off, our proposed algorithm realizes direct parallelization of the original DP algorithm with $O(n^\sigma)$ time and space overhead calculations, where $\sigma > 0$ controls the communication-space trade-off. The overall time and space complexity is $O(n^{\sigma+1}2^n)$. This algorithm splits the search space so that the required communication between independent calculations is minimal. Because of this advantage, our algorithm can run on distributed memory supercomputers. Through computational experiments, we confirmed that our algorithm can run in parallel using up to 256 processors with a parallelization efficiency of 0.74, compared to the original DP algorithm with a single processor. We also demonstrate optimal structure search for a 32-node network without any constraints, which is the largest network search presented in literature.

Keywords: optimal Bayesian network structure, parallel algorithm

1. Introduction

A Bayesian network represents conditional dependencies among random variables via a directed acyclic graph (DAG). Several methods can be used to construct a DAG structure from observed data, such as score-based structure search (Heckerman et al., 1995; Friedman et al., 2000; Imoto et al., 2002), statistical hypothesis testing-based structure search (Pearl, 1988), and a hybrid of these two methods (Tsamardinos et al., 2006). In this paper, we focus on a score-based learning algorithm and formalize it as a problem to search for an optimal structure that derives the maximal (or minimal) score using a score function defined on a structure with respect to an observed data set. A score function has to be decomposed as the sum of the local score functions for each node in a network. In general, posterior probability-based score functions derived from Bayesian statistics are used. The optimal score-based structure search of Bayesian networks is known to be an NP-hard

∗. Currently at Department of Computer Science, Graduate School of Information Science and Technology, The University of Tokyo. 7-3-1 Hongo, Bunkyo-ku, Tokyo 113-0033, Japan. tamada@is.s.u-tokyo.ac.jp.
†. Also at the Computational Science Research Program, RIKEN, 2-1 Hirosawa, Wako, Saitama 351-0198, Japan.

problem (Chickering et al., 1995). Several efficient dynamic programming (DP) algorithms have been proposed to solve such problems (Ott et al., 2004; Koivisto and Sood, 2004). Such algorithms have $O(n \cdot 2^n)$ time and space complexity, where n is the number of nodes in the network. When these algorithms are applied to real problems, the main bottleneck is bound to be the memory space requirement rather than the time requirement, because these algorithms need to store the intermediate optimal structures of all combinations of node subsets during DP steps. Because of this limitation, such algorithm can be applied to networks of only up to around 25 nodes in a typical desktop computer. Thus far, the maximum number of nodes treated in an optimal search without any constraints is 29 (Silander and Myllymäki, 2006). This was realized by using a 100 GB external hard disk drive as the memory space instead of using the internal memory, which is currently limited to only up to several tens of GB and in a typical desktop computer.

To overcome the above mentioned limitation, Perrier et al. (2008) proposed an algorithm to reduce the search space using structural constraints. Their algorithm searches for the optimal structure on a given predefined super-structure, which is often available in actual problems. However, it is still important to search for a globally optimal structure because Bayesian networks find a wide range of applications. As another approach to overcome the limitation, Parviainen and Koivisto (2009) proposed a space-time trade-off algorithm that can search for a globally optimal structure with less space. From empirical results for a partial sub-problem, they showed that their algorithm is computationally feasible for up to 31 nodes. They also mentioned that their algorithm can be easily parallelized with up to 2^p processors, where $p = 0, 1, \ldots, n/2$ is a parameter that is used to control the space-time trade-off. Using parallelization, they suggested that it might be possible to search larger-scale network structures using their algorithm. The time and space complexities of their algorithm are $O(n \cdot 2^n (3/2)^p)$ and $O(n \cdot 2^n (3/4)^p)$, respectively.

Of course, the memory space limitation can be overcome by simply using a computer with sufficient memory. The DP algorithm remains computationally feasible even for a 29-node network in terms of the time requirement. For such a purpose, a supercomputer with shared memory is required. Modern supercomputers can be equipped with several terabytes of memory space. Therefore, they can be used to search larger networks than the current 29-node network that requires 100 GB of memory. However, such supercomputers are typically very expensive and are not scalable in terms of the memory size and the number of processors. In contrast, massively parallel computers, a much cheaper type of supercomputers, make use of distributed memory; in such systems many independent computers or computation nodes are combined and linked through high-speed connections. This type of supercomputers is less expensive, as mentioned, and it is scalable in terms of both the memory space and the number of processors. However the DP algorithm cannot be executed on such a distributed memory computer because it requires memory access to be performed over a wide region of data, and splitting the search space across distributed processors and storing the intermediate results in the distributed memory are not trivial problems.

Here, we present a parallelized optimal Bayesian network search algorithm called *Para-OS*. The proposed algorithm is based on the OS algorithm using DP that was proposed by Ott et al. (2004). Our algorithm realizes direct parallelization of the DP steps in the original algorithm by splitting the search space of DP with $O(n^\sigma)$ time and space overhead calculations, where $\sigma = 1, 2, \ldots > 0$ is a parameter that is used to split the search space and controls the trade-off between the number of communications (not the volume of communication) and the memory space requirement. The main feature of this algorithm is that it guarantees that the amount of intermediate results that are required to be shared redundantly among independently split calculations is minimal. In other words,

our algorithm guarantees that minimal communications are required between independent parallel processors. Because of this advantage, our algorithm can be easily parallelized, and, in practice, it can run very efficiently on massively parallel computers with several hundreds of processors. Another important feature of our algorithm is that it calculates no redundant score functions, and it can distribute the entire calculation almost equally across all processors. The main operation in our algorithm is the calculation of the score function. In practice, this feature is important to actually search for the optimal structure. The overall time and space complexity is $O(n^{\sigma+1}2^n)$. Our algorithm adopts an approach opposite to that of Parviainen and Koivisto (2009) to overcome the bottleneck of the memory space problem. Although our algorithm has slightly greater space and time complexties, it makes it possible to realize large-scale optimal network search in practice using widely available low-cost supercomputers.

Through computational experiments, we show that our algorithm is applicable to large-scale optimal structure search. First, the scalability of the proposed algorithm to the number of processors is evaluated through computational experiments with simulated data. We confirmed that the program can run efficiently in parallel using up to 256 processors (CPU cores) with a parallelization efficiency of more than 0.74 on a current supercomputer system, and acceptably using up to 512 processors with a parallelization efficiency of 0.59. Finally, we demonstrate the largest optimal Bayesian network search attempted thus far on a 32-node network with 256 processors using our proposed algorithm without any constraints and without an external hard disk drive. Our algorithm was found to complete the optimal search including the score calculation within a week.

The remainder of this paper is organized as follows. Section 2 presents an overview of the Bayesian network and the optimal search algorithm, which serves as the basis for our proposed algorithm. Section 3 describes the parallel optimal search algorithm in detail. Section 4 describes the computational experiments used for evaluating our proposed algorithm and presents the obtained results. Section 5 concludes the paper with a brief discussion. The appendix contains some proofs and corollaries related to those described in the main paper.

2. Preliminaries

In this section, we first present a brief introduction to the Bayesian network model, and then, we describe the optimal search (OS) algorithm, which is the basal algorithm that we parallelize in the proposed algorithm.

2.1 Bayesian Network

A Bayesian network is a graphical model that is used to represent a joint probability of random variables. By assuming the conditional independencies among variables, the joint probability of all the variables can be represented by the simple product of the conditional probabilities. These independencies can be represented via a directed acyclic graph (DAG). In a DAG, each node corresponds to a variable and a directed edge, to the conditional dependencies among variables or to the independencies from other variables. Suppose that we have n random variables, $V = \{X_1, X_2, \ldots, X_n\}$. The joint probability of variables in V is represented as

$$P(X_1, X_2, \ldots, X_n) = \prod_{j=1}^{n} P(X_j | Pa^G(X_j)),$$

where $Pa^G(X_j)$ represents the set of variables that are direct parents of the j-th variable X_j in network structure G and $P(X_j|Pa^G(X_j))$, a conditional probability for variable X_j.

A score-based Bayesian network structure search or a Bayesian network estimation problem is to search for the DAG structure fitted to the observed data, in which the fitness of the structure to the given data is measured by a score function. The score function is defined on a node and its parent set. The scores of nodes obtained by a score function are called local scores. A network score is defined simply as the sum of local scores of all nodes in a network. Using a score function, the Bayesian network structure search can be defined as a problem to find a network structure $\hat{G}$ that satisfies the following equation:

$$\hat{G} = \arg\min_G \sum_{j=1}^{n} s(X_j, Pa^G(X_j), X),$$

where $s(X_j, Pa^G(X_j), X)$ is a score function $s : V \times 2^V \times \mathbb{R}^{N,n} \to \mathbb{R}$ for node X_j given the observed input data of an $(N \times n)$-matrix X, where N is the number of observed samples.

2.2 Optimal Search Algorithm using Dynamic Programming

Next, we briefly introduce the OS algorithm using DP proposed by Ott et al. (2004). Our proposed algorithm is a parallelized version of this algorithm. We employ score functions described as in the original paper by Ott et al. (2004). That is, a smaller score represents better fitting of the model. Therefore, the problem becomes one of finding the structure that minimizes the score function. The optimal network structure search by DP can be regarded as an optimal permutation search problem. The algorithm consists of two-layer DP: one for obtaining the optimal choice of the parent set for each node and one for obtaining the optimal permutation of nodes. First, we introduce some definitions.

Definition 1 (Optimal local score) *We define the function $F : V \times 2^V \to \mathbb{R}$ as*

$$F(v, A) \stackrel{\text{def}}{=} \min_{B \subset A} s(v, B, X).$$

That is, $F(v, A)$ calculates the optimal choice of the parent set from A for node v and returns its optimal local score. $B \subset A$ represents the actual optimal choice for v, and generally, we also need to include it in the algorithm along with the score, in order to reconstruct the network structure later.

Definition 2 (Optimal network score on a permutation) *Let $\pi : \{1, 2, \ldots, |A|\} \to A$ be a permutation on $A \subset V$ and Π^A be a set of all the permutations on A. Given a permutation $\pi \in \Pi^A$, the optimal network score on π can be described as*

$$Q^A(\pi) \stackrel{\text{def}}{=} \sum_{v \in A} F(v, \{u \in A : \pi^{-1}(u) < \pi^{-1}(v)\}).$$

Definition 3 (Optimal network score) *By using $Q^A(\pi)$ defined above, we can formalize the network structure search as a problem to find the optimal permutation that gives the minimal network score:*

$$M(A) \stackrel{\text{def}}{=} \arg\min_{\pi \in \Pi^A} Q^A(\pi).$$

Here, $M(A)$ represents the optimal permutation that derives the minimal score of the network consisting of nodes in A.

Finally, the following theorem provides an algorithm to calculate $F(v,A)$, $M(A)$, and $Q^A(M(A))$ by DP. See Ott et al. (2004) for the proof of this theorem.

Theorem 4 (Optimal network search by DP) *The functions $F(v,A)$, $M(A)$, and $Q^A(M(A))$ defined above can be respectively calculated by the following recursive formulae:*

$$F(v,A) = \min\{s(v,A,X), \min_{a\in A} F(v,A\setminus\{a\})\}, \tag{1}$$

$$M(A)(i) = \begin{cases} M(A\setminus\{v^*\})(i) & (i < |A|) \\ v^* & (i = |A|) \end{cases}, \tag{2}$$

$$Q^A(M(A)) = F(v^*, A\setminus\{v^*\}) + Q^{A\setminus\{v^*\}}(M(A\setminus\{v^*\})), \tag{3}$$

where

$$v^* = \arg\min_{v\in A}\{F(v,A\setminus\{v\}) + Q^{A\setminus\{v\}}(M(A\setminus\{v\}))\}.$$

By applying the above equations from $|A| = 0$ to $|A| = |V|$, we obtain the optimal permutation π on V and its score $Q^V(M(V))$ in $O(n\cdot 2^n)$ steps.

Note that in order to reconstruct the network structure, we need to keep the optimal choice of the parent set derived in Equation (1) and the optimal permutation $\pi = M(A)$ in Equation (3) for all the combinations of $A \subset V$ in an iterative loop for the next size of A.

3. Parallel Optimal Search Algorithm

The key to parallelizing the calculation of the optimal search algorithm by DP is splitting all the combinations of nodes in a single loop of DP for $F(v,A)$, $M(A)$, and $Q^A(M(A))$, given above by Equations (1), (2), and (3), respectively. Simultaneously, we need to consider how to reduce the amount of information that needs to be exchanged between processors. In the calculation of $M(A)$, we need to obtain all the results of $M(\cdot)$ for one-smaller subsets of A at hand, that is, $M(A\setminus\{a\})$ for all $a \in A$. Suppose that such $M(A\setminus\{a\})$'s are stored in the distributed memory space, and we have collected them for calculating $M(A)$. In order to reduce the number of communications, it would be better if we can re-use the collected results for another calculation. For example, we can calculate $M(B)$ ($|B| = |A|$) in the same processor that calculates $M(A)$ such that some of $M(B\setminus\{b\})$ ($b \in B$) *overlaps* $M(A\setminus\{a\})$. That is, if we can collect the maximal number of $M(X)$'s for any X such that $|X| = |A| - 1 \wedge X \subset \{(A\setminus\{a\}) \cap (B\setminus\{b\})\}$, then the number of communications required for calculating $M(A)$ and $M(B)$ can be minimized. Theorem 7 shows how we generate such a set of combinations, and we prove that it provides the optimally minimal choice of such combinations by allowing some redundant calculations.

In addition, as is evident from Equations (1), (2), and (3), the DP algorithm basically consists of simply searching for the best choice from the candidates that derive the minimal score. Time is mainly required to calculate the score function $s(v,A,X)$ for all the nodes and their parent combinations. Thus, our algorithm calculates $s(v,A,X)$ equally in independent processors without any redundant calculations for this part.

In this section, we first describe some basic definitions, and then, we present proofs of theorems that the proposed algorithm relies on. Finally, we present the proposed parallel algorithm.

3.1 Separation of Combinations

First, we define the combination, sub-combination, and super-combination of nodes.

Definition 5 (Combination) *In this paper, we refer to a set of nodes in V as a combination of nodes. We also assume a combination of k nodes, that is, $X = \{x_1, x_2, \ldots, x_k\} \subset V$ such that $ord(x_i) < ord(x_j)$ if $i < j$, where $ord : V \to \mathbb{N}$ is a function that returns the index of element $v \in V$. For example, suppose that $V = \{a, b, c, d\}$. $ord(a) = 1$ and $ord(d) = 4$.*

Definition 6 (Sub-/super-combination) *We define C' as a sub-combination of some combination C if $C' \subset C$, and C is a super-combination of C'. We say that a super-combination C is generated from C' if C is a super-combination of C'. In addition, we say that sub-combination C' is derived from C if C' is a sub-combination of C. For the sake of convenience, if we do not mention about the size of a sub-/super-combination of a combination, then we assume that it refers to a one-size smaller/larger sub-/super-combination. In addition, we say that two combinations A and B share sub-combinations if $\mathcal{A}' \cap \mathcal{B}' \neq \emptyset$, where $\mathcal{A}'$ and $\mathcal{B}'$ are sets of all the sub-combinations of A and B, respectively.*

We present two theorems that our algorithm relies on along with their proofs. These two theorems are used to split the calculation of $F(v, A)$, $M(A)$, and $Q^A(M(A))$; all these require the results for their sub-combinations. We show that the calculation can be split by the super-combination of A, and it is the optimal separation of the combinations in terms of the number of communications required.

Theorem 7 (Minimality of required sub-combinations) *Let $\mathcal{A}$ be a set of combinations of nodes in V, where $|A| = k > 0$ for $A \in \mathcal{A}$ and $|V| = n$. If $|\mathcal{A}| = \binom{k+\sigma}{k}$ ($\sigma > 0 \wedge \sigma + k \leq n$), then the minimal number of sub-combinations of length $k-1$ required to generate all the combinations in $\mathcal{A}$ is $\binom{k+\sigma}{k-1}$. Let S be a combination of nodes, where $|S| = k + \sigma$. We can generate a set of combinations of length k that satisfies the former condition by deriving all the sub-combinations of length k from S.*

Proof Because $\mathcal{A}$ contains $\binom{k+\sigma}{k}$ combinations of length k, the number of distinct elements (nodes) involved in $\mathcal{A}$ is $k+\sigma$ and is minimal. Therefore, the number of sub-combinations required to derive $\binom{k+\sigma}{k}$ combinations of length k in $\mathcal{A}$ is equal to the number of possible combinations that can be generated from $k+\sigma$ elements, and is $\binom{k+\sigma}{k-1}$. No more combinations can be generated from $\binom{k+\sigma}{k-1}$ sub-combinations. Therefore, it is the minimal number of required sub-combinations required to generate $\binom{k+\sigma}{k}$ combinations. A super-combination S of length $k+\sigma$ contains $k+\sigma$ elements and can derive $\binom{k+\sigma}{k}$ combinations of length k. Therefore, S can derive $\mathcal{A}$, and all the combinations of length $k-1$ derived from elements in S are the sub-combinations required to generate combinations in $\mathcal{A}$. ■

Theorem 8 (Minimality of required super-combinations) *The minimal number of super-combinations of length $k+\sigma$ from V that is required to generate all the sub-combinations of size k is $\binom{n-\sigma}{k}$.*

Proof Consider the set $T = V \setminus \{v_1, v_2, \ldots, v_\sigma\}$, where any $v_i \in V$, and thus, $|T| = n - \sigma$. If we generate all the combinations of length k taken from T, then these include all the combinations of length k from nodes in V without $v_1, \ldots, v_\sigma$, and the number of combinations is $\binom{n-\sigma}{k}$. Consider a set of combinations $\mathcal{S} = \{\{v_1, \ldots, v_\sigma\} \cup T' : T' \subset T \wedge |T'| = k\}$. Here, $|S| = k + \sigma$ for $S \in \mathcal{S}$ and $|\mathcal{S}| = \binom{n-\sigma}{k}$. Because $\mathcal{S}$ contains all the combinations of length k without $v_1, \ldots, v_\sigma$ and all the elements in $\mathcal{S}$ contain $v_1, \ldots, v_\sigma$, we can generate all the combinations in V of length k from some $S \in \mathcal{S}$ by combining $0 \leq \alpha \leq k$ nodes from $v_1, \ldots, v_\sigma$ and $k - \alpha$ nodes from $S \setminus \{v_1, \ldots, v_\sigma\}$. If we remove any $S \in \mathcal{S}$ from it, then there exist combinations that cannot be generated from another $S \in \mathcal{S}$ because $\mathcal{S}$ lists all the combinations except for nodes $v_1, \ldots, v_\sigma$. Therefore, $\mathcal{S}$ is the minimal set of super-combinations required to derive all the combinations of length k from V and its size is $|\mathcal{S}| = \binom{n-\sigma}{k}$. We can generate $\mathcal{S}$ by taking the first $\binom{n-\sigma}{k}$ combinations from $\binom{n}{k}$ combinations arranged in lexicographical order. ■

Theorem 7 can be used to split the search space of DP using by *super-combinations*, and Theorem 8 provides the number of super-combinations required in the parallel computation for a certain size of A for $M(A)$. From these two theorems, we can easily derive the following corollaries.

Corollary 9 (Optimal separation of combination) *The DP steps used to calculate $M(A)$ and $Q^A(M(A))$ in the OS algorithm can be split into $\binom{n-\sigma}{k}$ portions by super-combinations of A with length $k+\sigma$, where $|A| = k$. The size of each split problem is $\binom{k+\sigma}{k}$ and the number of required $M(B)$ for $B \subset A \wedge |B| = k-1$ is $\binom{k+\sigma}{k-1}$, which is the minimal number for $\binom{k+\sigma}{k}$ combinations of $M(A)$. Here, B is a set of sub-combinations of A. The calculation of $F(v,A)$ can also be split based on the sub-combinations B for $M(A)$.*

The separation of $M(A)$ by super-combinations causes some redundant calculations. The following corollary gives the amount of such overhead calculations and the overall complexity of the algorithm.

Corollary 10 (Amount of redundant calculations) *If we split the calculations of $M(A)$ ($|A| = k$) using super-combinations of size $k+\sigma$ ($\sigma > 0$), then the number of calculations of $M(A)$ for all $A \subset V$ is $\binom{n-\sigma}{k} \cdot \binom{k+\sigma}{k} = \binom{n}{k} O(n^\sigma)$. Thus, as compared to the original DP steps $\binom{n}{k}$, the overhead increment of the calculations for $M(A), Q^A(M(A))$, and $F(v,A)$ is at most $O(n^\sigma)$. The memory requirement to store the intermediate results is also dependent on the size of the sub-combinations for split calculations. Therefore, the overall time and space complexity of the algorithm is $O(n^{\sigma+1}2^n)$.*

We present a proof in Appendix B. The parameter $\sigma > 0$ can be used to control the size of split problems. Because using a large value of σ suppresses the number of required super-combinations, the number of communications required between independent calculations is also suppressed. Instead, the large value of σ requires a large memory space to store the sub-combinations in a processor. Therefore, σ can be used to control the trade-off between the number of communications and the memory space requirement. Because the algorithm requires the exchange of intermediate results $\binom{n-\sigma}{k}$ times for a loop with $|A| = k$ and is a relatively large number, decreasing the number of communications reduces the communication speed. In a case with many processors, however, a large value of σ can also reduce the communication speed because a large value of σ requires the transfer of a large amount of data, instead of reducing the number of communications. Table 1

	Increment for		
k	$\sigma = 1$	$\sigma = 2$	$\sigma = 3$
1	1	2	3
2	2	5	8
10	7	30	55
14	8	37	111
27	4	8	8

Table 1: Examples of the actual increment of various values of k and σ for $n = 32$. The increment is largest for all cases of σ for $k = 14$.

Algorithm 1 *Process-S*(S,a,n,n_p) calculates the functions $F(v,A)$, $M(A)$, and $Q^A(M(A))$ for combinations A derived from the given super-combination S.

Input: $S \subset V$: Super-combination, $a \in \mathbb{N}$: size of combination to be calculated, n: total number of nodes in the network, n_p: number of CPU processors (cores).

Output: $F(v,B)$, $Q(A)$, and $M(Q(A))$ for all sub-combinations of S with size a, $v \in A$, and $B = A \setminus \{v\}$.

1: $\mathcal{A} \leftarrow \{A \subset S : |A| = a\}$
2: Retrieve the local scores $s(v,B,X)$ for $B = A \setminus \{v\}$ $(v \in A \in \mathcal{A})$ from the $L^F(v,B,n,n_p)$-th processor.
3: Retrieve $F(u,B \setminus \{u\})$ for $u \in B$ $(B = A \setminus \{v\}, v \in A \in \mathcal{A})$ from the $L^F(u,B \setminus \{u\},n,n_p)$-th processor.
4: Retrieve $Q^{A\setminus\{v\}}(M(A \setminus \{v\}))$ for $v \in A \in \mathcal{A}$ from the $L^Q(A \setminus \{v\},n,n_p)$-th processor.
5: **for** each $A \in \mathcal{A}$ **do**
6: Calculate $F(v,B)$ for $v \in A, B = A \setminus \{v\}$ from $s(v,B)$ and $F(v,B \setminus \{u\})$ for $u \in B$ by Equation (1).
7: Calculate $M(A)$ and $Q^A(M(A))$ from $Q^{A\setminus\{v\}}(M(A \setminus \{v\}))$ and $F(v,A \setminus \{v\})$ by Equations (3) and (2).
8: **end for**
9: Store $Q(A)$ and $M^A(Q(A))$ $(A \in \mathcal{A})$ in the $L^Q(A,n,n_p)$-th processor.
10: Store $F(v,B)$ $(B = A \setminus \{v\}, A \in \mathcal{A})$ in the $L^F(v,B,n,n_p)$-th processor.

shows some examples of the actual overhead increment of the DP steps, that is, $\binom{n-\sigma}{k} \cdot \binom{k+\sigma}{k} / \binom{n}{k}$. As shown in the table, the increment because of redundant DP steps caused by the separation appears to be relatively small for a case of the practical size of n and σ. If the algorithm runs in parallel with hundreds of processors, the increment calculation in each processor is negligible as compared to the total amount of calculations, and thus, it does not noticeably affect the overall computation time. We discuss this later with the computational experiments presented in Section 4.2.

Algorithm 2 *Para-OS*(V, X, s, σ, n_p) calculates the exactly global optimal structure of the Bayesian network with respect to the input data X and the local score function s with n_p processors.

Input: V: set of input nodes (variables) where $|V| = n$, X: $(N \times n)$-input data matrix, $s(v, Pa, X)$: function $V \times 2^V \times \mathbb{R}^{N \times n} \to \mathbb{R}$ that returns the local Bayesian network score for variable v with its parent set $Pa \subset V$ w.r.t. the input data matrix X, $\sigma \in \mathbb{N}$: size of super-combination, n_p: number of CPU processors (cores).

Output: $G = (V, E)$: optimal Bayesian network structure.

1: {Initialization}
2: Calculate $F(v, \emptyset) = s(v, \emptyset, X)$ for all $v \in V$ and store it in the $L^F(v, \emptyset, n, n_p)$-th processor.
3: Store $F(v, \emptyset)$ as $Q^{\{v\}}(M(\{v\}))$ and $M(\{v\})(1) = v$ for all $v \in V$ in the $L^Q(\{v\}, n, n_p)$-th processor.
4: {Main Loop for size of A}
5: **for** $a = 1$ to $n - 1$ **do**
6: {S-phase: Execute the following for-loop on i in parallel. The $\{r = i \bmod n_p + 1\}$-th processor is responsible for the $(i+1)$-th loop.}
7: **for** $i = 0$ to $n\binom{n-1}{a} - 1$ **do**
8: $v \leftarrow i \bmod n + 1$
9: $j \leftarrow \lfloor i / n \rfloor + 1$
10: $Pa \leftarrow m(v, RLI^{-1}(j, n-1, a))$
11: Calculate $s(v, Pa, X)$ and store it in the local memory of the r-th processor.
12: **end for**
13: {Q-phase: Execute the following for-loop on i in parallel. The $\{r = i \bmod n_p + 1\}$-th processor is responsible for the $(i+1)$-th loop.}
14: **if** $a + \sigma + 1 > n$ **then**
15: $\sigma \leftarrow n - a - 1$.
16: **end if**
17: **for** $i = \binom{n}{a+\sigma+1} - \binom{n-\sigma}{a+1}$ to $\binom{n}{a+\sigma+1} - 1$ **do**
18: $S \leftarrow RLI^{-1}(i+1, n, a+\sigma+1)$.
19: Call *Process-S*$(S, a+1, n, n_p)$.
20: **end for**
21: **end for**
22: Construct network $G = (V, E)$ by collecting the final sets of the parents selected in line 6 of *Process-S*$(\cdot)$.
23: **return** $G = (V, E)$.

3.2 Para-OS Algorithm

According to Theorems 7 and 8 and Corollary 9, the DP steps in Equations (1), (2), and (3) of Theorem 4 can be split by super-combinations of A. The pseudocode of the proposed algorithm is given by Algorithms 1 and 2. The former is a sub-routine of the latter main algorithm.

The algorithm consists of two phases: the S-phase and the Q-phase. In the former, each processor calculates the score function $s(v, Pa, X)$ independently without communication, whereas the latter calculates $F(v, A)$, $M(A)$, and $Q^A(M(A))$ along with communications among each other to exchange the results of $F(v, A)$, $M(A)$, and $Q^A(M(A))$. Note that in line 6 of Algorithm 1, we need

to store not only the local scores but also the optimal choices of parent node sets, although we do not describe this explicitly. This is required to reconstruct the optimal structure after the algorithm terminates.

In this algorithm, we need to determine which processor stores the calculated intermediate results. In order to calculate this, we define some functions as given below.

Definition 11 *We define function* $m' : \mathbb{N} \times \mathbb{N} \to \mathbb{N}$ *as follows:*

$$m'(a,b) = \begin{cases} a & \text{if } a < b \\ a+1 & \text{otherwise} \end{cases}.$$

Using $m'(a,b)$*, we define function* $m : V \times 2^V \to 2^V$ *as follows:*

$$m(v,A) = \{ord^{-1}(m'(ord(u),ord(v))) : u \in A\}.$$

In addition, we define function $m'^{-1} : \mathbb{N} \times \mathbb{N} \to \mathbb{N}$ *as follows:*

$$m'^{-1}(a,b) = \begin{cases} a & \text{if } a < b \\ a-1 & \text{otherwise} \end{cases}.$$

Using $m'^{-1}(a,b)$*, we define function* $m^{-1} : V \times 2^V \to 2^V$ *as follows:*

$$m^{-1}(v,A) = \{ord^{-1}(m'^{-1}(ord(u),ord(v))) : u \in A\}.$$

The function $m(v,A)$ maps the combination A to a new combination in $V \setminus \{v\}$, and $m^{-1}(v,A)$ is the inverse function of $m(v,A)$. These are used in the proposed algorithm and the following function.

Definition 12 (Calculation of processor index to store and retrieve results) *We define functions* $L^Q : 2^V \times \mathbb{N} \times \mathbb{N} \to \mathbb{N}$ *and* $L^F : V \times 2^V \times \mathbb{N} \times \mathbb{N} \to \mathbb{N}$ *as follows:*

$$L^Q(A,n,n_p) \stackrel{\text{def}}{=} (RLI(A,n)-1) \bmod n_p + 1$$

and

$$L^F(v,A,n,n_p) \stackrel{\text{def}}{=} \left\{(RLI(m^{-1}(v,A),n-1)-1) \times n + (ord(v)-1)\right\} \bmod n_p + 1,$$

where $RLI(A,n)$ *is a function used to calculate the reverse lexicographical index (RLI) of combination* A *taken from* n *objects and* n_p*, the number of processors.*

Function $L^Q(A,n,n_p)$ locates the processor index used to store the results of $M(A)$ and $Q^A(M(A))$ and $L^F(v,A,n,n_p)$, the results of $F(v,A)$. By using RLIs, the algorithm can independently and discontinuously generate the required combinations and processor indices for storing/retrieving of intermediate results. We use RLIs instead of ordinal lexicographical indices because the conversion between a combination and the RLI can be calculated in linear time by preparing the index table once (Tamada et al., 2011). In Algorithm 2, the inverse function $RLI^{-1}(\cdot)$ is also used to reconstruct a combination from the index. See Appendix D for details of these calculations.

Figure 1 shows an example of the calculation of the DP for the super-combination S in a single processor in a single loop. Note that in the figure, although a super-combination is assigned to a single processor, $s(v,A,X)$ is calculated in a different processor from one that calculates $F(v,A)$ for the same $v \in V$ and $A \subset V \setminus \{v\}$.

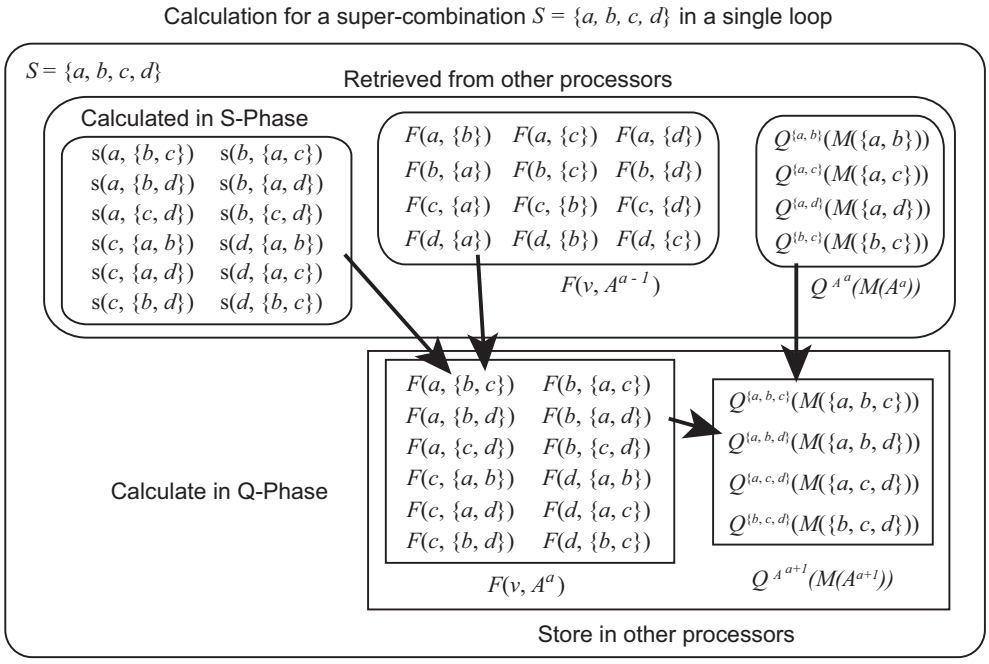


Figure 1: Schematic illustration of the calculation in a single loop on i for $a = 2$. A^a represents a subset $A \subset V$ where $|A| = a$.

4. Computational Experiments

In this section, we present computational experiments for evaluating the proposed algorithm. In the experiments, we first compared the running times and memory requirement for various values of σ. Next, we evaluated the running times of the original dynamic programming algorithm with a single processor and the proposed algorithm using 8 through 1024 processors. We also compared the results for different sizes of networks. In the experiments, we measured the running times using the continuous model score function BNRC proposed by Imoto et al. (2002). Finally, we tried to run the algorithm with as many nodes as possible on our supercomputers, as a proof of long-run practical execution that realizes the optimal large network structure learning. For this experiment, we used the discrete model score function BDe proposed by Heckerman et al. (1995), in addition to the BNRC score function. Brief definitions of BNRC and BDe are given in Appendix A. Before presenting the experimental results, we first describe the implementation of the algorithm and the computational environments used to execute the implemented programs.

4.1 Implementation and Computational Environment

We have implemented the proposed algorithm using the C programming language (ISO C99). The matrix computation in the BNRC score function is implemented using the BLAS/LAPACK library. The parallelization is implemented using MPI-1.1.

We have used two different supercomputer systems, RIKEN RICC and Human Genome Center Supercomputer System. The former is a massively parallel computer where each computation node has dual Intel Xeon 5570 (2.93 GHz) CPUs (8 CPU cores per node) and 12 GiB memory. The computation nodes are linked by X4 DDR InfiniBand. RICC employs Fujitsu's ParallelNavi that provides an MPI implementation, C compiler, BLAS/LAPACK library, and job scheduling. The latter system is similar to the former except that it has dual Intel Xeon 5450 (3 GHz) CPUs and 32 GiB memory per node. It employs OpenMPI 1.4 with Sun Grid Engine as a parallel computation environment. The compiler and the BLAS/LAPACK library are the Intel C compiler and Intel MKL, respectively.

In our implementation, each core in a CPU is treated equally as a single processor so that one MPI process runs in a single core. Therefore, 8 processes run in a computation node in both the systems. The memory in a single node is divided equally among these 8 processes.

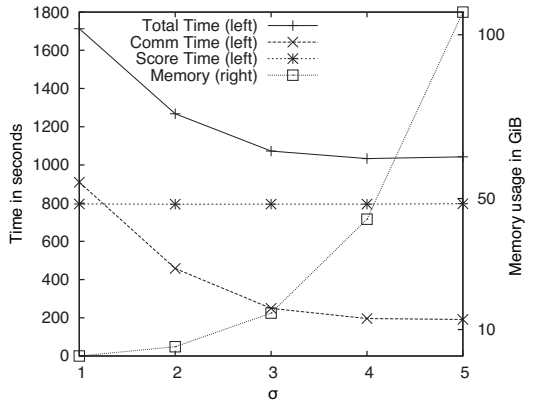
For the comparison presented later and the verification of the implementation, we also implemented the original *OS* algorithm proposed by Ott et al. (2004). The verification of the implementation was tested by comparing the optimal structures calculated by the implementations of both the original algorithm and the proposed algorithm for up to $p = 23$ using artificial simulated data with various numbers of processors. We also checked whether the greedy hill-climbing (HC) algorithm (Imoto et al., 2002) could search for a network structure having a better score than that of the optimal structure. We repeated the execution of the HC algorithm 10,000 times, and confirmed that no result was better than the optimal structure obtained using our algorithm.

4.2 Results

First, we generated artificial data with $N = 50$ (sample size) for the randomly generated DAG structure with $n = 23$ (node size). Refer to Appendix C for details on the generation of the artificial network and data. We used $n = 23$ because RICC has a limited running time of 72 hours. The calculation with a single processor for $n = 24$ exceeds this limit. In all the experiments, we carried out three measurements for each setting and took the average of these measured times as an observation for that setting. The total running times are measured for the entire execution of the program, including the input of the data from a file, output of the network to a file, and MPI initialization and finalization routine calls.

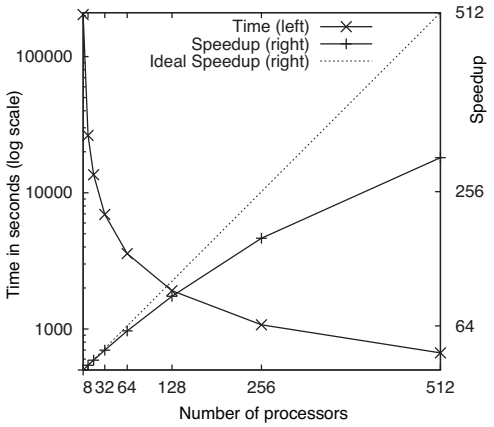
Figure 2 shows the result of the comparison of $\sigma = 1, \ldots, 5$ for $n = 23$. The row for $\sigma = 0$ shows the results of the original DP algorithm with a single processor. We measured the running times using 256 processors here. During the computation, we also measured the times required for calling MPI functions to exchange required data between processors, and the times required for calculating the score funtion $s(\cdot)$. As discussed in Section 3.1, σ controls the space-communication trade-off. We expected that an increase in the value of σ would reduce the time and increase the memory requirement. As expected, the total time decreased for up to $\sigma = 4$ with an increase in σ; however, it increased for $\sigma = 5$ and the memory requirement also increased significantly. As shown in the figure, σ does not affect the score calculation time. From these results, we employed $\sigma = 3$ for later experiments because the increase in the memory requirement and the decrease in the total time appeared reasonable.

Next, we compared the running times for various numbers of processors. We carried out measurements for $n_p = 8, 16, 32, 64, 128, 256, 512$, and 1024 processors using 1, 2, 4, 8, 16, 32, 64, and 128 computation nodes, respectively, on RICC. Here, n_p represents the number of processors.



σ	Total Time	Cm Time	Sc Time	Mem
0	-	-	-	1.27
1	1713.08	909.36	795.61	1.92
2	1267.98	458.32	794.27	4.75
3	1072.85	249.06	795.01	15.01
4	1033.20	196.04	794.75	43.69
5	1042.21	191.13	797.00	106.91

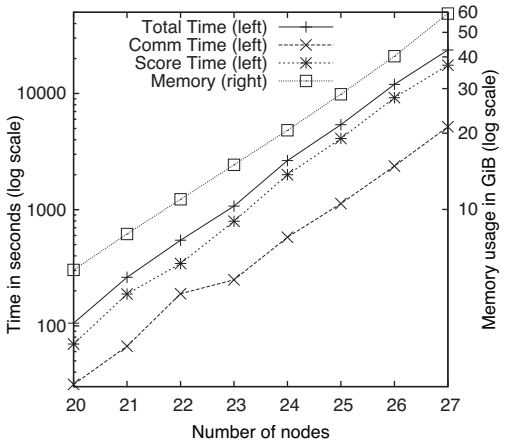
Figure 2: Running times and memory requirements with $\sigma = 1,\ldots,5$ for $n = 23$ and $N = 50$ with 256 processors. "Total Time" represents the total time required for execution in seconds, "Cm Time" represents the total communication time required for calling MPI functions within the total time; "Sc Time," the time required for score calculation; and "Mem," the memory requirement in GiB.



n_p	$T(n_p)$	$S(n_p)$	$E(n_p)$	$T_c(n_p)$	$R_c(n_p)$
1	203297.04	1.00	1.00	0.00	0.00
8	26367.58	7.71	0.96	145.24	0.01
16	13563.89	14.99	0.94	490.24	0.04
32	6932.09	29.33	0.92	398.29	0.06
64	3574.89	56.87	0.89	302.46	0.08
128	1909.72	106.45	0.83	269.90	0.14
256	1072.85	189.49	0.74	249.06	0.23
512	667.38	304.62	0.59	251.46	0.38
1024	515.29	394.53	0.39	305.12	0.59

Figure 3: Scalability test results for $n = 23$ and $N = 50$ with $\sigma = 3$. We did not present the result for $n_p = 1024$ in the graph on the left-hand side because the speedup was too low.

As mentioned above, we used $\sigma = 3$. For $n_p = 1$, we used the implementation of the original DP algorithm. Therefore, we do not use the super-combination-based separation of our proposed algorithm although it works for $n_p = 1$. Figure 3 shows the experimental result. We evaluated the parallelization scalability of the proposed algorithm from the speedup $S(n_p)$ and efficiency $E(n_p)$. The speedup $S(n_p)$ is defined as $S(n_p) = T(1)/T(n_p)$, where n_p is the number of processors and $T(n_p)$, the running time with n_p processors. If $S(n_p) = n_p$, then it is called the ideal speedup where n_p-hold speedup is obtained by n_p processors. The parallelization efficiency $E(n_p)$ is defined as $E(n_p) = S(n_p)/n_p$. In the case of ideal speedup, $E(n_p) = 1$ for any n_p. Generally, parallel programs



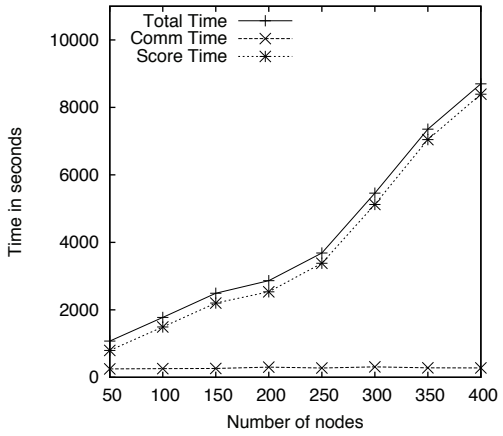
n	Total	Cm Time	(ratio)	Sc Time	Mem
20	105.54	31.35	0.30	69.71	5.77
21	261.84	66.77	0.26	187.91	8.00
22	546.07	188.80	0.35	344.00	10.97
23	1072.85	249.06	0.23	795.01	15.01
24	2645.04	579.61	0.22	1999.56	20.55
25	5386.38	1127.62	0.21	4095.74	28.50
26	11976.90	2374.81	0.20	9199.75	40.25
27	23686.51	5174.94	0.22	17517.04	59.31

Figure 4: Comparison of running tims for various network sizes. Column n represents the size of the network and "(ratio)," the ratio of "Cm Time" to "Total." "Mem" is represented in GiB. Other columns have the same meaning as in Figure 2.

that have $E(n_p) \geq 0.5$ are considered to be successfully parallelized. As shown in the table in Figure 3, the efficiencies are 0.74 and 0.59 for $n_p = 256$ and 512, respectively. However, with 1024 processors, the efficiency became 0.39 and the speedup was very low as compared to that with 512 processors, and therefore, it is not efficient and feasible. From these results, we can conclude that the program can run very efficiently in parallel for up to 256 processors, and acceptably for up to 512 processors.

$T_c(n_p)$ in Figure 3 represents the time required for calling MPI functions during the executions, and $R_c(n_p)$ is a ratio of $T_c(n_p)$ to the total time $T(n_p)$. Except for $n_p = 8$, $T_c(n_p)$ decreases with an increase in n_p because the amount of communication for which each processor is responsible decreases. However, it did not decrease linearly; in fact, for $n_p \geq 512$, it increased. This may indicate the current limitation of both our algorithm and the computer used to carry out this experiment. For $n_p = 8$, $T_c(n_p)$ was very small. This is mainly because communication between computation nodes was not required for this number of processors. If we subtract $T_c(n_p)$ from $T(n_p)$, then the efficiency $E(n_p)$ becomes 0.96, 0.95, and 0.94 for 256, 512, and 1024 processors, respectively. This result suggests that the redundant calculation in our proposed algorithm does not have a great effect, and the communication cost is the main cause of the inefficiency of our algorithm. Therefore, improving the communication speed in the future may significantly improve the efficiency of the algorithm with a larger number of processors.

Next, we compared the running times for various network sizes. We generated artificial simulated data for $n = 20$ to 27 as we did for the above experiment with $n = 23$. We measured the running times with 256 processors and $\sigma = 3$. Figure 4 shows the result. As shown in the figure, both the time and the space required increased exponentially. Note that both the left- and the right-hand side y-axes are in log scale. The communication time decreased slightly for up to $n = 26$ with an increase in n. However, for $n = 27$, it started to increase. From these results, we can say that the score calculation remains dominant and the communication does not contribute significantly to the total running times for this range of n with $n_p = 256$.



N	Total	Cm Time	Sc Time
50	1072.85	249.06	795.01
100	1775.73	255.55	1492.78
150	2490.77	261.01	2201.52
200	2863.91	302.04	2533.35
250	3684.08	276.14	3379.34
300	5460.16	309.71	5122.12
350	7354.91	281.40	7045.05
400	8698.11	276.68	8392.88

Figure 5: Comparison of running times for various sample sizes. Column N represents the number of samples. Other columns have the same meaning as in Figure 2.

To check the scalability of the algorithm to the sample size, we compared the running times for various sample sizes. We generated artificial simulated data with $N = 50, 100, 150, 200, 250, 300, 350,$ and 400 for the artificial network of $n = 23$, which is used for the previous analyses. Theoretically, the sample size does not affect the running times, except for the score calculation. Figure 5 shows the result. We confirmed that the communication times are almost constant for all the tested sample sizes and that the score calculation increased with the sample sizes, as was expected. Note that the calculation of the BNRC score function is not in linear time, and it is difficult to determine the exact time complexity because it involves an iterative optimization step (Imoto et al., 2002).

4.3 Structure Search for Large Networks

Finally, we tried to search for the optimal structure of nodes with as many nodes as possible in the HGC system because it allows long execution for up to two weeks with 256 processor cores and has a larger memory in each computation node.

As in the above experiment, we first generated random DAGs having various numbers of nodes, and then generated simulated data with 50 samples. With the BNRC score function, we have succeeded in searching for the optimal structure of a 31-node network by using 464.3 GiB memory in total (1.8 GiB per process) with 256 CPU cores in 32 computation nodes. For this calculation, we did not impose a restriction on the parent size or any other parameter restrictions. 8 days 6 hours 50 minutes 24 seconds were required to finish the calculation. The total time required for calling MPI functions was 2 days 15 hours 11 minutes 58 seconds. This is 32% of the total running time. Therefore, the communication time became a relatively large portion of the total computation time, relative to that in the case of $n = 27$ presented above. As described in Parviainen and Koivisto (2009), thus far, the largest network search that has been reported was for a 29-node network (Silander and Myllymäki, 2006). Therefore, our result improved upon this result without even using an external hard disk drive.

To search for the optimal structure of an even larger network, we used the BDe network score, which is a discrete model that is much faster than the BNRC score. Generally, the BDe score can be

calculated 100 times faster than the BNRC score (data not shown). Using the BDe score function, we successfully carried out optimal structure search for a 32-node network without any restriction using 836.1 GiB memory (3.3 GiB per process) in total with 256 CPU cores. The total computation time was 5 days 14 hours 24 minutes and 34 seconds. The MPI communication time was 4 days 12 hours 56 minutes 26 seconds, and this is 81% of the total time. Thus, for $n = 32$ with the BDe score function, the calculation of score functions requires relatively very little time (actually, it required only around 1.5 hours per process) as compared to the total time, and the communication cost becomes the dominant part and the bottleneck of the calculation.

These results show that our algorithm is applicable to the optimal structure search of relatively large-sized networks and it can be run on modern low-cost supercomputers.

5. Discussions

In this paper, we have presented a parallel algorithm to search for the score-based optimal structure of Bayesian networks. The main feature of our algorithm is that it can run very efficiently on massively parallel computers in parallel. We confirmed the scalability of the algorithm to the number of processors through computational experiments and successfully demonstrated optimal structure search for a 32-node network with 256 processors, an improvement over the most successful result reported thus far. Our algorithm overcomes the bottleneck of the previous algorithm by using a large amount of distributed memory for large-scale Bayesian network structure search.

Our algorithm has a feature similar to that of an algorithm recently proposed by Parviainen and Koivisto (2009) that requires less space. Both algorithms divide the search space of the problem, and provide a way to compute the optimal structure in parallel. Both are capable of breaking the current limitation of the network size in optimal network structure search. However, these two algorithms differ in several respects. First, Parviainen and Koivisto (2009) primarily intended to develop a space-time trade-off algorithm to overcome the bottleneck of the search problem. They found that the search problem can be divided into sub-problems and that these sub-problems can be solved independently with less space. Therefore, although the time requirement increases with a decrease in the space requirement, they mentioned that their algorithm can obtain the optimal structure for a 34-node network by massive parallelization. Our algorithm, on the other hand, overcomes the bottleneck in a more straightforward way. We found a way to divide the DP steps of the fastest known algorithm with a relatively low overhead cost. In terms of memory requirement, our algorithm consumes much more memory space than that of Parviainen and Koivisto (2009) and even more than the original DP algorithm to realize parallelization. However, our algorithm can actually search for the optimal structure of a 32-node network with 256 processors in less than a week, including score calculation, whereas Parviainen and Koivisto (2009) computed only the partial problems. Their estimate from their empirical result is 4 weeks using 100 processors to obtain the optimal structure for a 31-node network. In addition, their estimate ignores the parallelization overhead that generally becomes problematic in parallelization as well as score calculation, which requires the most time in the actual application. We showed that our algorithm works efficiently with up to 256 processors, and acceptably with up to 512 processors. An optimal search for even larger networks may be realized by improving the current implementation. Our implementation regards each processor core in a CPU equivalently. Therefore, exploiting the modern multi-core CPUs can reduce the communications required among computation nodes and increase the amount of memory space for independent calculations without requiring improved hardware relative to current supercomputers.

Acknowledgments

Computational time was provided by the Supercomputer System, Human Genome Center, Institute of Medical Science, The University of Tokyo; and RIKEN Supercomputer system, RICC. This research was supported by a Grant-in-Aid for Research and Development Project of the Next Generation Integrated Simulation of Living Matter at RIKEN and by MEXT, Japan. The authors would like to thank the anonymous referees for their valuable comments.

Appendix A. Definitions of Score Functions

In this paper, we use BNRC (Imoto et al., 2002) and BDe (Heckerman et al., 1995) as a score function $s(\cdot)$ for computational experiments of the proposed algorithm. Here, we present brief definitions of these score functions.

A.1 The BNRC Score Function

BNRC is a score function for modeling continuous variables. In a continuous model, we consider the joint density of the variables instead of their joint probability. We search the network structure G by maximizing the posterior of G for the input data matrix X. The posterior of G is given by

$$\pi(G|X) = \pi(G) \int \prod_{i=1}^{N} \prod_{j=1}^{n} f(x_{ij}|pa_{ij}^{G};\theta_j)\pi(\theta_G|\lambda)d\theta_G,$$

where $\pi(G)$ is the prior distribution of G; $f(x_{ij}|pa_{ij}^{G};\theta_j)$, the local conditional density for the j-th variable; $pa_{ij}^{G} = (pa_{i1}^{(j)},\ldots,pa_{iq_j}^{(j)})$, the set of observations in the i-th sample of q_j variables that represents the direct parents of the j-th node in a network; $\theta_G = (\theta_1,\ldots,\theta_n)$, the parameter vector of the conditional densities to be estimated; and $\pi(\theta_G|\lambda)$, the prior distribution of θ_G specified by the hyperparameter λ. Conditional density $f(x_{ij}|pa_{ij}^{G};\theta_j)$ is modeled by nonparametric regression with B-spline basis functions given by

$$f(x_{ij}|pa_{ij}^{G};\theta_j) = \frac{1}{\sqrt{2\pi\sigma_j^2}} \exp\left[-\frac{\{x_{ij} - \sum_{k=1}^{q_j} m_{jk}(pa_{ik}^{(j)})\}^2}{2\sigma_j^2}\right],$$

where $m_{jk}(p_{ik}^{(j)}) = \sum_{l=1}^{M_{jk}} \gamma_{lk}^{(j)} b_{lk}^{(j)}(p_{ik}^{(j)})$, $\{b_{1k}^{(j)}(\cdot),\ldots,b_{M_{jk},k}^{(j)}(\cdot)\}$ is the prescribed set of M_{jk} B-splines; σ_j, the variance, and $\gamma_{lk}^{(j)}$, the coefficient parameters. By taking a $-2\log$ of the posterior, the BNRC score function for the j-th node is defined as

$$s_{BNRC}(X_j, Pa^G(X_j), X) = -2\log\left\{\pi_j^G \int \prod_{i=1}^{N} f(x_{ij}|pa_{ij}^{G};\theta_j)\pi_j(\theta_j|\lambda_j)d\theta_j\right\},$$

where π_J^G is the prior distribution of the local structure associated with the j-th node; and $\pi_j(\theta_j|\lambda_j)$, the decomposed prior distribution of θ_j specified by the hyperparamter λ_j.

A.2 The BDe Score Function

BDe, a score function for the discrete model, can be applied to discrete (categorical) data. As in the case of BNRC, BDe also considers the posterior of G; that is,

$$P(G|X) \propto P(G) \int P(X|G,\theta)P(\theta|G)d\theta,$$

where $P(X|G,\theta)$ is the product of local conditional probabilities (likelihood of X given G) and $P(\theta|G)$, the prior distribution for parameters θ. In the discrete model, we employ multinomial distribution for modeling the conditional probability and the Dirichlet distribution as its prior distribution. Let X_j be a discrete random variable corresponding to the j-th node, which takes one of r values $\{u_1,\ldots,u_r\}$, where r is the number of categories of X_j. In this model, the conditional probability of X_j is parameterized as

$$P(X_j = u_k | Pa^G(X_j) = u_{jl}) = \theta_{jlk},$$

where u_{jl} $(l = 1,\ldots,r^{q_j})$ is a combination of values for the parents and q_j, the number of parents of the j-th node. Note that $\sum_{k=1}^{r} \theta_{jlk} = 1$. For the discrete model, the likelihood can be expressed as

$$P(X|G,\theta) = \prod_{j=1}^{n} \prod_{l=1}^{r^{q_j}} \prod_{k=1}^{r} \theta_{jlk}^{N_{jlk}},$$

where N_{jlk} is the number of observations for the j-th node whose values equal u_k in the data matrix X with respect to a combination of the parents' observation l. $N_{jl} = \sum_{k=1}^{r} N_{jlk}$ and θ denotes a set of parameters θ_{jlk}. For the parameter set θ, we assume the Dirichlet distribution as $\pi(\theta|G)$; then, the marginal likelihood can be described as

$$\int P(X|G,\theta)P(\theta|G)d\theta = \prod_{j=1}^{n} \prod_{l=1}^{r^{q_j}} \frac{\Gamma(\alpha_{jl})}{\Gamma(\alpha_{jl}+N_{jl})} \prod_{k=1}^{r} \frac{\Gamma(\alpha_{jlk}+N_{jlk})}{\Gamma(\alpha_{jlk})},$$

where θ is a set of parameters; α_{jlk}, a hyperparameter for the Dirichlet distribution; and $\alpha_{jl} = \sum_{k=1}^{r} \alpha_{jlk}$. By taking $-\log$ of the posterior, the BDe score function for the j-th node is defined as

$$s_{BDe}(X_j, Pa^G(X_j), X) = -\log \pi_j^G - \log \left\{ \prod_{l=1}^{r^{q_j}} \frac{\Gamma(\alpha_{jl})}{\Gamma(\alpha_{jl}+N_{jl})} \prod_{k=1}^{r} \frac{\Gamma(\alpha_{jlk}+N_{jlk})}{\Gamma(\alpha_{jlk})} \right\},$$

where π_j^G is the prior probability of the local structure associated with the j-th node.

Appendix B. Proof of Corollary 10

Proof We prove that $\binom{n-\sigma}{k} \cdot \binom{k+\sigma}{k} = \binom{n}{k} O(n^\sigma)$.

$$\begin{aligned} \binom{n-\sigma}{k} \cdot \binom{k+\sigma}{k} &= \frac{(n-\sigma)!}{k!(n-\sigma-k)!} \cdot \frac{(k+\sigma)!}{k!(k+\sigma-k)!} \\ &= \frac{(n-\sigma)!}{k!(n-\sigma-k)!} \cdot \frac{(k+\sigma)!}{k!\sigma!} \cdot \frac{(n-k)!}{(n-k)!} \cdot \frac{n!}{n!} \end{aligned}$$

$$
\begin{aligned}
&= \frac{n!}{k!(n-k)!} \cdot \frac{(n-\sigma)!(k+\sigma)!(n-k)!}{(n-\sigma-k)!k!\sigma!n!} \\
&= \binom{n}{k} \cdot \frac{(n-\sigma)!}{(n-\sigma-k)!} \cdot \frac{(k+\sigma)!}{k!\sigma!} \cdot \frac{(n-k)!}{n!} \\
&= \binom{n}{k} \cdot \frac{(n-\sigma)}{n} \frac{(n-\sigma-1)}{(n-1)} \cdots \frac{(n-\sigma-k+1)}{(n-k+1)} \cdot \frac{(k+\sigma)\cdots(k+1)k!}{\sigma!k!} \\
&= \binom{n}{k} \cdot O(1) \cdot O((k+\sigma)^{\sigma}) = \binom{n}{k} \cdot O((k+\sigma)^{\sigma}).
\end{aligned}
$$

For $n < k+\sigma$, we consider only super-combinations of size n. Thus, for each k, there are at most $\binom{n}{k} O(n^{\sigma})$ calculations. ■

Appendix C. Method for Generating Artificial Network and Data

To generate the random DAG structure, we simply added edges at random to an empty graph so that the acyclic structure is maintained and the average degree becomes $d = 4.0$. Consequently, the number of total edges equals $n \cdot d/2$. Note that the degree of DAGs does not affect the execution time of the algorithm as the algorithm searches all possible structures. To generate the artificial data, we first randomly assigned 8 different nonlinear or linear equations to the edges and then generated the artificial numerical values based on the normal distribution and the assigned equations. Figure 6 shows the assigned equations and examples of the generated data. If a node has more than two parent nodes, the generated values are summed before adding the noise. We set the noise ratio to be 0.2.

To apply BDe to the artificial data on the $n = 32$ network, we discretized the continuous data generated by the same method for all variables into three categories ($r = 3$) and then executed the algorithm for these discretized values.

Appendix D. Efficient Indexing of Combinations

When running the algorithm, we need to generate combination vectors discontinuously and independently in a processor. To do this efficiently, we require an algorithm that calculates a combination vector from its index and the index from its combination vector. Buckles and Lybanon (1977) presented an efficient lexicographical index - vector conversion algorithm. However, this algorithm requires the calculation of binomial coefficients for every possible element in a combination every time. To speed up this calculation, we developed a linear time algorithm that needed polynomial time to construct a reusable table (Tamada et al., 2011). Our algorithm actually deals with *the reverse lexicographical index* instead of the ordinal lexicographical index; this enables us to calculate the table only once and to make it reusable. In this section, we present the algorithms as described in Tamada et al. (2011). See Tamada et al. (2011) for details and the proofs of the theorems. In this section, note that we assume that $\sum_{i=k}^{n} f_i = 0$ for any f_i if $n < k$.

Theorem 13 (RLI calculation) *Let $\mathcal{C} = \{C_1, C_2, \ldots, C_m\}$ be the set of all the combinations of length k taken from n objects, arranged in lexicographical order, where $m = \binom{n}{k}$. We call i the lexicographical index of $C_i \in \mathcal{C}$. Let us define the reverse lexicographical index of $C_i \in \mathcal{C}$, RLI(C_i, n)*

(a) $f_a(X) = X + \varepsilon$

(b) $f_b(X) = -X + \varepsilon$

(c) $f_c(X) = \frac{1}{1+e^{-4X}} + \varepsilon$

(d) $f_d(X) = -\frac{1}{1+e^{-4X}} + \varepsilon$

(e) $f_e(X) = -(X-1.5)^2 + \varepsilon$

(f) $f_f(X) = (-X-2.5)^2 + \varepsilon$

(g) $f_g(X) = -(-X-2.5)^2 + \varepsilon$

(h) $f_h(X) = (X-1.5)^2 + \varepsilon$

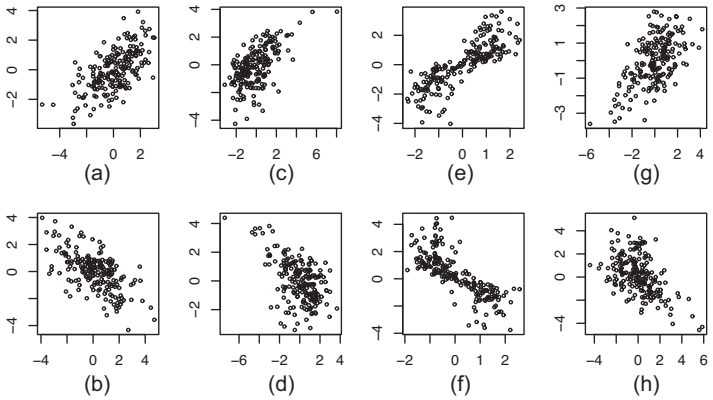


Figure 6: Left: Linear and nonlinear equations assigned to each edge in artificial networks. ε represents the noise based on the normal distribution. Right: Examples of the values generated by these equations.

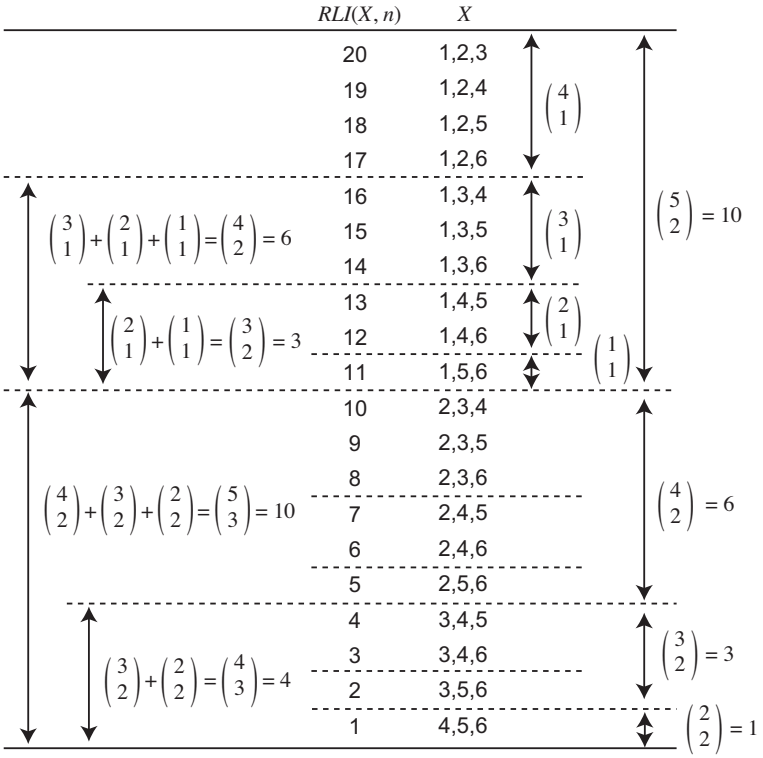


Figure 7: RLI calculation for $n = 6$ and $k = 3$.

$\stackrel{\text{def}}{=} m - i + 1$. *Suppose that we consider a combination of natural numbers, that is, some combination* $X = \{x_1, x_2, \ldots, x_k\} \in C$, *where* $x_i < x_j$ *if* $i < j$ *and* $x_i \in \{1, 2, \ldots, n\}$. *Then,* $RLI(X, n)$ *can be*

Algorithm 3 *RLITable*(m) generates the index table for conversion between a combination and the index.

Input: m: maximum number of elements appearing in a combination,
Output: $T : m \times m$ index table.

{Initialization}
$T(i,1) \leftarrow 0 \quad (1 \leq i \leq m)$.
$T(1,j) \leftarrow j-1 \quad (2 \leq j \leq m)$.
{Main Routine}
for $i = 2$ to m **do**
  for $j = 2$ to m **do**
    $T(i,j) \leftarrow T(i-1,j) + T(i,j-1)$.
  end for
end for
return T.

Algorithm 4 *RLI*(X,n,T) calculates the reverse lexicographical index of the given combination X.

Input: $X = \{x_1, x_2, \ldots x_k\} \quad (x_1 < \cdots < x_k \wedge 1 \leq x_i \leq n)$: input combination of length k taken from n objects, n: total number of elements, T: index table calculated by *RLITable*$(\cdot)$.
Output: reverse lexicographical index of combination X.

$r \leftarrow 0$
for $i = 1$ to k **do**
  $r \leftarrow z + T(k-i+1, n-k-x_i+i+1)$.
end for
return $r+1$.

calculated by

$$RLI(X,n) = \sum_{i=1}^{|X|} \binom{n-x_i}{|X|-i+1} + 1.$$

Figure 7 shows an example of the calculation of RLI for $n = 6$ and $k = 3$. For example, $RLI(\{1,3,5\},6) = \binom{5}{3} + \binom{3}{2} + \binom{1}{1} + 1 = 15$.

Corollary 14 (RLI calculation by the index table) *Let T be a $(k, n-k+1)$-size matrix whose element $T(\alpha,\beta) \stackrel{\text{def}}{=} \binom{\alpha+\beta-2}{\alpha}$. Matrix T can be calculated only by $(n-1)(n-k-1)$ time addition and by using T, $RLI(X,n)$ can be calculated in linear time by $RLI(X,n) = \sum_{i=1}^{k} T(k-i+1, n-k-x_i+i+1)$ + 1.*

Algorithm 3 shows the pseudocode used to generate T and Algorithm 4, the pseudocode of $RLI(X,n)$. The inverse function that generates the combination vector for an RLI can be calculated by simply finding the largest column position of T, subtracting the value in the table from the index, and then repeating this k times.

Corollary 15 *Let $RLI(X,n)$ be the reverse lexicographical index defined above for combination $X = \{x_1, x_2, \ldots, x_k\}$. The inverse function of $RLI(X,n)$, that is, the i-th element x_i of*

Algorithm 5 $RLI^{-1}(r,n,k,T)$ calculates the combination vector of length k from n elements, corresponding to the given reverse lexicographical index r.

Input: r: reverse lexicographical index of the combination to be calculated, n: total number of elements, k: length of the combination, T: index table calculated by $RLITable(\cdot)$.

Output: $X = \{x_1, x_2, \ldots, x_k\}$: combination corresponding to r.

```
r ← r − 1.
for i = 1 to k do
    for j = n − k + 1 to 1 do
        if r ≥ T(k − i + 1, j) then
            x_i ← n − k − j + i + 1.
            r ← r − T(k − i + 1, j).
            break
        end if
    end for
end for
return X = {x_1, x_2, ..., x_k}.
```

$RLI^{-1}(RLI(X,n),n,k)$ *can be calculated by, for* $i = 1,\ldots,k$,

$$x_i = \arg\max_j T(k-i+1, n-k-j+i+1) < RLI(X) - \sum_{\alpha=1}^{i-1} T(k-\alpha+1, n-k-x_\alpha+\alpha+1).$$

Algorithm 5 shows the pseudocode used to calculate $RLI^{-1}(r,n,k)$ *for RLI* r *in linear time.*

The search space of 15 is independent of k but dependent on n. This is because once x_i is identified, we need to search only x_j for $j > i$ such that $x_j > x_i$. Therefore, the inverse function $RLI^{-1}(\cdot)$ can generate the combination vector in linear time. By using the binary search, the search of proper objects in a vector can be calculated in log time. See Tamada et al. (2011) for the binary search version of this algorithm.

The advantage of using RLI is that once T is calculated for m, it can be used for calculating $RLI(X,n)$ and $RLI^{-1}(r,n,k)$ for any n and k, where $k \leq n \leq m$. The normal lexicographical order can also be calculated in a similar manner for fixed values of n and k. However, it is required for constructing a different table for different values of n and k.

References

B. P. Buckles and M. Lybanon. Algorithm 515: Generation of a vector from the lexicographical index [G6]. *ACM Transductions on Mathematical Software*, 3(2):180–182, 1977.

D. M. Chickering, D. Geiger, and D. Heckerman. Learning Bayesian networks: Search methods and experimental results. In *Proceedings of the Fifth Conference on Artificial Intelligence and Statistics*, pages 112–128, 1995.

N. Friedman, M. Linial, I. Nachman, and D. Pe'er. Using Bayesian networks to analyze expression data. *J. Computational Biology*, 7:601–620, 2000.

D. Heckerman, D. Geiger, and D. M. Chickering. Learning Bayesian networks: the combination of knowledge and statistical data. *Machine Learning*, 20:197–243, 1995.

S. Imoto, T. Goto, and S. Miyano. Estimation of genetic networks and functional structures between genes by using Bayesian networks and nonparametric regression. *Pacific Symposium on Biocomputing*, 7:175–186, 2002.

M. Koivisto and K. Sood. Exact Bayesian structure discovery in Bayesian networks. *Journal of Machine Learning Research*, 5:549–573, 2004.

S. Ott, S. Imoto, and S. Miyano. Finding optimal models for small gene networks. *Pacific Symposium on Biocomputing*, 9:557–567, 2004.

P. Parviainen and M. Koivisto. Exact structure discovery in Bayesian networks with less space. *Proceedings of the 25th Conference on Uncertainty in Artificial Intelligence (UAI 2009)*, 2009.

J. Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufman Publishers, San Mateo, CA, 1988.

E. Perrier, S. Imoto, and S. Miyano. Finding optimal Bayesian network given a super-structure. *J. Machine Learning Research*, 9:2251–2286, 2008.

T. Silander and P. Myllymäki. A simple approach for finding the globally optimal Bayesian network structure. *Proceedings of the 22th Conference on Uncertainty in Artificial Intelligence (UAI 2006)*, pages 445–452, 2006.

Y. Tamada, S. Imoto, and S. Miyano. Conversion between a combination vector and the lexicographical index in linear time with polynomial time preprocessing, 2011. *submitted*.

I. Tsamardinos, L. E. Brown, and C. F. Aliferis. The max-min hill-climbing Bayesian network structure learning algorithm. *Machine Learning*, 65:31–78, 2006.

Journal of Machine Learning Research 12 (2011) 2461-2488 Submitted 10/09; Revised 12/10; Published 8/10

Distance Dependent Chinese Restaurant Processes

David M. Blei BLEI@CS.PRINCETON.EDU
Department of Computer Science
Princeton University
Princeton, NJ 08544, USA

Peter I. Frazier PF98@CORNELL.EDU
School of Operations Research and Information Engineering
Cornell University
Ithaca, NY 14853, USA

Editor: Carl Edward Rasmussen

Abstract

We develop the distance dependent Chinese restaurant process, a flexible class of distributions over partitions that allows for dependencies between the elements. This class can be used to model many kinds of dependencies between data in infinite clustering models, including dependencies arising from time, space, and network connectivity. We examine the properties of the distance dependent CRP, discuss its connections to Bayesian nonparametric mixture models, and derive a Gibbs sampler for both fully observed and latent mixture settings. We study its empirical performance with three text corpora. We show that relaxing the assumption of exchangeability with distance dependent CRPs can provide a better fit to sequential data and network data. We also show that the distance dependent CRP representation of the traditional CRP mixture leads to a faster-mixing Gibbs sampling algorithm than the one based on the original formulation.

Keywords: Chinese restaurant processes, Bayesian nonparametrics

1. Introduction

Dirichlet process (DP) mixture models provide a valuable suite of flexible clustering algorithms for high dimensional data analysis. Such models have been adapted to text modeling (Teh et al., 2006; Goldwater et al., 2006), computer vision (Sudderth et al., 2005), sequential models (Dunson, 2006; Fox et al., 2007), and computational biology (Xing et al., 2007). Moreover, recent years have seen significant advances in scalable approximate posterior inference methods for this class of models (Liang et al., 2007; Daume, 2007; Blei and Jordan, 2005). DP mixtures have become a valuable tool in modern machine learning.

DP mixtures can be described via the Chinese restaurant process (CRP), a distribution over partitions that embodies the assumed prior distribution over cluster structures (Pitman, 2002). The CRP is fancifully described by a sequence of customers sitting down at the tables of a Chinese restaurant. Each customer sits at a previously occupied table with probability proportional to the number of customers already sitting there, and at a new table with probability proportional to a concentration parameter. In a CRP mixture, customers are identified with data points, and data sitting at the same table belong to the same cluster. Since the number of occupied tables is random, this provides a flexible model in which the number of clusters is determined by the data.

The customers of a CRP are exchangeable—under any permutation of their ordering, the probability of a particular configuration is the same—and this property is essential to connect the CRP mixture to the DP mixture. The reason is as follows. The Dirichlet process is a distribution over distributions, and the DP mixture assumes that the random parameters governing the observations are drawn from a distribution drawn from a Dirichlet process. The observations are conditionally independent given the random distribution, and thus they must be marginally exchangeable.[1] If the CRP mixture did not yield an exchangeable distribution, it could not be equivalent to a DP mixture.

Exchangeability is a reasonable assumption in some clustering applications, but in many it is not. Consider data ordered in time, such as a time-stamped collection of news articles. In this setting, each article should tend to cluster with other articles that are nearby in time. Or, consider spatial data, such as pixels in an image or measurements at geographic locations. Here again, each datum should tend to cluster with other data that are nearby in space. While the traditional CRP mixture provides a flexible prior over partitions of the data, it cannot accommodate such non-exchangeability.

In this paper, we develop the *distance dependent Chinese restaurant process*, a new CRP in which the random seating assignment of the customers depends on the distances between them.[2] These distances can be based on time, space, or other characteristics. Distance dependent CRPs can recover a number of existing dependent distributions (Ahmed and Xing, 2008; Zhu et al., 2005). They can also be arranged to recover the traditional CRP distribution. The distance dependent CRP expands the palette of infinite clustering models, allowing for many useful non-exchangeable distributions as priors on partitions.[3]

The key to the distance dependent CRP is that it represents the partition with *customer assignments*, rather than table assignments. While the traditional CRP connects customers to tables, the distance dependent CRP connects customers to other customers. The partition of the data, that is, the table assignment representation, arises from these customer connections. When used in a Bayesian model, the customer assignment representation allows for a straightforward Gibbs sampling algorithm for approximate posterior inference (see Section 3). This provides a new tool for flexible clustering of non-exchangeable data, such as time-series or spatial data, as well as a new algorithm for inference with traditional CRP mixtures.

1.1 Related Work

Several other non-exchangeable priors on partitions have appeared in recent research literature. Some can be formulated as distance dependent CRPs, while others represent a different class of models. The most similar to the distance dependent CRP is the probability distribution on partitions presented in Dahl (2008). Like the distance dependent CRP, this distribution may be constructed through a collection of independent priors on customer assignments to other customers, which then implies a prior on partitions. Unlike the distance dependent CRP, however, the distribution pre-

1. That these parameters will exhibit a clustering structure is due to the discreteness of distributions drawn from a Dirichlet process (Ferguson, 1973; Antoniak, 1974; Blackwell, 1973).
2. This is an expanded version of our shorter conference paper on this subject (Blei and Frazier, 2010). This version contains new perspectives on inference and new results.
3. We avoid calling these clustering models "Bayesian nonparametric" (BNP) because they cannot necessarily be cast as a mixture model originating from a random measure, such as the DP mixture model. The DP mixture is BNP because it includes a prior over the infinite space of probability densities, and the CRP mixture is only BNP in its connection to the DP mixture. That said, most applications of this machinery are based around letting the data determine their number of clusters. The fact that it actually places a distribution on the infinite-dimensional space of probability measures is usually not exploited.

sented in Dahl (2008) requires normalization of these customer assignment probabilities. The model in Dahl (2008) may always be written as a distance dependent CRP, although the normalization requirement prevents the reverse from being true (see Section 2). We note that Dahl (2008) does not present an algorithm for sampling from the posterior, but the Gibbs sampler presented here for the distance dependent CRP can also be employed for posterior inference in that model.

There are a number of Bayesian nonparametric models that allow for dependence between (marginal) partition membership probabilities. These include the dependent Dirichlet process (MacEachern, 1999) and other similar processes (Duan et al., 2007; Griffin and Steel, 2006; Xue et al., 2007). Such models place a prior on collections of sampling distributions drawn from Dirichlet processes, with one sampling distribution drawn per possible value of covariate and sampling distributions from similar covariates more likely to be similar. Marginalizing out the sampling distributions, these models induce a prior on partitions by considering two customers to be clustered together if their sampled values are equal. (Recall, these sampled values are drawn from the sampling distributions corresponding to their respective covariates.) This prior need not be exchangeable if we do not condition on the covariate values.

Distance dependent CRPs represent an alternative strategy for modeling non-exchangeability. The difference hinges on marginal invariance, the property that a missing observation does not affect the joint distribution. In general, dependent DPs exhibit marginal invariance while distance dependent CRPs do not. For the practitioner, this property is a modeling choice, which we discuss in Section 2. Section 4 shows that distance dependent CRPs and dependent DPs represent nearly distinct classes of models, intersecting only in the original DP or CRP.

Still other prior distributions on partitions include those presented in Ahmed and Xing (2008) and Zhu et al. (2005), both of which are special cases of the distance dependent CRP. Rasmussen and Ghahramani (2002) use a gating network similar to the distance dependent CRP to partition datapoints among experts in way that is more likely to assign nearby points to the same cluster. Also included are the product partition models of Hartigan (1990), their recent extension to dependence on covariates (Muller et al., 2008), and the dependent Pitman-Yor process (Sudderth and Jordan, 2008). A review of prior probability distributions on partitions is presented in Mueller and Quintana (2008). The Indian Buffet Process, a Bayesian non-parametric prior on sparse binary matrices, has also been generalized to model non-exchangeable data by Miller et al. (2008). We further discuss these priors in relation to the distance dependent CRP in Section 2.

The rest of this paper is organized as follows. In Section 2 we develop the distance dependent CRP and discuss its properties. We show how the distance dependent CRP may be used to model discrete data, both fully-observed and as part of a mixture model. In Section 3 we show how the customer assignment representation allows for an efficient Gibbs sampling algorithm. In Section 4 we show that distance dependent CRPs and dependent DPs represent distinct classes of models. Finally, in Section 5 we describe an empirical study of three text corpora using the distance dependent CRP. We show that relaxing the assumption of exchangeability with distance dependent CRPs can provide a better fit to sequential data. We also show its alternative formulation of the traditional CRP leads to a faster-mixing Gibbs sampling algorithm than the one based on the original formulation.

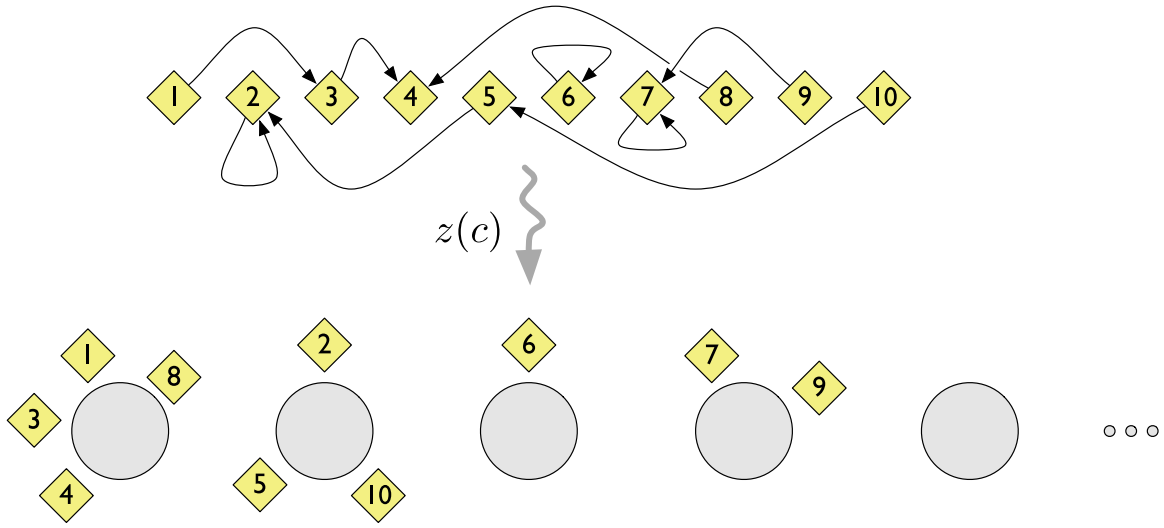


Figure 1: An illustration of the distance dependent CRP. The process operates at the level of customer assignments, where each customer chooses either another customer or no customer according to Equation (2). Customers that chose not to connect to another are indicated with a self link The table assignments, a representation of the partition that is familiar to the CRP, are derived from the customer assignments.

2. Distance-dependent CRPs

The Chinese restaurant process (CRP) is a probability distribution over partitions (Pitman, 2002). It is described by considering a Chinese restaurant with an infinite number of tables and a sequential process by which customers enter the restaurant and each sit down at a randomly chosen table. After N customers have sat down, their configuration at the tables represents a random partition. Customers sitting at the same table are in the same cycle.

In the traditional CRP, the probability of a customer sitting at a table is computed from the number of other customers already sitting at that table. Let z_i denote the table assignment of the ith customer, assume that the customers $z_{1:(i-1)}$ occupy K tables, and let n_k denote the number of customers sitting at table k. The traditional CRP draws each z_i sequentially,

$$p(z_i = k \,|\, z_{1:(i-1)}, \alpha) \propto \begin{cases} n_k & \text{for} \quad k \leq K \\ \alpha & \text{for} \quad k = K+1, \end{cases} \tag{1}$$

where α is a given scaling parameter. When all N customers have been seated, their table assignments provide a random partition. Though the process is described sequentially, the CRP is exchangeable. The probability of a particular partition of N customers is invariant to the order in which they sat down.

We now introduce the *distance dependent CRP*. In this distribution, the seating plan probability is described in terms of the probability of a customer sitting with each of the other *customers*. The allocation of customers to tables is a by-product of this representation. If two customers are

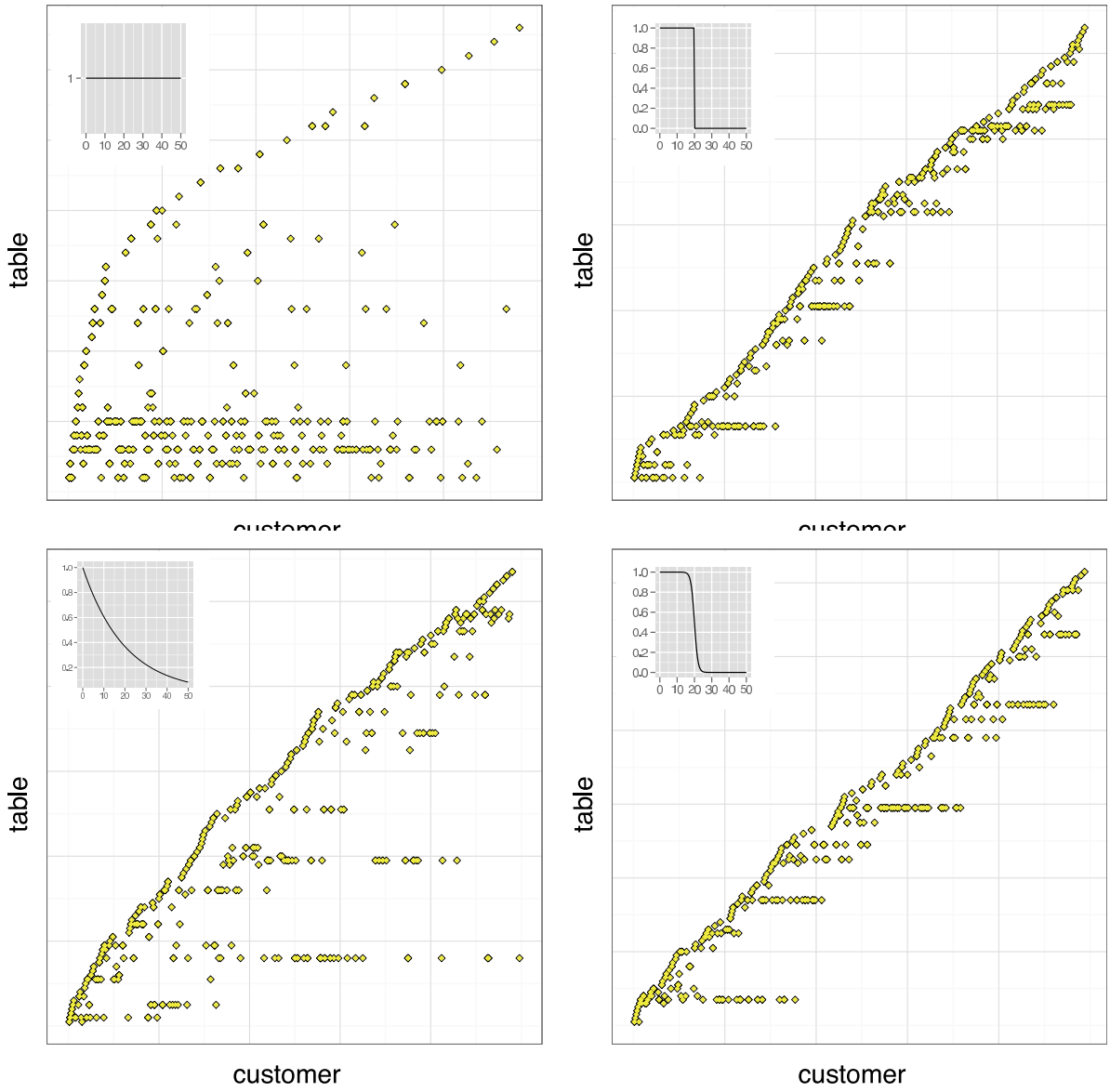


Figure 2: Draws from sequential CRPs. Illustrated are draws for different decay functions, which are inset: (1) The traditional CRP; (2) The window decay function; (3) The exponential decay function; (4) The logistic decay function. The table assignments are illustrated, which are derived from the customer assignments drawn from the distance dependent CRP. The decay functions (inset) are functions of the distance between the current customer and each previous customer.

reachable by a sequence of interim customer assignments, then they at the same table. This is illustrated in Figure 1.

Let c_i denote the ith customer assignment, the index of the customer with whom the ith customer is sitting. Let d_{ij} denote the distance measurement between customers i and j, let D denote the set of all distance measurements between customers, and let f be a decay function (described in more detail below). The distance dependent CRP independently draws the customer assignments conditioned on the distance measurements,

$$p(c_i = j \mid D, \alpha) \propto \begin{cases} f(d_{ij}) & \text{if} \quad j \neq i \\ \alpha & \text{if} \quad i = j. \end{cases} \tag{2}$$

Notice the customer assignments do not depend on other customer assignments, only the distances between customers. Also notice that j ranges over the entire set of customers, and so any customer may sit with any other. (If desirable, restrictions are possible through the distances d_{ij}. See the discussion below of sequential CRPs.)

As we mentioned above, customers are assigned to tables by considering sets of customers that are reachable from each other through the customer assignments. (Again, see Figure 1.) We denote the induced table assignments $z(\mathbf{c})$, and notice that many configurations of customer assignments $\mathbf{c}$ might lead to the same table assignment. Finally, customer assignments can produce a cycle, for example, customer 1 sits with 2 and customer 2 sits with 1. This still determines a valid table assignment: All customers sitting in a cycle are assigned to the same table.

By being defined over customer assignments, the distance dependent CRP provides a more expressive distribution over partitions than models based on table assignments. This distribution is determined by the nature of the distance measurements and the decay function. For example, if each customer is time-stamped, then d_{ij} might be the time difference between customers i and j; the decay function can encourage customers to sit with those that are contemporaneous. If each customer is associated with a location in space, then d_{ij} might be the Euclidean distance between them; the decay function can encourage customers to sit with those that are in proximity.[4] For many sets of distance measurements, the resulting distribution over partitions is no longer exchangeable; this is an appropriate distribution to use when exchangeability is not a reasonable assumption.

2.1 Decay Functions

In general, the decay function mediates how distances between customers affect the resulting distribution over partitions. We assume that the decay function f is non-increasing, takes non-negative finite values, and satisfies $f(\infty) = 0$. We consider several types of decay as examples, all of which satisfy these nonrestrictive assumptions.

The *window decay* $f(d) = 1[d < a]$ only considers customers that are at most distance a from the current customer. The *exponential decay* $f(d) = e^{-d/a}$ decays the probability of linking to an earlier customer exponentially with the distance to the current customer. The *logistic decay* $f(d) = \exp(-d+a)/(1+\exp(-d+a))$ is a smooth version of the window decay. Each of these affects the distribution over partitions in a different way.

4. The probability distribution over partitions defined by Equation (2) is similar to the distribution over partitions presented in Dahl (2008). That probability distribution may be specified by Equation (2) if $f(d_{ij})$ is replaced by a non-negative value h_{ij} that satisfies a normalization requirement $\sum_{i \neq j} h_{ij} = N - 1$ for each j. Thus, the model presented in Dahl (2008) may be understood as a normalized version of the distance dependent CRP. To write this model as a distance dependent CRP, take $d_{ij} = 1/h_{ij}$ and $f(d) = 1/d$ (with $1/0 = \infty$ and $1/\infty = 0$), so that $f(d_{ij}) = h_{ij}$.

2.2 Sequential CRPs and the Traditional CRP

With certain types of distance measurements and decay functions, we obtain the special case of *sequential CRPs*.[5] A sequential CRP is constructed by assuming that $d_{ij} = \infty$ for those $j > i$. With our previous requirement that $f(\infty) = 0$, this guarantees that no customer can be assigned to a later customer, that is, $p(c_i \leq i \,|\, D) = 1$. The sequential CRP lets us define alternative formulations of some previous time-series models. For example, with a window decay function and $a = 1$, we recover the model studied in Ahmed and Xing (2008). With a logistic decay function, we recover the model studied in Zhu et al. (2005). In our empirical study we will examine sequential models in detail.

The sequential CRP can re-express the traditional CRP. Specifically, the traditional CRP is recovered when $f(d) = 1$ for $d \neq \infty$ and $d_{ij} < \infty$ for $j < i$. To see this, consider the marginal distribution of a customer sitting at a particular table, given the previous customers' assignments. The probability of being assigned to each of the other customers at that table is proportional to one. Thus, the probability of sitting at that table is proportional to the number of customers already sitting there. Moreover, the probability of not being assigned to a previous customer is proportional to the scaling parameter α. This is precisely the traditional CRP distribution of Equation (1). Although these models are the same, the corresponding Gibbs samplers are different (see Section 5.4).

Figure 2 illustrates seating assignments (at the *table* level) derived from draws from sequential CRPs with each of the decay functions described above, including the original CRP. (To adapt these settings to the sequential case, the distances are $d_{ij} = i - j$ for $j < i$ and $d_{ij} = \infty$ for $j > i$.) Compared to the traditional CRP, customers tend to sit at the same table with other nearby customers. We emphasize that sequential CRPs are only one type of distance dependent CRP. Other distances, combined with the formulation of Equation (2), lead to a variety of other non-exchangeable distributions over partitions.

2.3 Marginal Invariance

The traditional CRP is *marginally invariant*: Marginalizing over a particular customer gives the same probability distribution as if that customer were not included in the model at all. The distance dependent CRP does not generally have this property, allowing it to capture the way in which influence might be transmitted from one point to another. See Section 4 for a precise characterization of the class of distance dependent CRPs that are marginally invariant.

To see when this might be a relevant property, consider the goal of modeling preferences of people within a social network. The model used should reflect the fact that persons A and B are more likely to share preferences if they also share a common friend C. Any marginally invariant model, however, would insist that the distribution of the preferences of A and B is the same whether (1) they have no such common friend C, or (2) they do but his preferences are unobserved and hence marginalized out. In this setting, we might prefer a model that is not marginally invariant. Knowing that they have a common friend affects the probability that A and B share preferences, regardless of whether the friend's preferences are observed. A similar example is modeling the spread of disease. Suddenly discovering a city between two others—even if the status of that city

5. Even though the traditional CRP is described as a sequential process, it gives an exchangeable distribution. Thus, sequential CRPs, which include both the traditional CRP as well as non-exchangeable distributions, are more expressive than the traditional CRP.

is unobserved—should change our assessment of the probability that the disease travels between them.

We note, however, that if observations are missing then models that are not marginally invariant require that relevant conditional distributions be computed as ratios of normalizing constants. In contrast, marginally invariant models afford a more convenient factorization, and so allow easier computation. Even when faced with data that clearly deviates from marginal invariance, the modeler may be tempted to use a marginally invariant model, choosing computational convenience over fidelity to the data.

We have described a general formulation of the distance dependent CRP. We now describe two applications to Bayesian modeling of discrete data, one in a fully observed model and the other in a mixture model. These examples illustrate how one might use the posterior distribution of the partitions, given data and an assumed generating process based on the distance dependent CRP. We will focus on models of discrete data and we will use the terminology of document collections to describe these models.[6] Thus, our observations are assumed to be collections of words from a fixed vocabulary, organized into documents.

2.4 Language Modeling

In the language modeling application, each document is associated with a distance dependent CRP, and its tables are embellished with IID draws from a base distribution over terms or words. (The documents share the same base distribution.) The generative process of words in a document is as follows. The data are first placed at tables via customer assignments, and then assigned to the word associated with their tables. Subsets of the data exhibit a partition structure by sharing the same table.

When using a traditional CRP, this is a formulation of a simple Dirichlet-smoothed language model. Alternatives to this model, such as those using the Pitman-Yor process, have also been applied in this setting (Teh, 2006; Goldwater et al., 2006). We consider a sequential CRP, which assumes that a word is more likely to occur near itself in a document. Words are still considered contagious—seeing a word once means we're likely to see it again—but the window of contagion is mediated by the decay function.

More formally, given a decay function f, sequential distances D, scaling parameter α, and base distribution G_0 over discrete words, N words are drawn as follows,

1. For each word $i \in \{1,\ldots,N\}$ draw assignment $c_i \sim \text{dist-CRP}(\alpha, f, D)$.
2. For each table, $k \in \{1,\ldots\}$, draw a word $w^* \sim G_0$.
3. For each word $i \in \{1,\ldots,N\}$, assign the word $w_i = w^*_{z(\mathbf{c})_i}$.

The notation $z(\mathbf{c})_i$ is the table assignment of the ith customer in the table assignments induced by the complete collection of customer assignments.

For each document, we observe a sequence of words $w_{1:N}$ from which we can infer their seating assignments in the distance dependent CRP. The partition structure of observations—that is, which words are the same as other words—indicates either that they share the same table in the seating

6. While we focus on text, these models apply to any discrete data, such as genetic data, and, with modification, to non-discrete data as well. That said, CRP-based methods have been extensively applied to text modeling and natural language processing (Teh et al., 2006; Johnson et al., 2007; Li et al., 2007; Blei et al., 2010).

arrangement, or that two tables share the same term drawn from G_0. We have not described the process sequentially, as one would with a traditional CRP, in order to emphasize the three stage process of the distance dependent CRP—first the customer assignments and table parameters are drawn, and then the observations are assigned to their corresponding parameter. However, the sequential distances D guarantee that we can draw each word successively. This, in turn, means that we can easily construct a predictive distribution of future words given previous words. (See Section 3 below.)

2.5 Mixture Modeling

The second model we study is akin to the CRP mixture or (equivalently) the DP mixture, but differs in that the mixture component for a data point depends on the mixture component for nearby data. Again, each table is endowed with a draw from a base distribution G_0, but here that draw is a distribution over mixture component parameters. In the document setting, observations are documents (as opposed to individual words), and G_0 is typically a Dirichlet distribution over distributions of words (Teh et al., 2006). The data are drawn as follows:

1. For each document $i \in [1,N]$ draw assignment $c_i \sim \text{dist-CRP}(\alpha, f, D)$.
2. For each table, $k \in \{1,\ldots\}$, draw a parameter $\theta^*_k \sim G_0$.
3. For each document $i \in [1,N]$, draw $w_i \sim F(\theta_{z(\mathbf{c})_i})$.

In Section 5, we will study the sequential CRP in this setting, choosing its structure so that contemporaneous documents are more likely to be clustered together. The distances d_{ij} can be the differences between indices in the ordering of the data, or lags between external measurements of distance like date or time. (Spatial distances or distances based on other covariates can be used to define more general mixtures, but we leave these settings for future work.) Again, we have not defined the generative process sequentially but, as long as D respects the assumptions of a sequential CRP, an equivalent sequential model is straightforward to define.

2.6 Relationship to Dependent Dirichlet Processes

More generally, the distance dependent CRP mixture provides an alternative to the dependent Dirichlet process (DDP) mixture as an infinite clustering model that models dependencies between the latent component assignments of the data (MacEachern, 1999). The DDP has been extended to sequential, spatial, and other kinds of dependence (Griffin and Steel, 2006; Duan et al., 2007; Xue et al., 2007). In all these settings, statisticians have appealed to truncations of the stick-breaking representation for approximate posterior inference, citing the dependency between data as precluding the more efficient techniques that integrate out the component parameters and proportions. In contrast, distance dependent CRP mixtures are amenable to Gibbs sampling algorithms that integrate out these variables (see Section 3).

An alternative to the DDP formalism is the Bayesian density regression (BDR) model of Dunson et al. (2007). In BDR, each data point is associated with a random measure and is drawn from a mixture of per-data random measures where the mixture proportions are related to the distance between data points. Unlike the DDP, this model affords a Gibbs sampler where the random measures can be integrated out.

However, it is still different in spirit from the distance dependent CRP. Data are drawn from distributions that are similar to distributions of nearby data, and the particular values of nearby data impose softer constraints than those in the distance dependent CRP. As an extreme case, consider a random partition of the nodes of a network, where distances are defined in terms of the number of hops between nodes. Further, suppose that there are several disconnected components in this network, that is, pairs of nodes that are not reachable from each other. In the DDP model, these nodes are very likely not to be partitioned in the same group. In the ddCRP model, however, it is impossible for them to be grouped together.

We emphasize that DDP mixtures (and BDR) and distance dependent CRP mixtures are *different* classes of models. DDP mixtures are Bayesian nonparametric models, interpretable as data drawn from a random measure, while the distance dependent CRP mixtures generally are not. DDP mixtures exhibit marginal invariance, while distance dependent CRPs generally do not (see Section 4). In their ability to capture dependence, these two classes of models capture similar assumptions, but the appropriate choice of model depends on the modeling task at hand.

3. Posterior Inference and Prediction

The central computational problem for distance dependent CRP modeling is posterior inference, determining the conditional distribution of the hidden variables given the observations. This posterior is used for exploratory analysis of the data and how it clusters, and is needed to compute the predictive distribution of a new data point given a set of observations.

Regardless of the likelihood model, the posterior will be intractable to compute because the distance dependent CRP places a prior over a combinatorial number of possible customer configurations. In this section we provide a general strategy for approximating the posterior using Monte Carlo Markov chain (MCMC) sampling. This strategy can be used in either fully-observed or mixture settings, and can be used with arbitrary distance functions. (For example, in Section 5 we illustrate this algorithm with both sequential distance functions and graph-based distance functions and in both fully-observed and mixture settings.)

In MCMC, we aim to construct a Markov chain whose stationary distribution is the posterior of interest. For distance dependent CRP models, the state of the chain is defined by c_i, the customer assignments for each data point. We will also consider $z(\mathbf{c})$, which are the table assignments that follow from the customer assignments (see Figure 1). Let $\eta = \{D, \alpha, f, G_0\}$ denote the set of model hyperparameters. It contains the distances D, the scaling factor α, the decay function f, and the base measure G_0. Let x denote the observations.

In Gibbs sampling, we iteratively draw from the conditional distribution of each latent variable given the other latent variables and observations. (This defines an appropriate Markov chain, see Neal 1993.) In distance dependent CRP models, the Gibbs sampler iteratively draws from

$$p(c_i^{(\text{new})} \mid \mathbf{c}_{-i}, \mathbf{x}, \eta) \propto p(c_i^{(\text{new})} \mid D, \alpha) p(\mathbf{x} \mid z(\mathbf{c}_{-i} \cup c_i^{(\text{new})}), G_0).$$

The first term is the distance dependent CRP prior from Equation (2).

The second term is the likelihood of the observations under the partition given by $z(\mathbf{c}_{-i} \cup c_i^{(new)})$. This can be thought of as removing the current link from the ith customer and then considering how each alternative new link affects the likelihood of the observations. Before examining this likelihood, we describe how removing and then replacing a customer link affects the underlying partition (i.e., table assignments).

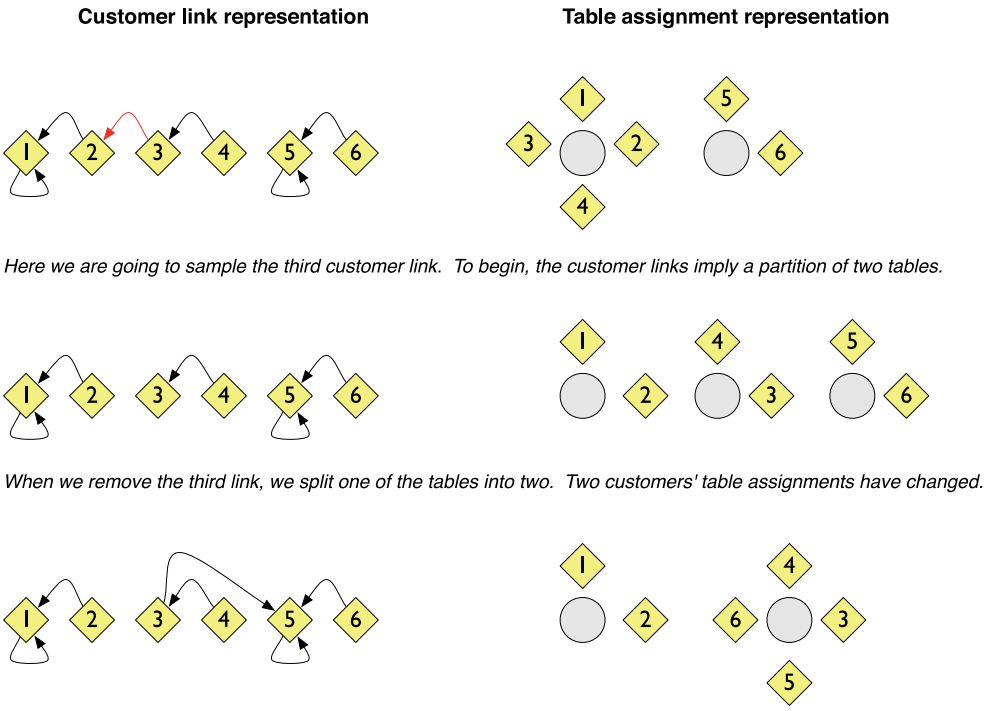


Figure 3: An example of a single step of the Gibbs sampler. Here we illustrate a scenario that highlights all the ways that the sampler can move: A table can be split when we remove the customer link before conditioning; and two tables can join when we resample that link.

To begin, consider the effect of removing a customer link. What is the difference between the partition $z(\mathbf{c})$ and $z(\mathbf{c}_{-i})$? There are two cases.

The first case is that a table splits. This happens when c_i is the only connection between the ith data point and a particular table. Upon removing c_i, the customers at its table are split in two: those customers pointing (directly or indirectly) to i are at one table; the other customers previously seated with i are at a different table. (See the change from the first to second rows of Figure 3.)

The second case is that there is no change. If the ith link is not the only connection between customer i and his table or if c_i was a self-link ($c_i = i$) then the tables remain the same. In this case, $z(\mathbf{c}_{-i}) = z(\mathbf{c})$.

Now consider the effect of replacing the customer link. What is the difference between the partition $z(\mathbf{c}_{-i})$ and $z(\mathbf{c}_{-i} \cup c_i^{(\text{new})})$? Again there are two cases. The first case is that $c_i^{(\text{new})}$ joins two tables in $z(\mathbf{c}_{-i})$. Upon adding $c_i^{(\text{new})}$, the customers at its table become linked to another set of customers. (See the change from the second to third rows of Figure 3.)

The second case, as above, is that there is no change. This occurs if $c_i^{(\text{new})}$ points to a customer that is already at its table under $z(\mathbf{c}_{-i})$ or if $c_i^{(\text{new})}$ is a self-link.

With the changed partition in hand, we now compute the likelihood term. We first compute the likelihood term for partition $z(\mathbf{c})$. The likelihood factors into a product of terms, each of which is the probability of the set of observations at each table. Let $|z(\mathbf{c})|$ be the number of tables and $z^k(\mathbf{c})$ be the set of indices that are assigned to table k. The likelihood term is

$$p(\mathbf{x} \,|\, z(\mathbf{c}), G_0) = \prod_{k=1}^{|z(\mathbf{c})|} p(\mathbf{x}_{z^k(\mathbf{c})} \,|\, G_0). \tag{3}$$

Because of this factorization, the Gibbs sampler need only compute terms that correspond to changes in the partition. Consider the partition $z(\mathbf{c}_{-i})$, which may have split a table, and the new partition $z(\mathbf{c}_{-i} \cup c^{(\text{new})})$. There are three cases to consider. First, c_i might link to itself—there will be no change to the likelihood function because a self-link cannot join two tables. Second, c_i might link to another table but cause no change in the partition. Finally, c_i might link to another table and join two tables k and ℓ. The Gibbs sampler for the distance dependent CRP is thus

$$p(c_i^{(\text{new})} \,|\, \mathbf{c}_{-i}, \mathbf{x}, \eta) \propto \begin{cases} \alpha & \text{if } c_i^{(\text{new})} \text{ is equal to } i. \\ f(d_{ij}) & \text{if } c_i^{(\text{new})} = j \text{ does not join two tables.} \\ f(d_{ij}) \dfrac{p(\mathbf{x}_{z^k(\mathbf{c}_{-i}) \cup z^\ell(\mathbf{c}_{-i})} \,|\, G_0)}{p(\mathbf{x}_{z^k(\mathbf{c}_{-i})} \,|\, G_0) p(\mathbf{x}_{z^\ell(\mathbf{c}_{-i})} \,|\, G_0)} & \text{if } c_i^{(\text{new})} = j \text{ joins tables } k \text{ and } \ell. \end{cases}$$

The specific form of the terms in Equation (3) depend on the model. We first consider the fully observed case (i.e., "language modeling"). Recall that the partition corresponds to words of the same type, but that more than one table can contain identical types. (For example, four tables could contain observations of the word "peanut." But, observations of the word "walnut" cannot sit at any of the peanut tables.) Thus, the likelihood of the data is simply the probability under G_0 of a representative from each table, for example, the first customer, times a product of indicators to ensure that all observations are equal,

$$p(\mathbf{x}_{z^k(\mathbf{c})} \,|\, G_0) = p(x_{z^k(\mathbf{c})_1} \,|\, G_0) \textstyle\prod_{i \in z^k(\mathbf{c})} 1(x_i = x_{z^k(\mathbf{c})_1}),$$

where $z^k(\mathbf{c})_1$ is the index of the first customer assigned to table k.

In the mixture model, we compute the marginal probability that the set of observations from each table are drawn independently from the same parameter, which itself is drawn from G_0. Each term is

$$p(\mathbf{x}_{z^k(\mathbf{c})} \,|\, G_0) = \int \left(\prod_{i \in z^k(\mathbf{c})} p(x_i \,|\, \theta) \right) p(\theta \,|\, G_0) d\theta.$$

Because this term marginalizes out the mixture component θ, the result is a collapsed sampler for the mixture model. When G_0 and $p(x \,|\, \theta)$ form a conjugate pair, the integral is straightforward to compute. In nonconjugate settings, an additional layer of sampling is needed.

3.1 Prediction

In prediction, our goal is to compute the conditional probability distribution of a new data point x_{new} given the data set $\mathbf{x}$. This computation relies on the posterior. Recall that D is the set of distances between all the data points. The predictive distribution is

$$p(x_{\text{new}} | \mathbf{x}, D, G_0, \alpha) = \sum_{c_{\text{new}}} p(c_{\text{new}} \,|\, D, \alpha) \sum_{\mathbf{c}} p(x_{\text{new}} | c_{\text{new}}, \mathbf{c}, \mathbf{x}, G_0) p(\mathbf{c} | \mathbf{x}, D, \alpha, G_0).$$

The outer summation is over the customer assignment of the new data point; its prior probability only depends on the distance matrix D. The inner summation is over the posterior customer assignments of the data set; it determines the probability of the new data point conditioned on the previous data and its partition. In this calculation, the difference between sequential distances and arbitrary distances is important.

Consider sequential distances and suppose that x_{new} is a future data point. In this case, the distribution of the data set customer assignments $\mathbf{c}$ does not depend on the new data point's location in time. The reason is that data points can only connect to data points in the past. Thus, the posterior $p(\mathbf{c} \,|\, \mathbf{x}, D, \alpha, G_0)$ is unchanged by the addition of the new data, and we can use previously computed Gibbs samples to approximate it.

In other situations—nonsequential distances or sequential distances where the new data occurs somewhere in the middle of the sequence—the discovery of the new data point changes the posterior $p(\mathbf{c} \,|\, \mathbf{x}, D, \alpha, G_0)$. The reason is that the knowledge of where the new data is relative to the others (i.e., the information in D) changes the prior over customer assignments and thus changes the posterior as well. This new information requires rerunning the Gibbs sampler to account for the new data point. Finally, note that the special case where we know the new data's location in advance (without knowing its value) does not require rerunning the Gibbs sampler.

4. Marginal Invariance

In Section 2 we discussed the property of *marginal invariance*, where removing a customer leaves the partition distribution over the remaining customers unchanged. When a model has this property, unobserved data may simply be ignored. We mentioned that the traditional CRP is marginally invariant, while the distance dependent CRP does not necessarily have this property.

In fact, the traditional CRP is the *only* distance dependent CRP that is marginally invariant.[7] The details of this characterization are given in the appendix. This characterization of marginally

7. One can also create a marginally invariant distance dependent CRP by combining several independent copies of the traditional CRP. Details are discussed in the appendix.

invariant CRPs contrasts the distance dependent CRP with the alternative priors over partitions induced by random measures, such as the Dirichlet process.

In addition to the Dirichlet process, random-measure models include the dependent Dirichlet process (MacEachern, 1999) and the order-based dependent Dirichlet process (Griffin and Steel, 2006). These models suppose that data from a given covariate were drawn independently from a fixed latent sampling probability measure. These models then suppose that these sampling measures were drawn from some parent probability measure. Dependence between the randomly drawn sampling measures is achieved through this parent probability measure.

We formally define a random-measure model as follows. Let $\mathbb{X}$ and $\mathbb{Y}$ be the sets in which covariates and observations take their values, let $x_{1:N} \subset \mathbb{X}$, $y_{1:N} \subset \mathbb{Y}$ be the set of observed covariates and their corresponding sampled values, and let $M(\mathbb{Y})$ be the space of probability measures on $\mathbb{Y}$. A random-measure model is any probability distribution on the samples $y_{1:N}$ induced by a probability measure G on the space $M(\mathbb{Y})^{\mathbb{X}}$. This random-measure model may be written

$$y_n \mid x_n \sim \mathbb{P}_{x_n}, \qquad (\mathbb{P}_x)_{x\in\mathbb{X}} \sim G,$$

where the y_n are conditionally independent of each other given $(\mathbb{P}_x)_{x\in\mathbb{X}}$. Such models implicitly induce a distribution on partitions of the data by taking all points n whose sampled values y_n are equal to be in the same cluster.

In such random-measure models, the (prior) distribution on y_{-n} does not depend on x_n, and so such models are marginally invariant, regardless of the points $x_{1:n}$ and the distances between them. From this observation, and the lack of marginal invariance of the distance dependent CRP, it follows that the distributions on partitions induced by random-measure models are different from the distance dependent CRP. The only distribution that is both a distance dependent CRP, and is also induced by a random-measure model, is the traditional CRP.

Thus, distance dependent CRPs are generally not marginally invariant, and so are appropriate for modeling situations that naturally depart from marginal invariance. This distinguishes priors obtained with distance dependent CRPs from those obtained from random-measure models, which are appropriate when marginal invariance is a reasonable assumption.

5. Empirical Study

We studied the distance dependent CRP in the language modeling and mixture settings on four text data sets. We explored both time dependence, where the sequential ordering of the data is respected via the decay function and distance measurements, and network dependence, where the data are connected in a graph. We show below that the distance dependent CRP gives better fits to text data in both the fully-observed and mixture modeling settings.[8]

Further, we compared the traditional Gibbs sampler for DP mixtures to the Gibbs sampler for the distance dependent CRP formulation of DP mixtures. We found that the sampler based on customer assignments mixes faster than the traditional sampler.

5.1 Language Modeling

We evaluated the fully-observed distance dependent CRP models on two data sets: a collection of 100 OCR'ed documents from the journal *Science* and a collection of 100 world news articles from

8. Our R implementation of Gibbs sampling for ddCRP models is available at `http://www.cs.princeton.edu/~blei/downloads/ddcrp.tgz`

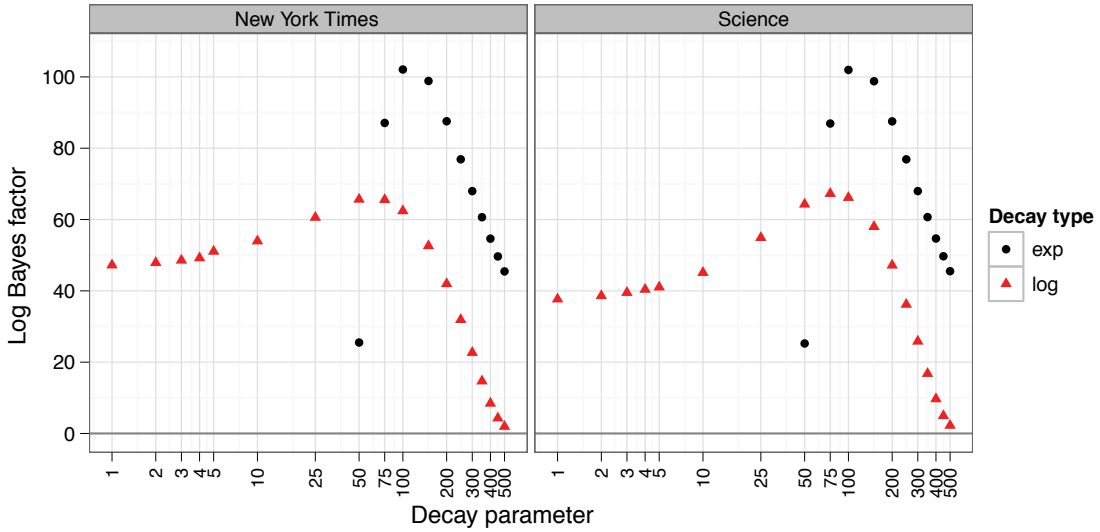


Figure 4: Bayes factors of the distance dependent CRP versus the traditional CRP on documents from *Science* and the *New York Times*. The black line at 0 denotes an equal fit between the traditional CRP and distance dependent CRP, while positive values denote a better fit for the distance dependent CRP. Also illustrated are standard errors across documents.

the *New York Times*. We modeled each document independently. We assess sampler convergence visually, examining the autocorrelation plots of the log likelihood of the state of the chain (Robert and Casella, 2004).

We compare models by estimating the Bayes factor, the ratio of the probability under the distance dependent CRP to the probability under the traditional CRP (Kass and Raftery, 1995). For a decay function f, this Bayes factor is

$$BF_{f,\alpha} = p(w_{1:N} \,|\, \text{dist-CRP}_{f,\alpha}) / p(w_{1:N} \,|\, \text{CRP}_{\alpha}).$$

A value greater than one indicates an improvement of the distance dependent CRP over the traditional CRP. Following Geyer and Thompson (1992), we estimate this ratio with a Monte Carlo estimate from posterior samples.

Figure 4 illustrates the average log Bayes factors across documents for various settings of the exponential and logistic decay functions. The logistic decay function always provides a better model than the traditional CRP; the exponential decay function provides a better model at certain settings of its parameter. (These curves are for the hierarchical setting with the base distribution over terms G_0 unobserved; the shapes of the curves are similar in the non-hierarchical settings.)

5.2 Mixture Modeling

We examined the distance dependent CRP mixture on two text corpora. We analyzed one month of the *New York Times* (NYT) time-stamped by day, containing 2,777 articles, 3,842 unique terms and

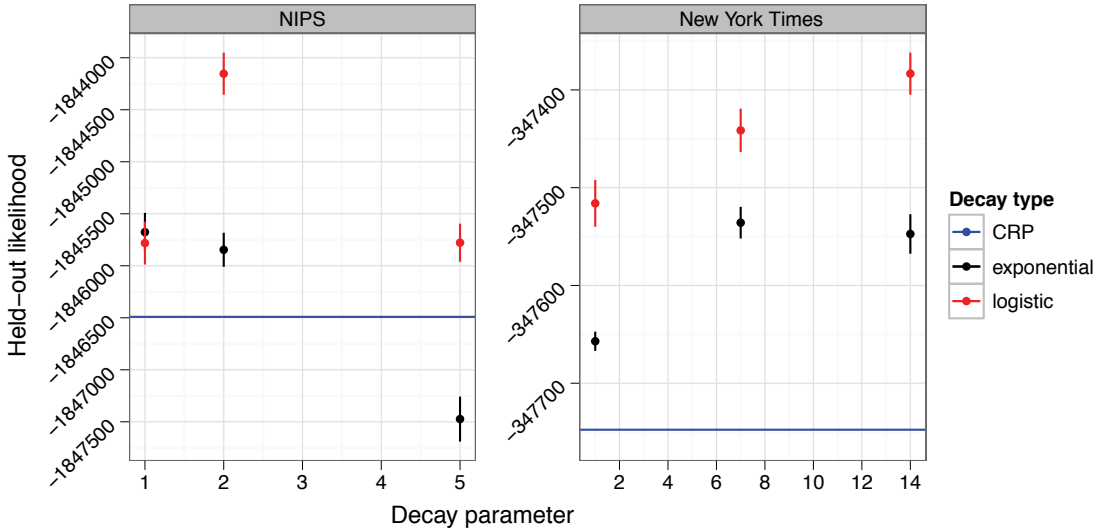


Figure 5: Predictive held-out log likelihood for the last year of NIPS and last three days of the *New York Times* corpus. Error bars denote standard errors across MCMC samples. On the NIPS data, the distance dependent CRP outperforms the traditional CRP for the logistic decay with a decay parameter of 2 years. On the *New York Times* data, the distance dependent CRP outperforms the traditional CRP in almost all settings tested.

530K observed words. We also analyzed 12 years of NIPS papers time-stamped by year, containing 1,740 papers, 5,146 unique terms, and 1.6M observed words. Distances D were differences between time-stamps.

In both corpora we removed the last 250 articles as held out data. In the NYT data, this amounts to three days of news; in the NIPS data, this amounts to papers from the 11th and 12th year. (We retain the time stamps of the held-out articles because the predictive likelihood of an article's contents depends on its time stamp, as well as the time stamps of earlier articles.) We evaluate the models by estimating the predictive likelihood of the held out data. The results are in Figure 5. On the NYT corpus, the distance dependent CRPs definitively outperform the traditional CRP. A logistic decay with a window of 14 days performs best. On the NIPS corpus, the logistic decay function with a decay parameter of 2 years outperforms the traditional CRP. In general, these results show that non-exchangeable models given by the distance dependent CRP mixture provide a better fit than the exchangeable CRP mixture.

5.3 Modeling Networked Data

The previous two examples have considered data analysis settings with a sequential distance function. However, the distance dependent CRP is a more general modeling tool. Here, we demonstrate its flexibility by analyzing a set of *networked documents* with a distance dependent CRP mixture model. Networked data induces an entirely different distance function, where any data point may

link to an arbitrary set of other data. We emphasize that we can use the same Gibbs sampling algorithms for both the sequential and networked settings.

Specifically, we analyzed the CORA data set, a collection of Computer Science abstracts that are connected if one paper cites the other (McCallum et al., 2000). One natural distance function is the number of connections between data (and ∞ if two data points are not reachable from each other). We use the window decay function with parameter 1, enforcing that a customer can only link to itself or to another customer that refers to an immediately connected document. We treat the graph as undirected.

Figure 6 shows a subset of the MAP estimate of the clustering under these assumptions. Note that the clusters form connected groups of documents, though several clusters are possible within a large connected group. Traditional CRP clustering does not lean towards such solutions. Overall, the distance dependent CRP provides a better model. The log Bayes factor is 13,062, strongly in favor of the distance dependent CRP, although we emphasize that much of this improvement may occur simply because the distance dependent CRP avoids clustering abstracts from unconnected components of the network. Further analysis is needed to understand the abilities of the distance dependent CRP beyond those of simpler network-aware clustering schemes.

We emphasize that this analysis is meant to be a proof of concept to demonstrate the flexibility of distance dependent CRP mixtures. Many modeling choices can be explored, including longer windows in the decay function and treating the graph as a directed graph. A similar modeling set-up could be used to analyze spatial data, where distances are natural to compute, or images (e.g., for image segmentation), where distances might be the Manhattan distance between pixels.

5.4 Comparison to the Traditional Gibbs Sampler

The distance dependent CRP can express a number of flexible models. However, as we describe in Section 2, it can also re-express the traditional CRP. In the mixture model setting, the Gibbs sampler of Section 3 thus provides an alternative algorithm for approximate posterior inference in DP mixtures. We compare this Gibbs sampler to the widely used collapsed Gibbs sampler for DP mixtures, that is, Algorithm 3 from Neal (2000), which is applicable when the base measure G_0 is conjugate to the data generating distribution.

The Gibbs sampler for the distance dependent CRP iteratively samples the customer assignment of each data point, while the collapsed Gibbs sampler iteratively samples the cluster assignment of each data point. The practical difference between the two algorithms is that the distance dependent CRP based sampler can change several customers' cluster assignments via a single customer assignment. This allows for larger moves in the state space of the posterior and, we will see below, faster mixing of the sampler.

Moreover, the computational complexity of the two samplers is the same. Both require computing the change in likelihood of adding or removing either a set of points (in the distance dependent CRP case) or a single point (in the traditional CRP case) to each cluster. Whether adding or removing one or a set of points, this amounts to computing a ratio of normalizing constants for each cluster, and this is where the bulk of the computation of each sampler lies.[9]

9. In some settings, removing a single point—as is done in Neal (2000)—allows faster computation of each sampler iteration. This is true, for example, if the observations are single words (as opposed to a document of words) or single draws from a Gaussian. Although each iteration may be faster with the traditional sampler, that sampler may spend many more iterations stuck in local optima.

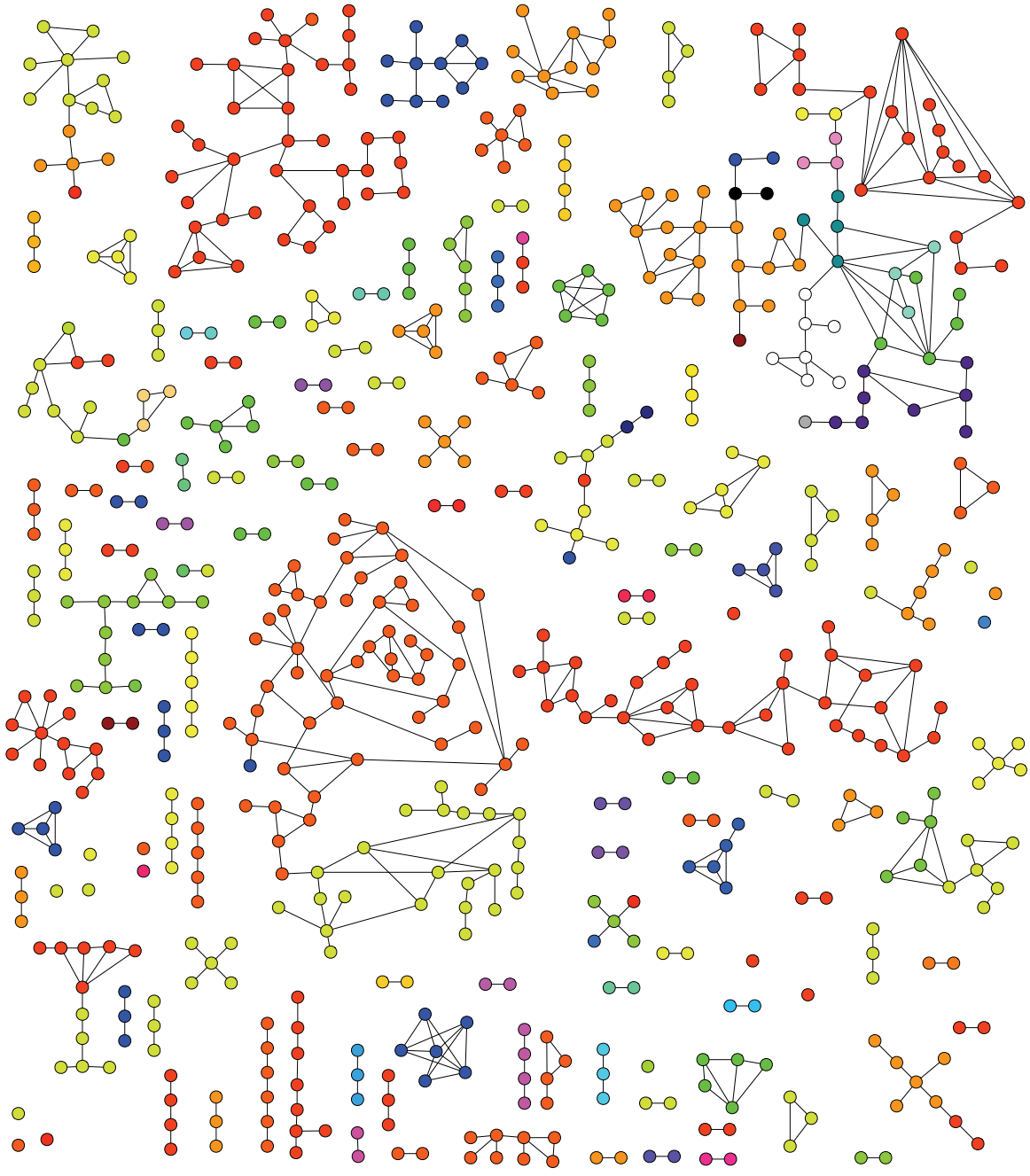


Figure 6: The MAP clustering of a subset of CORA. Each node is an abstract in the collection and each link represents a citation. Colors are repeated across connected components—no two data points from disconnected components in the graph can be assigned to the same cluster. Within each connected component, colors are not repeated, and nodes with the same color are assigned to the same cluster.

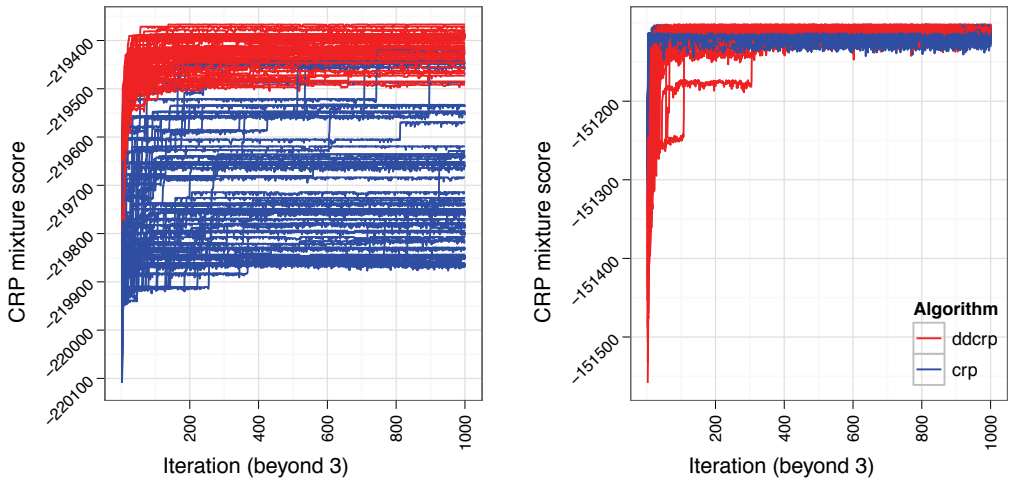


Figure 7: Each panel illustrates 100 Gibbs runs using Algorithm 3 of Neal (2000) (CRP, in blue) and the sampler from Section 3 with the identity decay function (distance dependent CRP, in red). Both samplers have the same limiting distribution because the distance dependent CRP with identity decay is the traditional CRP. We plot the log probability of the CRP representation (i.e., the divergence) as a function of its iteration. The left panel shows the *Science* corpus, and the right panel shows the *New York Times* corpus. Higher values indicate that the chain has found a better local mode of the posterior. In these examples, the distance dependent CRP Gibbs sampler mixes faster.

To compare the samplers, we analyzed documents from the *Science* and *New York Times* collections under a CRP mixture with scaling parameter equal to one and uniform Dirichlet base measure. Figure 7 illustrates the log probability of the state of the traditional CRP Gibbs sampler as a function of Gibbs sampler iteration. The log probability of the state is proportional to the posterior; a higher value indicates a state with higher posterior likelihood. These numbers are comparable because the models, and thus the normalizing constant, are the same for both the traditional representation and customer based CRP. Iterations 3–1000 are plotted, where each sampler is started at the same (random) state. The traditional Gibbs sampler is much more prone to stagnation at local optima, particularly for the *Science* corpus.

6. Discussion

We have developed the distance dependent Chinese restaurant process, a distribution over partitions that accommodates a flexible and non-exchangeable seating assignment distribution. The distance dependent CRP hinges on the customer assignment representation. We derived a general-purpose Gibbs sampler based on this representation, and examined sequential models of text.

The distance dependent CRP opens the door to a number of further developments in infinite clustering models. We plan to explore spatial dependence in models of natural images, and multi-level models akin to the hierarchical Dirichlet process (Teh et al., 2006). Moreover, the simplicity

and fixed dimensionality of the corresponding Gibbs sampler suggests that a variational method is worth exploring as an alternative deterministic form of approximate inference.

Acknowledgments

David M. Blei is supported by ONR 175-6343, NSF CAREER 0745520, AFOSR 09NL202, the Alfred P. Sloan foundation, and a grant from Google. Peter I. Frazier is supported by AFOSR YIP FA9550-11-1-0083. Both authors thank the three anonymous reviewers for their insightful comments and suggestions.

Appendix A. A Formal Characterization of Marginal Invariance

In this section, we formally characterize the class of distance dependent CRPs that are marginally invariant. This family is a very small subset of the entire set of distance dependent CRPs, containing only the traditional CRP and variants constructed from independent copies of it. This characterization is used in Section 4 to contrast the distance dependent CRP with random-measure models.

Throughout this section, we assume that the decay function satisfies a relaxed version of the triangle inequality, which uses the notation $d_{ij} = \min(d_{ij}, d_{ji})$. We assume: if $d_{ij} = 0$ and $d_{jk} = 0$ then $d_{ik} = 0$; and if $d_{ij} < \infty$ and $d_{jk} < \infty$ then $d_{ik} < \infty$.

A.1 Sequential Distances

We first consider sequential distances. We begin with the following proposition, which shows that a very restricted class of distance dependent CRPs may also be constructed by collections of independent CRPs.

Proposition 1 *Fix a set of sequential distances between each of n customers, a real number $a > 0$, and a set $A \in \{\emptyset, \{0\}, \mathbb{R}\}$. Then there is a (non-random) partition $B_1, \ldots, B_K$ of $\{1, \ldots, n\}$ for which two distinct customers i and j are in the same set B_k iff $d_{ij} \in A$. For each $k = 1, \ldots, K$, let there be an independent CRP with concentration parameter α/a, and let customers within B_k be clustered among themselves according to this CRP.*

Then, the probability distribution on clusters induced by this construction is identical to the distance dependent CRP with decay function $f(d) = a1[d \in A]$. Furthermore, this probability distribution is marginally invariant.

Proof We begin by constructing a partition $B_1, \ldots, B_K$ with the stated property. Let $J(i) = \min\{j : j = i \text{ or } d_{ij} \in A\}$, and let $\mathcal{J} = \{J(i) : i = 1, \ldots, n\}$ be the set of unique values taken by J. Each customer i will be placed in the set containing customer $J(i)$. Assign to each value $j \in \mathcal{J}$ a unique integer $k(j)$ between 1 and $|\mathcal{J}|$. For each $j \in \mathcal{J}$, let $B_{k(j)} = \{i : J(i) = j\} = \{i : i = j \text{ or } d_{ij} \in A\}$. Each customer i is in exactly one set, $B_{k(J(i))}$, and so $B_1, \ldots, B_{|\mathcal{J}|}$ is a partition of $\{1, \ldots, n\}$.

To show that $i \neq i'$ are both in B_k iff $d_{ii'} \in A$, we consider two possibilities. If $A = \emptyset$, then $J(i) = i$ and each B_k contains only a single point. If $A = \{0\}$ or $A = \mathbb{R}$, then it follows from the relaxed triangle inequality assumed at the beginning of Appendix A.

With this partition $B_1, \ldots, B_K$, the probability of linkage under the distance dependent CRP with decay function $f(d) = a1[d \in A]$ may be written

$$p(c_i = j) \propto \begin{cases} \alpha & \text{if } i = j, \\ a & \text{if } j < i \text{ and } j \in B_{k(i)}, \\ 0 & \text{if } j > i \text{ or } j \notin B_{k(i)}. \end{cases}$$

By noting that linkages between customers from different sets B_k occur with probability 0, we see that this is the same probability distribution produced by taking K independent distance dependent CRPs, where the kth distance dependent CRP governs linkages between customers in B_k using

$$p(c_i = j) \propto \begin{cases} \alpha & \text{if } i = j, \\ a & \text{if } j < i, \\ 0 & \text{if } j > i, \end{cases}$$

for $i, j \in B_k$.

Finally, dividing the unnormalized probabilities by a, we rewrite the linkage probabilities for the kth distance dependent CRP as

$$p(c_i = j) \propto \begin{cases} \alpha/a & \text{if } i = j, \\ 1 & \text{if } j < i, \\ 0 & \text{if } j > i, \end{cases}$$

for $i, j \in B_k$. This is identical to the distribution of the traditional CRP with concentration parameter α/a.

This shows that the distance dependent CRP with decay function $f(d) = a1[d \in A]$ induces the same probability distribution on clusters as the one produced by a collection of K independent traditional CRPs, each with concentration parameter α/a, where the kth traditional CRP governs the clusters of customers within B_k.

The marginal invariance of this distribution follows from the marginal invariance of each traditional CRP, and their independence from one another. ■

The probability distribution described in this proposition separates customers into groups $B_1, \ldots, B_K$ based on whether inter-customer distances fall within the set A, and then governs clustering within each group independently using a traditional CRP. Clustering across groups does not occur.

We consider what this means for specific choices of A. If $A = \{0\}$, then each group contains those customers whose distance from one another is 0. This group is well-defined because of the assumption that $d_{ij} = 0$ and $d_{jk} = 0$ implies $d_{ik} = 0$. If $A = \mathbb{R}$, then each group contains those customers whose distance from one another is finite. Similarly to the $A = \{0\}$ case, this group is well-defined because of the assumption that $d_{ij} < \infty$ and $d_{jk} < \infty$ implies $d_{ik} < \infty$. If $A = \emptyset$, then each group contains only a single customer. In this case, each customer will be in his own cluster.

Since the resulting construction is marginally invariant, Proposition 1 provides a sufficient condition for marginal invariance. The following proposition shows that this condition is necessary as well.

Proposition 2 *If the distance dependent CRP for a given decay function f is marginally invariant over all sets of sequential distances then f is of the form $f(d) = a1[d \in A]$ for some $a > 0$ and A equal to either $\emptyset$, $\{0\}$, or $\mathbb{R}$.*

Proof Consider a setting with 3 customers, in which customer 2 may either be absent, or present with his seating assignment marginalized out. Fix a non-increasing decay function f with $f(\infty) = 0$ and suppose that the distances are sequential, so $d_{13} = d_{23} = d_{12} = \infty$. Suppose that the distance dependent CRP resulting from this f and any collection of sequential distances is marginally invariant. Then the probability that customers 1 and 3 share a table must be the same whether customer 2 is absent or present.

If customer 2 is absent,

$$\mathbb{P}\{1 \text{ and } 3 \text{ sit at same table} \mid 2 \text{ absent}\} = \frac{f(d_{31})}{f(d_{31}) + \alpha}. \tag{4}$$

If customer 2 is present, customers 1 and 3 may sit at the same table in two different ways: 3 sits with 1 directly ($c_3 = 1$); or 3 sits with 2, and 2 sits with 1 ($c_3 = 2$ and $c_2 = 1$). Thus,

$$\mathbb{P}\{1 \text{ and } 3 \text{ sit at same table} \mid 2 \text{ present}\} = \frac{f(d_{31})}{f(d_{31}) + f(d_{32}) + \alpha} + \left(\frac{f(d_{32})}{f(d_{31}) + f(d_{32}) + \alpha}\right)\left(\frac{f(d_{21})}{f(d_{21}) + \alpha}\right). \tag{5}$$

For the distance dependent CRP to be marginally invariant, Equation (4) and Equation (5) must be identical. Writing Equation (4) on the left side and Equation (5) on the right, we have

$$\frac{f(d_{31})}{f(d_{31}) + \alpha} = \frac{f(d_{31})}{f(d_{31}) + f(d_{32}) + \alpha} + \left(\frac{f(d_{32})}{f(d_{31}) + f(d_{32}) + \alpha}\right)\left(\frac{f(d_{21})}{f(d_{21}) + \alpha}\right). \tag{6}$$

We now consider two different possibilities for the distances d_{32} and d_{21}, always keeping $d_{31} = d_{21} + d_{32}$.

First, suppose $d_{21} = 0$ and $d_{32} = d_{31} = d$ for some $d \geq 0$. By multiplying Equation (6) through by $(2f(d) + \alpha)(f(0) + \alpha)(f(d) + \alpha)$ and rearranging terms, we obtain

$$0 = \alpha f(d)(f(0) - f(d)).$$

Thus, either $f(d) = 0$ or $f(d) = f(0)$. Since this is true for each $d \geq 0$ and f is nonincreasing, $f = a1[d \in A]$ with $a \geq 0$ and either $A = \emptyset$, $A = \mathbb{R}$, $A = [0, b]$, or $A = [0, b)$ with $b \in [0, \infty)$. Because $A = \emptyset$ is among the choices, we may assume $a > 0$ without loss of generality. We now show that if $A = [0, b]$ or $A = [0, b)$, then we must have $b = 0$ and A is of the form claimed by the proposition.

Suppose for contradiction that $A = [0, b]$ or $A = [0, b)$ with $b > 0$. Consider distances given by $d_{32} = d_{21} = d = b - \varepsilon$ with $\varepsilon \in (0, b/2)$. By multiplying Equation (5) through by

$$(f(2d) + f(d) + \alpha)(f(d) + \alpha)(f(2d) + \alpha)$$

and rearranging terms, we obtain

$$0 = \alpha f(d)(f(d) - f(2d)).$$

Since $f(d) = a > 0$, we must have $f(2d) = f(d) > 0$. But, $2d = 2(b-\varepsilon) > b$ implies together with $f(2d) = a1[2d \in A]$ that $f(2d) = 0$, which is a contradiction. ■

These two propositions are combined in the following corollary, which states that the class of decay functions considered in Propositions 1 and 2 is both necessary and sufficient for marginal invariance.

Corollary 3 *Fix a particular decay function f. The distance dependent CRP resulting from this decay function is marginally invariant over all sequential distances if and only if f is of the form $f(d) = a1[d \in A]$ for some $a > 0$ and some $A \in \{\emptyset, \{0\}, \mathbb{R}\}$.*

Proof Sufficiency for marginal invariance is shown by Proposition 1. Necessity is shown by Proposition 2. ■

Although Corollary 3 allows any choice of $a > 0$ in the decay function $f(d) = a1[d \in A]$, the distribution of the distance dependent CRP with a particular f and α remains unchanged if both f and α are multiplied by a constant factor (see Equation (2)). Thus, the distance dependent CRP defined by $f(d) = a1[d \in A]$ and concentration parameter α is identical to the one defined by $f(d) = 1[d \in A]$ and concentration parameter α/a. In this sense, we can restrict the choice of a in Corollary 3 (and also Propositions 1 and 2) to $a = 1$ without loss of generality.

A.2 General Distances

We now consider all sets of distances, including non-sequential distances. The class of distance dependent CRPs that are marginally invariant over this larger class of distances is even more restricted than in the sequential case. We have the following proposition providing a necessary condition for marginal invariance.

Proposition 4 *If the distance dependent CRP for a given decay function f is marginally invariant over all sets of distances, both sequential and non-sequential, then f is identically 0.*

Proof From Proposition 2, we have that any decay function that is marginally invariant under all sequential distances must be of the form $f(d) = a1[d \in A]$, where $a > 0$ and $A \in \{\emptyset, \{0\}, \mathbb{R}\}$. We now show that if the decay function is marginally invariant under *all* sets of distances (not just those that are sequential), then $f(0) = 0$. The only decay function of the form $f(d) = a1[d \in A]$ that satisfies $f(0) = 0$ is the one that is identically 0, and so this will show our result.

To show $f(0) = 0$, suppose that we have $n+1$ customers, all of whom are a distance 0 away from one another, so $d_{ij} = 0$ for $i, j = 1, \ldots, n+1$. Under our assumption of marginal invariance, the probability that the first n customers sit at separate tables should be invariant to the absence or presence of customer $n+1$.

When customer $n+1$ is absent, the only way in which the first n customers may sit at separate tables is for each to link to himself. Let $p_n = \alpha/(\alpha + (n-1)f(0))$ denote the probability of a given customer linking to himself when customer $n+1$ is absent. Then

$$\mathbb{P}\{1, \ldots, n \text{ sit separately} \mid n+1 \text{ absent}\} = (p_n)^n. \tag{7}$$

We now consider the case when customer $n+1$ is present. Let $p_{n+1} = \alpha/(\alpha + nf(0))$ be the probability of a given customer linking to himself, and let $q_{n+1} = f(0)/(\alpha + nf(0))$ be the probability of a given customer linking to some other given customer. The first n customers may each sit at separate tables in two different ways. First, each may link to himself, which occurs with probability $(p_{n+1})^n$. Second, all but one of these first n customers may link to himself, with the remaining customer linking to customer $n+1$, and customer $n+1$ linking either to himself or to the customer that linked to him. This occurs with probability $n(p_{n+1})^{n-1}q_{n+1}(p_{n+1}+q_{n+1})$. Thus, the total probability that the first n customers sit at separate tables is

$$\mathbb{P}\{1,\ldots,n \text{ sit separately} \mid n+1 \text{ present}\} = (p_{n+1})^n + n(p_{n+1})^{n-1}q_{n+1}(p_{n+1}+q_{n+1}). \quad (8)$$

Under our assumption of marginal invariance, Equation (7) must be equal to Equation (8), and so

$$0 = (p_{n+1})^n + n(p_{n+1})^{n-1}q_{n+1}(p_{n+1}+q_{n+1}) - (p_n)^n. \quad (9)$$

Consider $n = 2$. By substituting the definitions of p_2, p_3, and q_3, and then rearranging terms, we may rewrite Equation (9) as

$$0 = \frac{\alpha f(0)^2(2f(0)^2 - \alpha^2)}{(\alpha + f(0))^2(\alpha + 2f(0))^3},$$

which is satisfied only when $f(0) \in \{0, \alpha/\sqrt{2}\}$. Consider the second of these roots, $\alpha/\sqrt{2}$. When $n = 3$, this value of $f(0)$ violates Equation (9). Thus, the first root is the only possibility and we must have $f(0) = 0$.

■

The decay function $f = 0$ described in Proposition 4 is a special case of the decay function from Proposition 2, obtained by taking $A = \emptyset$. As described above, the resulting probability distribution is one in which each customer links to himself, and is thus clustered by himself. This distribution is marginally invariant. From this observation quickly follows the following corollary.

Corollary 5 *The decay function $f = 0$ is the only one for which the resulting distance dependent CRP is marginally invariant over all distances, both sequential and non-sequential.*

Proof Necessity of $f = 0$ for marginal invariance follows from Proposition 4. Sufficiency follows from the fact that the probability distribution on partitions induced by $f = 0$ is the one under which each customer is clustered alone almost surely, which is marginally invariant. ■

Appendix B. Gibbs Sampling for the Hyperparameters

To enhance our models, we place a prior on the concentration parameter α and augment our Gibbs sampler accordingly, just as is done in the traditional CRP mixture (Escobar and West, 1995). To sample from the posterior of α given the customer assignments $\mathbf{c}$ and data, we begin by noting that α is conditionally independent of the observed data given the customer assignments. Thus, the quantity needed for sampling is

$$p(\alpha \mid \mathbf{c}) \propto p(\mathbf{c} \mid \alpha)p(\alpha),$$

where $p(\alpha)$ is a prior on the concentration parameter.

From the independence of the c_i under the generative process, $p(\mathbf{c}\,|\,\alpha) = \prod_{i=1}^{N} p(c_i\,|\,D,\alpha)$. Normalizing provides

$$\begin{aligned} p(\mathbf{c}\,|\,\alpha) &= \prod_{i=1}^{N} \frac{1[c_i = i]\alpha + 1[c_i \neq i] f(d_{ic_i})}{\alpha + \sum_{j\neq i} f(d_{ij})} \\ &\propto \alpha^K \left[\prod_{i=1}^{N} \left(\alpha + \sum_{j\neq i} f(d_{ij})\right)\right]^{-1}, \end{aligned}$$

where K is the number of self-links $c_i = i$ in the customer assignments $\mathbf{c}$. Although K is equal to the number of tables $|z(\mathbf{c})|$ when distances are sequential, K and $|z(\mathbf{c})|$ generally differ when distances are non-sequential. Then,

$$p(\alpha\,|\,\mathbf{c}) \propto \alpha^K \left[\prod_{i=1}^{N} \left(\alpha + \sum_{j\neq i} f(d_{ij})\right)\right]^{-1} p(\alpha). \tag{10}$$

Equation (10) reduces further in the following special case: f is the window decay function, $f(d) = 1[d < a]$; $d_{ij} = i - j$ for $i > j$; and distances are sequential so $d_{ij} = \infty$ for $i < j$. In this case, $\sum_{j=1}^{i-1} f(d_{ij}) = (i-1) \wedge (a-1)$, where $\wedge$ is the minimum operator, and

$$\prod_{i=1}^{N} \left(\alpha + \sum_{j=1}^{i-1} f(d_{ij})\right) = (\alpha + a - 1)^{[N-a]^+} \Gamma(\alpha + a \wedge N)/\Gamma(\alpha), \tag{11}$$

where $[N-a]^+ = \max(0, N-a)$ is the positive part of $N-a$. Then,

$$p(\alpha\,|\,\mathbf{c}) \propto \frac{\Gamma(\alpha)}{\Gamma(\alpha + a \wedge N)} \frac{\alpha^K}{(\alpha + a - 1)^{[N-a]^+}} p(\alpha).$$

If we use the identity decay function, which results in the traditional CRP, then we recover an expression from Antoniak (1974): $p(\alpha\,|\,\mathbf{c}) \propto \frac{\Gamma(\alpha)}{\Gamma(\alpha+N)} \alpha^K p(\alpha)$. This expression is used in Escobar and West (1995) to sample exactly from the posterior of α when the prior is gamma distributed.

In general, if the prior on α is continuous then it is difficult to sample exactly from the posterior of Equation (10). There are a number of ways to address this. We may, for example, use the Griddy-Gibbs method (Ritter and Tanner, 1992). This method entails evaluating Equation (10) on a finite set of points, approximating the inverse cdf of $p(\alpha\,|\,\mathbf{c})$ using these points, and transforming a uniform random variable with this approximation to the inverse cdf.

We may also sample over any hyperparameters in the decay function used (e.g., the window size in the window decay function, or the rate parameter in the exponential decay function) within our Gibbs sampler. For the rest of this section, we use a to generically denote a hyperparameter in the decay function, and we make this dependence explicit by writing $f(d,a)$.

To describe Gibbs sampling over these hyperparameters in the decay function, we first write

$$\begin{aligned} p(\mathbf{c}\mid \alpha, a) &= \prod_{i=1}^{N} \frac{1[c_i = i]\alpha + 1[c_i \neq i] f(d_{ic_i}, a)}{\alpha + \sum_{j=1}^{i-1} f(d_{ij}, a)} \\ &= \alpha^K \left[\prod_{i:c_i\neq i} f(d_{ij}, a)\right] \left[\prod_{i=1}^{N} \left(\alpha + \sum_{j=1}^{i-1} f(d_{ij}, a)\right)\right]^{-1}. \end{aligned}$$

Since a is conditionally independent of the observed data given $\mathbf{c}$ and α, to sample over a in our Gibbs sampler it is enough to know the density

$$p(a \mid \mathbf{c}, \alpha) \propto \left[\prod_{i:c_i \neq i} f(d_{ij}, a)\right] \left[\prod_{i=1}^{N} \left(\alpha + \sum_{j=1}^{i-1} f(d_{ij}, a)\right)\right]^{-1} p(a \mid \alpha). \tag{12}$$

In many cases our prior $p(a \mid \alpha)$ on a will not depend on α.

In the case of the window decay function with sequential distances and $d_{ij} = i - j$ for $i > j$, we can simplify this further as we did above with Equation (11). Noting that $\prod_{i:c_i \neq i} f(d_{ij}, a)$ will be 1 for those $a > \max_i i - c_i$, and 0 for other a, we have

$$p(a \mid \mathbf{c}, \alpha) \propto \frac{\Gamma(\alpha)}{\Gamma(\alpha + a \wedge N)} \frac{p(a \mid \alpha) 1[a > \max_i i - c_i]}{(\alpha + a - 1)^{[N-a]^+}}.$$

If the prior distribution on a is discrete and concentrated on a finite set, as it might be with the window decay function, one can simply evaluate and normalize Equation (12) on this set. If the prior is continuous, as it might be with the exponential decay function, then it is difficult to sample exactly from Equation (12), but one can again use the Griddy-Gibbs approach of Ritter and Tanner (1992) to sample approximately.

References

A. Ahmed and E. Xing. Dynamic non-parametric mixture models and the recurrent Chinese restaurant process with applications to evolutionary clustering. In *International Conference on Data Mining*, 2008.

C. Antoniak. Mixtures of Dirichlet processes with applications to Bayesian nonparametric problems. *The Annals of Statistics*, 2(6):1152–1174, 1974.

D. Blackwell. Discreteness of Ferguson selections. *The Annals of Statistics*, 1(2):356–358, 1973.

D. Blei and P. Frazier. Distance dependent Chinese restaurant processes. In *International Conference on Machine Learning*, 2010.

D. Blei and M. Jordan. Variational inference for Dirichlet process mixtures. *Journal of Bayesian Analysis*, 1(1):121–144, 2005.

D. Blei, T. Griffiths, and M. Jordan. The nested Chinese restaurant process and Bayesian nonparametric inference of topic hierarchies. *Journal of the ACM*, 57(2):1–30, 2010.

D.B. Dahl. Distance-based probability distribution for set partitions with applications to Bayesian nonparametrics. In *JSM Proceedings. Section on Bayesian Statistical Science, American Statistical Association, Alexandria, Va*, 2008.

H. Daume. Fast search for Dirichlet process mixture models. In *Artificial Intelligence and Statistics*, San Juan, Puerto Rico, 2007. URL `http://pub.hal3.name/#daume07astar-dp`.

J. Duan, M. Guindani, and A. Gelfand. Generalized spatial Dirichlet process models. *Biometrika*, 94:809–825, 2007.

D. Dunson. Bayesian dynamic modeling of latent trait distributions. *Biostatistics*, 2006.

D. Dunson, N. Pillai, and J. Park. Bayesian density regression. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 69(2):163–183, 2007.

M. Escobar and M. West. Bayesian density estimation and inference using mixtures. *Journal of the American Statistical Association*, 90:577–588, 1995.

T. Ferguson. A Bayesian analysis of some nonparametric problems. *The Annals of Statistics*, 1: 209–230, 1973.

E. Fox, E. Sudderth, M. Jordan, and A. Willsky. Developing a tempered HDP-HMM for systems with state persistence. Technical report, MIT Laboratory for Information and Decision Systems, 2007.

C. Geyer and E. Thompson. Constrained Monte Carlo maximum likelihood for dependent data. *Journal of the American Statistical Association*, 54(657–699), 1992.

S. Goldwater, T. Griffiths, and M. Johnson. Interpolating between types and tokens by estimating power-law generators. In *Neural Information Processing Systems*, 2006.

J. Griffin and M. Steel. Order-based dependent Dirichlet processes. *Journal of the American Statistical Association*, 101(473):179–194, 2006.

J.A. Hartigan. Partition models. *Communications in Statistics-Theory and Methods*, 19(8):2745–2756, 1990.

M. Johnson, T. Griffiths, and Goldwater S. Adaptor grammars: A framework for specifying compositional nonparametric Bayesian models. In B. Schölkopf, J. Platt, and T. Hoffman, editors, *Advances in Neural Information Processing Systems 19*, pages 641–648, Cambridge, MA, 2007. MIT Press.

R. Kass and A. Raftery. Bayes factors. *Journal of the American Statistical Association*, 90(430): 773–795, 1995.

W. Li, D. Blei, and A. McCallum. Nonparametric Bayes pachinko allocation. In *The 23rd Conference on Uncertainty in Artificial Intelligence*, 2007.

P. Liang, M. Jordan, and B. Taskar. A permutation-augmented sampler for DP mixture models. In *International Conference on Machine Learning*, 2007.

S. MacEachern. Dependent nonparametric processes. In *ASA Proceedings of the Section on Bayesian Statistical Science*, 1999.

A. McCallum, K. Nigam, J. Rennie, and K. Seymore. Automating the construction of internet portals with machine learning. *Information Retrieval*, 2000.

K.T. Miller, T.L. Griffiths, and M.I. Jordan. The phylogenetic indian buffet process: A non-exchangeable nonparametric prior for latent features. In David A. McAllester and Petri Myllymäki, editors, *UAI*, pages 403–410. AUAI Press, 2008.

P. Mueller and F. Quintana. Random partition models with regression on covariates. In *International Conference on Interdisciplinary Mathematical and Statistical Techniques*, 2008.

P. Muller, F. Quintana, and G. Rosner. Bayesian clustering with regression. Working paper, 2008.

R. Neal. Probabilistic inference using Markov chain Monte Carlo methods. Technical Report CRG-TR-93-1, Department of Computer Science, University of Toronto, 1993.

R. Neal. Markov chain sampling methods for Dirichlet process mixture models. *Journal of Computational and Graphical Statistics*, 9(2):249–265, 2000.

J. Pitman. *Combinatorial Stochastic Processes*. Lecture Notes for St. Flour Summer School. Springer-Verlag, New York, NY, 2002.

C. Rasmussen and Z. Ghahramani. Infinite mixtures of Gaussian process experts. In T. Dietterich, S. Becker, and Z. Ghahramani, editors, *Advances in Neural Information Processing Systems 14*, Cambridge, MA, 2002. MIT Press.

C. Ritter and M. Tanner. Facilitating the Gibbs sampler: The Gibbs stopper and the Griddy-Gibbs sampler. *Journal of the American Statistical Association*, 87(419):861–868, 1992.

C. Robert and G. Casella. *Monte Carlo Statistical Methods*. Springer Texts in Statistics. Springer-Verlag, New York, NY, 2004.

E. Sudderth, A. Torralba, W. Freeman, and A. Willsky. Describing visual scenes using transformed Dirichlet processes. In *Advances in Neural Information Processing Systems 18*, 2005.

E.B. Sudderth and M. I. Jordan. Shared segmentation of natural scenes using dependent pitman-yor processes. In Daphne Koller, Dale Schuurmans, Yoshua Bengio, and Léon Bottou, editors, *NIPS*, pages 1585–1592. MIT Press, 2008.

Y. Teh. A hierarchical Bayesian language model based on Pitman-Yor processes. In *Proceedings of the Association of Computational Linguistics*, 2006.

Y. Teh, M. Jordan, M. Beal, and D. Blei. Hierarchical Dirichlet processes. *Journal of the American Statistical Association*, 101(476):1566–1581, 2006.

E. Xing, M. Jordan, and R. Sharan. Bayesian haplotype inference via the Dirichlet process. *Journal of Computational Biology*, 14(3):267–284, 2007.

Y. Xue, D. Dunson, and L. Carin. The matrix stick-breaking process for flexible multi-task learning. In *International Conference on Machine Learning*, 2007.

X. Zhu, Z. Ghahramani, and J. Lafferty. Time-sensitive Dirichlet process mixture models. Technical Report CMU-CALD-05-104, Carnegie Mellon University, 2005.

Journal of Machine Learning Research 12 (2011) 2489-2492 Submitted 4/11; Revised 7/11; Published 8/11

LPmade: Link Prediction Made Easy

Ryan N. Lichtenwalter RLICHTEN@ND.EDU
Nitesh V. Chawla NCHAWLA@ND.EDU
Department of Computer Science
University of Notre Dame
Notre Dame, IN 46556, USA

Editor: Geoff Holmes

Abstract

LPmade is a complete cross-platform software solution for multi-core link prediction and related tasks and analysis. Its first principal contribution is a scalable network library supporting high-performance implementations of the most commonly employed unsupervised link prediction methods. Link prediction in longitudinal data requires a sophisticated and disciplined procedure for correct results and fair evaluation, so the second principle contribution of LPmade is a sophisticated GNU `make` architecture that completely automates link prediction, prediction evaluation, and network analysis. Finally, LPmade streamlines and automates the procedure for creating multivariate supervised link prediction models with a version of WEKA modified to operate effectively on extremely large data sets. With mere minutes of manual work, one may start with a raw stream of records representing a network and progress through hundreds of steps to generate plots, gigabytes or terabytes of output, and actionable or publishable results.

Keywords: link prediction, network analysis, multicore, GNU make, PropFlow, HPLP

1. Introduction

Link prediction is succinctly stated as the problem of identifying yet-unobserved links in a network. This task is of increasing interest in both research and corporate contexts. Virtually every major conference and journal in data mining or machine learning now has a significant network science component, and these often include treatments of link prediction. Link prediction is of great use in domains ranging from biology to corporate recruiting, but it is a difficult problem for which to develop models because of extreme class imbalance, the longitudinal nature of the data, the difficulties inherent in effective evaluation, and other issues raised by Lichtenwalter et al. (2010). Further, even for standard prediction algorithms, researchers must often write new code or cobble together existing code fragments. The work flow to achieve predictions and fair evaluation is time-consuming, challenging, and error-prone. LPmade is the first library to focus on link prediction specifically, incorporating general and extensible forms of the predictors introduced by Liben-Nowell and Kleinberg (2007). It also streamlines and parameterizes the complex link prediction work flow so that researchers can start with source data and achieve predictions in minimal time.

There is no shortage of graph libraries: the Boost Graph Library, SNAP, igraph, JGraphT, GraphCrunch, GOBLIN, and many others. Some offer extreme generality, some offer extreme efficiency, some offer modeling utilities, and some have a dizzying array of algorithms. LPmade is not just yet another graph library. Its software components are, by necessity, designed for high performance, and it offers a wide array of graph analysis algorithms, but it is first and foremost

an extensive toolkit for performing link prediction to achieve both research and application goals. Unlike other options, LPmade provides an organized collection of link prediction algorithms in a build framework that is accessible to researchers across many disciplines. The software is available at `http://mloss.org/software/view/307/`.

2. The Software Package

The purpose of LPmade is to provide a workbench on which others may conduct link prediction research and applications. For link prediction tasks in many large networks even a restricted set of predictions may involve millions, billions, or even trillions of lines of output. Each unsupervised link prediction method, the supervised classification framework from Lichtenwalter et al. (2010), and all the evaluation tools are optimized for just such quantities of data. Nonetheless, the entire process of starting from raw source data and ending with predictions, evaluations, and plots involves an extensive series of steps that may each take a long time. The software includes a carefully constructed dependency tracking system that minimizes overhead and simplifies the management of correct procedures. Both the build system and the link prediction library are modular and extensible. Researchers can incorporate their own prediction methods into the library and the automation framework just by writing a C++ class and changing a `make` variable.

2.1 Network Library

The LPmade network library is written entirely in scalable, high-performance C/C++ that minimizes memory consumption with a compact adjacency list format based on a vector-of-vectors to represent edges and a translation vector to associate external vertex names to internal identifiers. The library includes clearly written yet optimized versions of the most common asymptotically optimal network analysis algorithms for sampling, finding connected components, computing centrality measures, and calculating useful statistics.

LPmade specializes in link prediction by including commonly used unsupervised link prediction methods: Adamic/Adar, common neighbors, Jaccard's coefficient, Katz, preferential attachment, PropFlow, rooted PageRank, SimRank, and weighted rooted PageRank. The library also has some simpler methods useful in producing feature vectors for supervised learners: clustering coefficient, geodesic distance, degree, PageRank, volume or gregariousness, mutuality, path count, and shortest path count. These methods may be selectively incorporated as features into the supervised framework by Lichtenwalter et al. (2010).

Several graph libraries such as the Boost Graph Library are brilliantly designed for maximum generality and flexibility with template parameters and complex inheritance models. One minor drawback to such libraries is that the code is complex to read and modify. The code base for this library takes a narrower approach by offering fewer mechanisms for generality, but as a result it has a much shallower learning curve.

2.2 GNU `make` Script and Supporting Tools

Although it can be used and extended as such, LPmade is not just a library of C++ code for network analysis and link prediction. It is additionally an extensive set of scripts designed for sophisticated automation and dependency resolution. These scripts are all incorporated into a set of 2 co-dependent Makefiles: task-specific and common. Each new raw data set requires its own task-

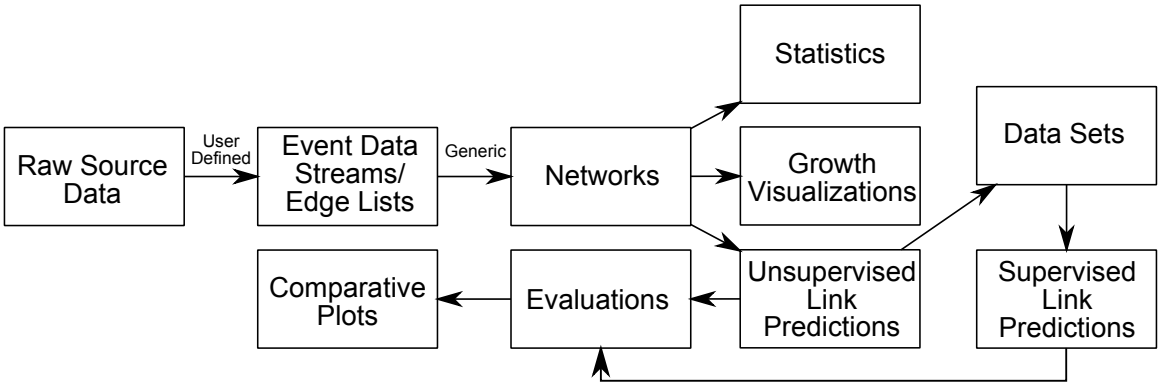


Figure 1: A simplified depiction of some of the build paths in the automation script. Only the first transition is user-defined. Each step involves multiple invocations of many programs to properly assemble data and perform fair evaluation.

specific Makefile, which generally requires less than 20 lines of user code. This Makefile is where users specify the manner in which raw source data is converted to the initial data stream required by subsequent steps in the pipeline. It is also where rules from the common Makefile can be overridden for task-specific reasons. The common Makefile, Makefile.common, includes all the general rules that apply to any network analysis or link prediction task once the task-specific Makefile is written to enable proper handling of raw input. The common Makefile script is designed with advanced template features that allow `make` to modify original Makefile rules in accordance with user requirements. Logical tasks are aggressively provided with their own rules so that the multi-core features of GNU `make` are of optimal benefit. In general, users need not be familiar with writing Makefiles. The important options for the behavior of the automatic build system are presented at the top of the common Makefile along with documentation. For instance, to predict within the 2nd and 3rd degree neighborhoods, set `NEIGHBORHOOD := 2 3`.

Figure 1 illustrates some simplified build paths, and the sample calls below demonstrate several targets with their corresponding actions:

```
make -j 28 sm  # using 28 cores, build a data stream from source, generate required networks, run predictors, and perform evaluations
make -j 8 stats  # using 8 cores, compute several network statistics on the complete network represented by the entire data set
make classify  # construct data sets then use parameters specified in Makefile to train, test, and evaluate
make -j 6 growth  # using 6 cores, generate growth information and plots to describe network saturation
```

Parallelism in these cases is all coarse-grained. Each rule in the Makefile script with no outstanding prerequisites is handled by a separate process to make use of additional cores.

For many large networks, link prediction and supporting analysis yields very large output files. When this prolific output is further combined into data sets, both the I/O capacity and bandwidth requirements may be problematic. To combat this, most steps in the work flow create, accept, and output `gzip`-compressed results. Especially on multi-core systems, this results in a hefty decrease in I/O capacity and bandwidth requirements with a minimal impact on performance. In most cases, the output from `gunzip` is produced faster than the consuming process can accept it. Where necessary, named pipes are used to ameliorate potentially large temporary storage requirements.

2.3 WEKA Modifications

LPmade includes a modified version of WEKA 3.5.8 (Witten and Frank, 2005). It is not meant for direct user invocation. Instead the build system uses WEKA classifier implementations to construct

supervised models for link prediction. Unmodified, WEKA has several limitations that make even its command-line mode problematic for operation on enormous link prediction testing sets. These include processing overhead for unwanted computations, Java string overflow and potential thrashing from in-memory result concatenation, and inability to handle compressed C4.5 format input. Alternatives such as MOA solve some but not all of these problems, and WEKA internal classes such as AbstractOutput are unavailable at the command line. We have chosen to modify the WEKA command-line evaluation path to compute only the necessary information and to output directly to standard output for LPmade scripted downstream processing. We have integrated support for processing `gzip`-compressed C4.5 input and use this support in the build system to take advantage of significant space savings on disk.

3. Documentation and Requirements

LPmade comes with `man` pages and a PDF user manual that describes all aspects of the software, most notably describing the setup process, how to use or extend the raw network library, and how to leverage the existing build system to complete many complex steps with short commands. The network library includes an easily extended testing architecture for testing and verification of individual binaries.

The C++ library is written in platform-independent C++ code using only STL extensions. The library may thus be built on any architecture and any operating system that provides a C++ compiler. An included set of high-speed evaluation tools is written in C99 and builds on any system with such a compiler. The bundled distribution of WEKA is cross-platform but requires version 1.5 or higher of the JRE. The automated build system requires GNU `make`. The common Makefile additionally employs many standard tools such as `cut`, `paste`, `sed`, `awk`, `perl`, `sort`, `gzip`, and bundled `gnuplot` 4.4.3.

Acknowledgments

Research was sponsored in part by the Army Research Laboratory under Cooperative Agreement Number W911NF-09-2-0053 and in part by the National Science Foundation Grant BCS-0826958.

References

David Liben-Nowell and Jon Kleinberg. The link-prediction problem for social networks. *Journal of the American Society for Inf. Science and Tech.*, 58(7):1019–1031, 2007.

Ryan N. Lichtenwalter, Jake T. Lussier, and Nitesh V. Chawla. New perspectives and methods in link prediction. In *Proc. of the 16th ACM SIGKDD Intl. Conf. on Knowledge Disc. and Data Mining*, pages 243–252, 2010.

Ian H. Witten and Eibe Frank. *Data Mining: Practical Machine Learning Tools and Techniques*. Morgan Kaufmann, San Francisco, California, USA, second edition, 2005.

Journal of Machine Learning Research 12 (2011) 2493-2537 Submitted 1/10; Revised 11/10; Published 8/11

Natural Language Processing (Almost) from Scratch

Ronan Collobert[*] RONAN@COLLOBERT.COM
Jason Weston[†] JWESTON@GOOGLE.COM
Léon Bottou[‡] LEON@BOTTOU.ORG
Michael Karlen MICHAEL.KARLEN@GMAIL.COM
Koray Kavukcuoglu[§] KORAY@CS.NYU.EDU
Pavel Kuksa[¶] PKUKSA@CS.RUTGERS.EDU
NEC Laboratories America
4 Independence Way
Princeton, NJ 08540

Editor: Michael Collins

Abstract

We propose a unified neural network architecture and learning algorithm that can be applied to various natural language processing tasks including part-of-speech tagging, chunking, named entity recognition, and semantic role labeling. This versatility is achieved by trying to avoid task-specific engineering and therefore disregarding a lot of prior knowledge. Instead of exploiting man-made input features carefully optimized for each task, our system learns internal representations on the basis of vast amounts of mostly unlabeled training data. This work is then used as a basis for building a freely available tagging system with good performance and minimal computational requirements.

Keywords: natural language processing, neural networks

1. Introduction

Will a computer program ever be able to convert a piece of English text into a programmer friendly data structure that describes the meaning of the natural language text? Unfortunately, no consensus has emerged about the form or the existence of such a data structure. Until such fundamental Articial Intelligence problems are resolved, computer scientists must settle for the reduced objective of extracting simpler representations that describe limited aspects of the textual information.

These simpler representations are often motivated by specific applications (for instance, bag-of-words variants for information retrieval), or by our belief that they capture something more general about natural language. They can describe syntactic information (e.g., part-of-speech tagging, chunking, and parsing) or semantic information (e.g., word-sense disambiguation, semantic role labeling, named entity extraction, and anaphora resolution). Text corpora have been manually annotated with such data structures in order to compare the performance of various systems. The availability of standard benchmarks has stimulated research in Natural Language Processing (NLP)

∗. Ronan Collobert is now with the Idiap Research Institute, Switzerland.
†. Jason Weston is now with Google, New York, NY.
‡. Léon Bottou is now with Microsoft, Redmond, WA.
§. Koray Kavukcuoglu is also with New York University, New York, NY.
¶. Pavel Kuksa is also with Rutgers University, New Brunswick, NJ.

and effective systems have been designed for all these tasks. Such systems are often viewed as software components for constructing real-world NLP solutions.

The overwhelming majority of these state-of-the-art systems address their single benchmark task by applying linear statistical models to ad-hoc features. In other words, the researchers themselves discover intermediate representations by engineering task-specific features. These features are often derived from the output of preexisting systems, leading to complex runtime dependencies. This approach is effective because researchers leverage a large body of linguistic knowledge. On the other hand, there is a great temptation to optimize the performance of a system for a specific benchmark. Although such performance improvements can be very useful in practice, they teach us little about the means to progress toward the broader goals of natural language understanding and the elusive goals of Artificial Intelligence.

In this contribution, we try to excel on *multiple benchmarks* while *avoiding task-specific engineering*. Instead we use a *single learning system* able to discover adequate internal representations. In fact we view the benchmarks as indirect measurements of the relevance of the internal representations discovered by the learning procedure, and we posit that these intermediate representations are more general than any of the benchmarks. Our desire to avoid task-specific engineered features prevented us from using a large body of linguistic knowledge. Instead we reach good performance levels in most of the tasks by transferring intermediate representations discovered on large unlabeled data sets. We call this approach "almost from scratch" to emphasize the reduced (but still important) reliance on a priori NLP knowledge.

The paper is organized as follows. Section 2 describes the benchmark tasks of interest. Section 3 describes the unified model and reports benchmark results obtained with supervised training. Section 4 leverages large unlabeled data sets ($\sim$ 852 million words) to train the model on a language modeling task. Performance improvements are then demonstrated by transferring the unsupervised internal representations into the supervised benchmark models. Section 5 investigates multitask supervised training. Section 6 then evaluates how much further improvement can be achieved by incorporating standard NLP task-specific engineering into our systems. Drifting away from our initial goals gives us the opportunity to construct an all-purpose tagger that is simultaneously accurate, practical, and fast. We then conclude with a short discussion section.

2. The Benchmark Tasks

In this section, we briefly introduce four standard NLP tasks on which we will benchmark our architectures within this paper: Part-Of-Speech tagging (POS), chunking (CHUNK), Named Entity Recognition (NER) and Semantic Role Labeling (SRL). For each of them, we consider a standard experimental setup and give an overview of state-of-the-art systems on this setup. The experimental setups are summarized in Table 1, while state-of-the-art systems are reported in Table 2.

2.1 Part-Of-Speech Tagging

POS aims at labeling each word with a unique tag that indicates its *syntactic role*, for example, plural noun, adverb, ...A standard benchmark setup is described in detail by Toutanova et al. (2003). Sections 0–18 of Wall Street Journal (WSJ) data are used for training, while sections 19–21 are for validation and sections 22–24 for testing.

The best POS classifiers are based on classifiers trained on windows of text, which are then fed to a bidirectional decoding algorithm during inference. Features include preceding and following

Task	**Benchmark**	**Data set**	**Training set** (#tokens)	**Test set** (#tokens)	(#tags)
POS	Toutanova et al. (2003)	WSJ	sections 0–18 (912,344)	sections 22–24 (129,654)	(45)
Chunking	CoNLL 2000	WSJ	sections 15–18 (211,727)	section 20 (47,377)	(42) (IOBES)
NER	CoNLL 2003	Reuters	"eng.train" (203,621)	"eng.testb" (46,435)	(17) (IOBES)
SRL	CoNLL 2005	WSJ	sections 2–21 (950,028)	section 23 + 3 Brown sections (63,843)	(186) (IOBES)

Table 1: Experimental setup: for each task, we report the standard benchmark we used, the data set it relates to, as well as training and test information.

System	Accuracy
Shen et al. (2007)	97.33%
Toutanova et al. (2003)	97.24%
Giménez and Màrquez (2004)	97.16%

(a) POS

System	F1
Shen and Sarkar (2005)	95.23%
Sha and Pereira (2003)	94.29%
Kudo and Matsumoto (2001)	93.91%

(b) CHUNK

System	F1
Ando and Zhang (2005)	89.31%
Florian et al. (2003)	88.76%
Kudo and Matsumoto (2001)	88.31%

(c) NER

System	F1
Koomen et al. (2005)	77.92%
Pradhan et al. (2005)	77.30%
Haghighi et al. (2005)	77.04%

(d) SRL

Table 2: State-of-the-art systems on four NLP tasks. Performance is reported in per-word accuracy for POS, and F1 score for CHUNK, NER and SRL. Systems in bold will be referred as *benchmark systems* in the rest of the paper (see Section 2.6).

tag context as well as multiple words (bigrams, trigrams...) context, and handcrafted features to deal with unknown words. Toutanova et al. (2003), who use maximum entropy classifiers and inference in a bidirectional dependency network (Heckerman et al., 2001), reach 97.24% per-word accuracy. Giménez and Màrquez (2004) proposed a SVM approach also trained on text windows, with bidirectional inference achieved with two Viterbi decoders (left-to-right and right-to-left). They obtained 97.16% per-word accuracy. More recently, Shen et al. (2007) pushed the state-of-the-art up to 97.33%, with a new learning algorithm they call *guided learning*, also for bidirectional sequence classification.

2.2 Chunking

Also called shallow parsing, chunking aims at labeling segments of a sentence with syntactic constituents such as noun or verb phrases (NP or VP). Each word is assigned only one unique tag, often encoded as a begin-chunk (e.g., B-NP) or inside-chunk tag (e.g., I-NP). Chunking is often evaluated using the CoNLL 2000 shared task.[1] Sections 15–18 of WSJ data are used for training and section 20 for testing. Validation is achieved by splitting the training set.

Kudoh and Matsumoto (2000) won the CoNLL 2000 challenge on chunking with a F1-score of 93.48%. Their system was based on Support Vector Machines (SVMs). Each SVM was trained in a pairwise classification manner, and fed with a window around the word of interest containing POS and words as features, as well as surrounding tags. They perform dynamic programming at test time. Later, they improved their results up to 93.91% (Kudo and Matsumoto, 2001) using an ensemble of classifiers trained with different tagging conventions (see Section 3.3.3).

Since then, a certain number of systems based on second-order random fields were reported (Sha and Pereira, 2003; McDonald et al., 2005; Sun et al., 2008), all reporting around 94.3% F1 score. These systems use features composed of words, POS tags, and tags.

More recently, Shen and Sarkar (2005) obtained 95.23% using a voting classifier scheme, where each classifier is trained on different tag representations[2] (IOB, IOE, ...). They use POS features coming from an external tagger, as well carefully hand-crafted *specialization* features which again change the data representation by concatenating some (carefully chosen) chunk tags or some words with their POS representation. They then build trigrams over these features, which are finally passed through a Viterbi decoder a test time.

2.3 Named Entity Recognition

NER labels atomic elements in the sentence into categories such as "PERSON" or "LOCATION". As in the chunking task, each word is assigned a tag prefixed by an indicator of the beginning or the inside of an entity. The CoNLL 2003 setup[3] is a NER benchmark data set based on Reuters data. The contest provides training, validation and testing sets.

Florian et al. (2003) presented the best system at the NER CoNLL 2003 challenge, with 88.76% F1 score. They used a combination of various machine-learning classifiers. Features they picked included words, POS tags, CHUNK tags, prefixes and suffixes, a large gazetteer (not provided by the challenge), as well as the output of two other NER classifiers trained on richer data sets. Chieu (2003), the second best performer of CoNLL 2003 (88.31% F1), also used an external gazetteer (their performance goes down to 86.84% with no gazetteer) and several hand-chosen features.

Later, Ando and Zhang (2005) reached 89.31% F1 with a semi-supervised approach. They trained jointly a linear model on NER with a linear model on two auxiliary unsupervised tasks. They also performed Viterbi decoding at test time. The unlabeled corpus was 27M words taken from Reuters. Features included words, POS tags, suffixes and prefixes or CHUNK tags, but overall were less specialized than CoNLL 2003 challengers.

1. See `http://www.cnts.ua.ac.be/conll2000/chunking`.
2. See Table 3 for tagging scheme details.
3. See `http://www.cnts.ua.ac.be/conll2003/ner`.

2.4 Semantic Role Labeling

SRL aims at giving a semantic role to a syntactic constituent of a sentence. In the PropBank (Palmer et al., 2005) formalism one assigns roles ARG0-5 to words that are arguments of a verb (or more technically, a *predicate*) in the sentence, for example, the following sentence might be tagged "[John]$_{ARG0}$ [ate]$_{REL}$ [the apple]$_{ARG1}$ ", where "ate" is the predicate. The precise arguments depend on a verb's *frame* and if there are multiple verbs in a sentence some words might have multiple tags. In addition to the ARG0-5 tags, there there are several modifier tags such as ARGM-LOC (locational) and ARGM-TMP (temporal) that operate in a similar way for all verbs. We picked CoNLL 2005[4] as our SRL benchmark. It takes sections 2–21 of WSJ data as training set, and section 24 as validation set. A test set composed of section 23 of WSJ concatenated with 3 sections from the Brown corpus is also provided by the challenge.

State-of-the-art SRL systems consist of several stages: producing a parse tree, identifying which parse tree nodes represent the arguments of a given verb, and finally classifying these nodes to compute the corresponding SRL tags. This entails extracting numerous base features from the parse tree and feeding them into statistical models. Feature categories commonly used by these system include (Gildea and Jurafsky, 2002; Pradhan et al., 2004):

- the parts of speech and syntactic labels of words and nodes in the tree;
- the node's position (left or right) in relation to the verb;
- the syntactic path to the verb in the parse tree;
- whether a node in the parse tree is part of a noun or verb phrase;
- the voice of the sentence: active or passive;
- the node's head word; and
- the verb sub-categorization.

Pradhan et al. (2004) take these base features and define additional features, notably the part-of-speech tag of the head word, the predicted named entity class of the argument, features providing word sense disambiguation for the verb (they add 25 variants of 12 new feature types overall). This system is close to the state-of-the-art in performance. Pradhan et al. (2005) obtain 77.30% F1 with a system based on SVM classifiers and simultaneously using the two parse trees provided for the SRL task. In the same spirit, Haghighi et al. (2005) use log-linear models on each tree node, re-ranked globally with a dynamic algorithm. Their system reaches 77.04% using the five top Charniak parse trees.

Koomen et al. (2005) hold the state-of-the-art with Winnow-like (Littlestone, 1988) classifiers, followed by a decoding stage based on an integer program that enforces specific constraints on SRL tags. They reach 77.92% F1 on CoNLL 2005, thanks to the five top parse trees produced by the Charniak (2000) parser (only the first one was provided by the contest) as well as the Collins (1999) parse tree.

4. See `http://www.lsi.upc.edu/~srlconll`.

2.5 Evaluation

In our experiments, we strictly followed the standard evaluation procedure of each CoNLL challenges for NER, CHUNK and SRL. In particular, we chose the hyper-parameters of our model according to a simple validation procedure (see Remark 8 later in Section 3.5), performed over the validation set available for each task (see Section 2). All these three tasks are evaluated by computing the F1 scores over *chunks* produced by our models. The POS task is evaluated by computing the *per-word* accuracy, as it is the case for the standard benchmark we refer to (Toutanova et al., 2003). We used the `conlleval` script[5] for evaluating POS,[6] NER and CHUNK. For SRL, we used the `srl-eval.pl` script included in the `srlconll` package.[7]

2.6 Discussion

When participating in an (open) challenge, it is legitimate to increase generalization by all means. It is thus not surprising to see many top CoNLL systems using *external labeled data*, like additional NER classifiers for the NER architecture of Florian et al. (2003) or additional parse trees for SRL systems (Koomen et al., 2005). Combining multiple systems or tweaking carefully features is also a common approach, like in the chunking top system (Shen and Sarkar, 2005).

However, when *comparing* systems, we do not learn anything of the quality of each system if they were trained with *different* labeled data. For that reason, we will refer to *benchmark systems*, that is, top existing systems which avoid usage of external data and have been well-established in the NLP field: Toutanova et al. (2003) for POS and Sha and Pereira (2003) for chunking. For NER we consider Ando and Zhang (2005) as they were using additional *unlabeled* data only. We picked Koomen et al. (2005) for SRL, keeping in mind they use 4 additional parse trees not provided by the challenge. These benchmark systems will serve as baseline references in our experiments. We marked them in bold in Table 2.

We note that for the four tasks we are considering in this work, it can be seen that for the more complex tasks (with corresponding lower accuracies), the best systems proposed have more engineered features relative to the best systems on the simpler tasks. That is, the POS task is one of the simplest of our four tasks, and only has relatively few engineered features, whereas SRL is the most complex, and many kinds of features have been designed for it. This clearly has implications for as yet unsolved NLP tasks requiring more sophisticated semantic understanding than the ones considered here.

3. The Networks

All the NLP tasks above can be seen as tasks assigning labels to words. The traditional NLP approach is: extract from the sentence a rich set of hand-designed features which are then fed to a standard classification algorithm, for example, a Support Vector Machine (SVM), often with a linear kernel. The choice of features is a completely empirical process, mainly based first on linguistic intuition, and then trial and error, and the feature selection is task dependent, implying additional research for each new NLP task. Complex tasks like SRL then require a large number of possibly

5. Available at `http://www.cnts.ua.ac.be/conll2000/chunking/conlleval.txt`.
6. We used the "`-r`" option of the `conlleval` script to get the per-word accuracy, for POS only.
7. Available at `http://www.lsi.upc.es/~srlconll/srlconll-1.1.tgz`.

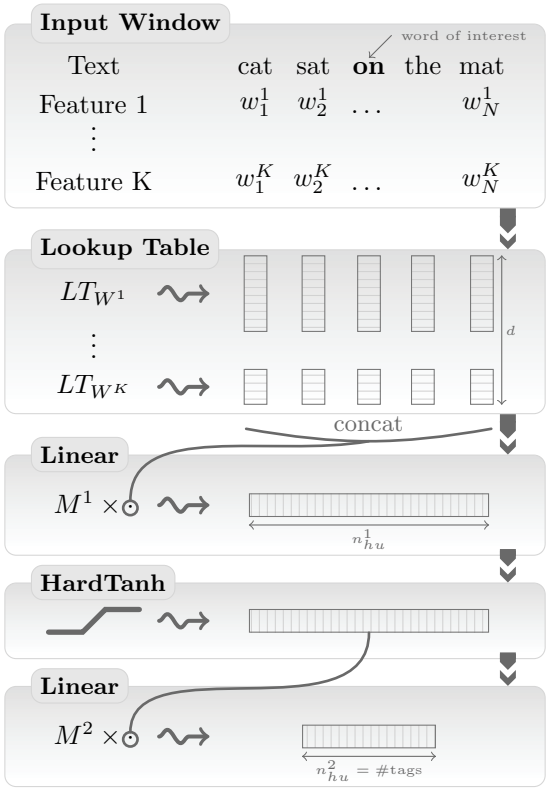


Figure 1: Window approach network.

complex features (e.g., extracted from a parse tree) which can impact the computational cost which might be important for large-scale applications or applications requiring real-time response.

Instead, we advocate a radically different approach: as input we will try to pre-process our features as little as possible and then use a multilayer neural network (NN) architecture, trained in an end-to-end fashion. The architecture takes the input sentence and learns several layers of feature extraction that process the inputs. The features computed by the deep layers of the network are automatically trained by backpropagation to be relevant to the task. We describe in this section a general multilayer architecture suitable for all our NLP tasks, which is generalizable to other NLP tasks as well.

Our architecture is summarized in Figure 1 and Figure 2. The first layer extracts features for each word. The second layer extracts features from a window of words or from the whole sentence, treating it as a *sequence* with local and global structure (i.e., it is not treated like a bag of words). The following layers are standard NN layers.

3.1 Notations

We consider a neural network $f_\theta(\cdot)$, with parameters θ. Any feed-forward neural network with L layers, can be seen as a composition of functions $f^l_\theta(\cdot)$, corresponding to each layer l:

$$f_\theta(\cdot) = f^L_\theta(f^{L-1}_\theta(\ldots f^1_\theta(\cdot)\ldots))\,.$$

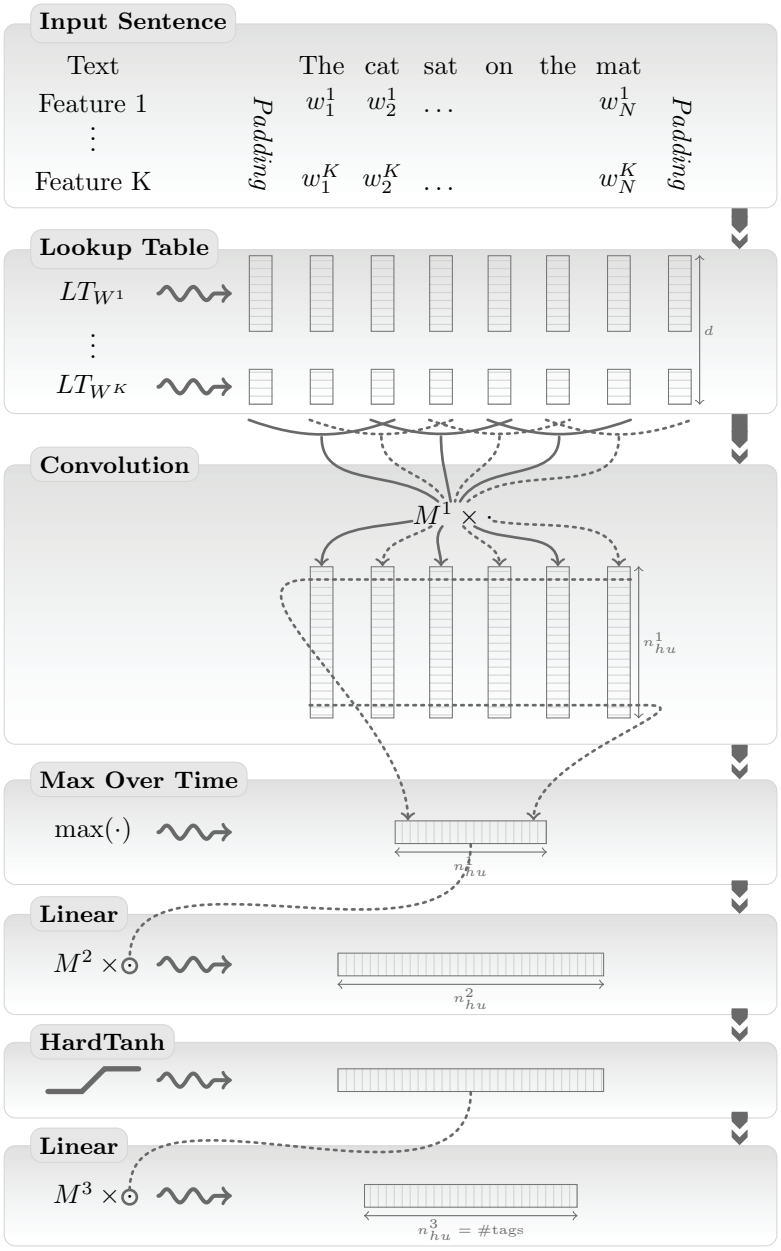


Figure 2: Sentence approach network.

In the following, we will describe each layer we use in our networks shown in Figure 1 and Figure 2. We adopt few notations. Given a matrix A we denote $[A]_{i,j}$ the coefficient at row i and column j in the matrix. We also denote $\langle A \rangle_i^{d_{win}}$ the vector obtained by concatenating the d_{win} column vectors around the i^{th} column vector of matrix $A \in \mathbb{R}^{d_1 \times d_2}$:

$$\left[\langle A \rangle_i^{d_{win}}\right]^{\mathrm{T}} = \left([A]_{1,i-d_{win}/2} \cdots [A]_{d_1,i-d_{win}/2}, \ldots, [A]_{1,i+d_{win}/2} \cdots [A]_{d_1,i+d_{win}/2}\right).$$

As a special case, $\langle A \rangle_i^1$ represents the i^{th} column of matrix A. For a vector v, we denote $[v]_i$ the scalar at index i in the vector. Finally, a sequence of element $\{x_1, x_2, \ldots, x_T\}$ is written $[x]_1^T$. The i^{th} element of the sequence is $[x]_i$.

3.2 Transforming Words into Feature Vectors

One of the key points of our architecture is its ability to perform well with the use of (almost[8]) *raw words*. The ability for our method to learn good word representations is thus crucial to our approach. For efficiency, words are fed to our architecture as indices taken from a finite dictionary $\mathcal{D}$. Obviously, a simple index does not carry much useful information about the word. However, the first layer of our network maps each of these word indices into a feature vector, by a lookup table operation. Given a task of interest, a relevant representation of each word is then given by the corresponding lookup table feature vector, which is *trained* by backpropagation, starting from a random initialization.[9] We will see in Section 4 that we can learn very good word representations from unlabeled corpora. Our architecture allow us to take advantage of better trained word representations, by simply initializing the word lookup table with these representations (instead of randomly).

More formally, for each word $w \in \mathcal{D}$, an internal d_{wrd}-dimensional feature vector representation is given by the *lookup table* layer $LT_W(\cdot)$:

$$LT_W(w) = \langle W \rangle_w^1,$$

where $W \in \mathbb{R}^{d_{wrd} \times |\mathcal{D}|}$ is a matrix of parameters to be learned, $\langle W \rangle_w^1 \in \mathbb{R}^{d_{wrd}}$ is the w^{th} column of W and d_{wrd} is the word vector size (a hyper-parameter to be chosen by the user). Given a sentence or any sequence of T words $[w]_1^T$ in $\mathcal{D}$, the lookup table layer applies the same operation for each word in the sequence, producing the following output matrix:

$$LT_W([w]_1^T) = \begin{pmatrix} \langle W \rangle_{[w]_1}^1 & \langle W \rangle_{[w]_2}^1 & \ldots & \langle W \rangle_{[w]_T}^1 \end{pmatrix}. \quad (1)$$

This matrix can then be fed to further neural network layers, as we will see below.

3.2.1 EXTENDING TO ANY DISCRETE FEATURES

One might want to provide features other than words if one suspects that these features are helpful for the task of interest. For example, for the NER task, one could provide a feature which says if a word is in a gazetteer or not. Another common practice is to introduce some basic pre-processing, such as word-stemming or dealing with upper and lower case. In this latter option, the word would be then represented by three discrete features: its lower case stemmed root, its lower case ending, and a capitalization feature.

Generally speaking, we can consider a word as represented by K discrete features $w \in \mathcal{D}^1 \times \cdots \times \mathcal{D}^K$, where $\mathcal{D}^k$ is the dictionary for the k^{th} feature. We associate to each feature a lookup table $LT_{W^k}(\cdot)$, with parameters $W^k \in \mathbb{R}^{d_{wrd}^k \times |\mathcal{D}^k|}$ where $d_{wrd}^k \in \mathbb{N}$ is a user-specified vector size. Given a

8. We did some pre-processing, namely lowercasing and encoding capitalization as another feature. With enough (unlabeled) training data, presumably we could learn a model without this processing. Ideally, an even more raw input would be to learn from letter sequences rather than words, however we felt that this was beyond the scope of this work.

9. As any other neural network layer.

word w, a feature vector of dimension $d_{wrd} = \sum_k d^k_{wrd}$ is then obtained by concatenating all lookup table outputs:

$$LT_{W^1,\ldots,W^K}(w) = \begin{pmatrix} LT_{W^1}(w_1) \\ \vdots \\ LT_{W^K}(w_K) \end{pmatrix} = \begin{pmatrix} \langle W^1 \rangle^1_{w_1} \\ \vdots \\ \langle W^K \rangle^1_{w_K} \end{pmatrix}.$$

The matrix output of the lookup table layer for a sequence of words $[w]^T_1$ is then similar to (1), but where extra rows have been added for each discrete feature:

$$LT_{W^1,\ldots,W^K}([w]^T_1) = \begin{pmatrix} \langle W^1 \rangle^1_{[w_1]_1} & \cdots & \langle W^1 \rangle^1_{[w_1]_T} \\ \vdots & & \vdots \\ \langle W^K \rangle^1_{[w_K]_1} & \cdots & \langle W^K \rangle^1_{[w_K]_T} \end{pmatrix}. \tag{2}$$

These vector features in the lookup table effectively learn features for words in the dictionary. Now, we want to use these trainable features as input to further layers of trainable feature extractors, that can represent groups of words and then finally sentences.

3.3 Extracting Higher Level Features from Word Feature Vectors

Feature vectors produced by the lookup table layer need to be combined in subsequent layers of the neural network to produce a tag decision for each word in the sentence. Producing tags for each element in variable length sequences (here, a sentence is a sequence of words) is a standard problem in machine-learning. We consider two common approaches which tag *one word at the time*: a window approach, and a (convolutional) sentence approach.

3.3.1 WINDOW APPROACH

A window approach assumes the tag of a word depends mainly on its neighboring words. Given a word to tag, we consider a fixed size k_{sz} (a hyper-parameter) window of words around this word. Each word in the window is first passed through the lookup table layer (1) or (2), producing a matrix of word features of fixed size $d_{wrd} \times k_{sz}$. This matrix can be viewed as a $d_{wrd}\, k_{sz}$-dimensional vector by concatenating each column vector, which can be fed to further neural network layers. More formally, the word feature window given by the first network layer can be written as:

$$f^1_\theta = \langle LT_W([w]^T_1) \rangle^{d_{win}}_t = \begin{pmatrix} \langle W \rangle^1_{[w]_{t-d_{win}/2}} \\ \vdots \\ \langle W \rangle^1_{[w]_t} \\ \vdots \\ \langle W \rangle^1_{[w]_{t+d_{win}/2}} \end{pmatrix}. \tag{3}$$

Linear Layer. The fixed size vector f^1_θ can be fed to one or several standard neural network layers which perform affine transformations over their inputs:

$$f^l_\theta = W^l f^{l-1}_\theta + b^l, \tag{4}$$

where $W^l \in \mathbb{R}^{n^l_{hu} \times n^{l-1}_{hu}}$ and $b^l \in \mathbb{R}^{n^l_{hu}}$ are the parameters to be *trained*. The hyper-parameter n^l_{hu} is usually called the *number of hidden units* of the l^{th} layer.

HardTanh Layer. Several linear layers are often stacked, interleaved with a non-linearity function, to extract highly non-linear features. If no non-linearity is introduced, our network would be a simple linear model. We chose a "hard" version of the hyperbolic tangent as non-linearity. It has the advantage of being slightly cheaper to compute (compared to the exact hyperbolic tangent), while leaving the generalization performance unchanged (Collobert, 2004). The corresponding layer l applies a HardTanh over its input vector:

$$\left[f_\theta^l\right]_i = \text{HardTanh}(\left[f_\theta^{l-1}\right]_i),$$

where

$$\text{HardTanh}(x) = \begin{cases} -1 & \text{if } x < -1 \\ x & \text{if } -1 <= x <= 1 \\ 1 & \text{if } x > 1 \end{cases} . \quad (5)$$

Scoring. Finally, the output size of the last layer L of our network is equal to the number of possible tags for the task of interest. Each output can be then interpreted as a *score* of the corresponding tag (given the input of the network), thanks to a carefully chosen cost function that we will describe later in this section.

Remark 1 (Border Effects) *The feature window (3) is not well defined for words near the beginning or the end of a sentence. To circumvent this problem, we augment the sentence with a special "PADDING" word replicated $d_{win}/2$ times at the beginning and the end. This is akin to the use of "start" and "stop" symbols in sequence models.*

3.3.2 SENTENCE APPROACH

We will see in the experimental section that a window approach performs well for most natural language processing tasks we are interested in. However this approach fails with SRL, where the tag of a word depends on a verb (or, more correctly, predicate) chosen beforehand in the sentence. If the verb falls outside the window, one cannot expect this word to be tagged correctly. In this particular case, tagging a word requires the consideration of the *whole* sentence. When using neural networks, the natural choice to tackle this problem becomes a convolutional approach, first introduced by Waibel et al. (1989) and also called Time Delay Neural Networks (TDNNs) in the literature.

We describe in detail our convolutional network below. It successively takes the complete sentence, passes it through the lookup table layer (1), produces local features around each word of the sentence thanks to convolutional layers, combines these feature into a global feature vector which can then be fed to standard affine layers (4). In the semantic role labeling case, this operation is performed for each word in the sentence, and for each verb in the sentence. It is thus necessary to encode in the network architecture which verb we are considering in the sentence, and which word we want to tag. For that purpose, each word at position i in the sentence is augmented with two features in the way described in Section 3.2.1. These features encode the relative distances $i - pos_v$ and $i - pos_w$ with respect to the chosen verb at position pos_v, and the word to tag at position pos_w respectively.

Convolutional Layer. A convolutional layer can be seen as a generalization of a window approach: given a sequence represented by columns in a matrix f_θ^{l-1} (in our lookup table matrix (1)), a matrix-vector operation as in (4) is applied to each window of successive windows in the sequence.

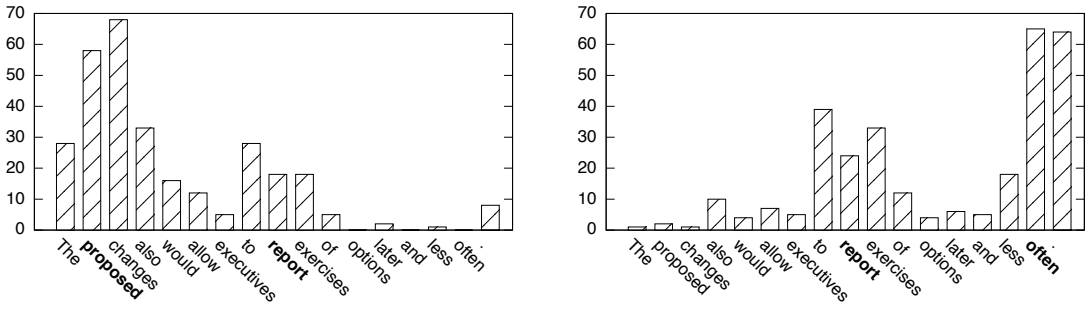


Figure 3: Number of features chosen at each word position by the Max layer. We consider a sentence approach network (Figure 2) trained for SRL. The number of "local" features output by the convolution layer is 300 *per word*. By applying a Max over the sentence, we obtain 300 features for the *whole sentence*. It is interesting to see that the network catches features mostly around the verb of interest (here "report") and word of interest ("proposed" (left) or "often" (right)).

Using previous notations, the t^{th} output column of the l^{th} layer can be computed as:

$$\langle f_\theta^l \rangle_t^1 = W^l \, \langle f_\theta^{l-1} \rangle_t^{d_{win}} + b^l \quad \forall t \,, \tag{6}$$

where the weight matrix W^l is the same across all windows t in the sequence. Convolutional layers extract local features around each window of the given sequence. As for standard affine layers (4), convolutional layers are often stacked to extract higher level features. In this case, each layer must be followed by a non-linearity (5) or the network would be equivalent to one convolutional layer.

Max Layer. The size of the output (6) depends on the number of words in the sentence fed to the network. Local feature vectors extracted by the convolutional layers have to be combined to obtain a global feature vector, with a fixed size independent of the sentence length, in order to apply subsequent standard affine layers. Traditional convolutional networks often apply an average (possibly weighted) or a max operation over the "time" t of the sequence (6). (Here, "time" just means the position in the sentence, this term stems from the use of convolutional layers in, for example, speech data where the sequence occurs over time.) The average operation does not make much sense in our case, as in general most words in the sentence do not have any influence on the semantic role of a given word to tag. Instead, we used a max approach, which forces the network to capture the most useful local features produced by the convolutional layers (see Figure 3), for the task at hand. Given a *matrix* f_θ^{l-1} output by a convolutional layer $l-1$, the Max layer l outputs a *vector* f_θ^l:

$$\left[f_\theta^l \right]_i = \max_t \left[f_\theta^{l-1} \right]_{i,t} \qquad 1 \leq i \leq n_{hu}^{l-1} \,. \tag{7}$$

This fixed sized global feature vector can be then fed to standard affine network layers (4). As in the window approach, we then finally produce one score per possible tag for the given task.

Remark 2 *The same border effects arise in the convolution operation (6) as in the window approach (3). We again work around this problem by padding the sentences with a special word.*

Scheme	Begin	Inside	End	Single	Other
IOB	B-X	I-X	I-X	B-X	O
IOE	I-X	I-X	E-X	E-X	O
IOBES	B-X	I-X	E-X	S-X	O

Table 3: Various tagging schemes. Each word in a segment labeled "X" is tagged with a prefixed label, depending of the word position in the segment (begin, inside, end). Single word segment labeling is also output. Words not in a labeled segment are labeled "O". Variants of the IOB (and IOE) scheme exist, where the prefix B (or E) is replaced by I for all segments not contiguous with another segment having the same label "X".

3.3.3 TAGGING SCHEMES

As explained earlier, the network output layers compute scores for all the possible tags for the task of interest. In the window approach, these tags apply to the word located in the center of the window. In the (convolutional) sentence approach, these tags apply to the word designated by additional markers in the network input.

The POS task indeed consists of marking the syntactic role of each word. However, the remaining three tasks associate labels with segments of a sentence. This is usually achieved by using special tagging schemes to identify the segment boundaries, as shown in Table 3. Several such schemes have been defined (IOB, IOE, IOBES, ...) without clear conclusion as to which scheme is better in general. State-of-the-art performance is sometimes obtained by combining classifiers trained with different tagging schemes (e.g., Kudo and Matsumoto, 2001).

The ground truth for the NER, CHUNK, and SRL tasks is provided using two different tagging schemes. In order to eliminate this additional source of variations, we have decided to use the most expressive IOBES tagging scheme for all tasks. For instance, in the CHUNK task, we describe noun phrases using four different tags. Tag "S-NP" is used to mark a noun phrase containing a single word. Otherwise tags "B-NP", "I-NP", and "E-NP" are used to mark the first, intermediate and last words of the noun phrase. An additional tag "O" marks words that are not members of a chunk. During testing, these tags are then converted to the original IOB tagging scheme and fed to the standard performance evaluation scripts mentioned in Section 2.5.

3.4 Training

All our neural networks are trained by maximizing a likelihood over the training data, using stochastic gradient ascent. If we denote θ to be all the trainable parameters of the network, which are trained using a training set $\mathcal{T}$ we want to maximize the following log-likelihood with respect to θ:

$$\theta \mapsto \sum_{(x,y)\in\mathcal{T}} \log p(y \,|\, x, \theta)\,, \tag{8}$$

where x corresponds to either a training word window or a sentence and its associated features, and y represents the corresponding tag. The probability $p(\cdot)$ is computed from the outputs of the neural network. We will see in this section two ways of interpreting neural network outputs as probabilities.

3.4.1 WORD-LEVEL LOG-LIKELIHOOD

In this approach, each word in a sentence is considered independently. Given an input example x, the network with parameters θ outputs a score $\left[f_\theta(x)\right]_i$, for the i^{th} tag with respect to the task of interest. To simplify the notation, we drop x from now, and we write instead $\left[f_\theta\right]_i$. This score can be interpreted as a conditional tag probability $p(i\,|\,x,\theta)$ by applying a softmax (Bridle, 1990) operation over all the tags:

$$p(i\,|\,x,\theta) = \frac{e^{\left[f_\theta\right]_i}}{\sum_j e^{\left[f_\theta\right]_j}}\,. \tag{9}$$

Defining the log-add operation as

$$\underset{i}{\text{logadd}}\, z_i = \log\Big(\sum_i e^{z_i}\Big)\,, \tag{10}$$

we can express the log-likelihood for one training example (x,y) as follows:

$$\log p(y\,|\,x,\theta) = [f_\theta]_y - \underset{j}{\text{logadd}}\,[f_\theta]_j\,. \tag{11}$$

While this training criterion, often referred as *cross-entropy* is widely used for classification problems, it might not be ideal in our case, where there is often a correlation between the tag of a word in a sentence and its neighboring tags. We now describe another common approach for neural networks which enforces dependencies between the predicted tags in a sentence.

3.4.2 SENTENCE-LEVEL LOG-LIKELIHOOD

In tasks like chunking, NER or SRL we know that there are dependencies between word tags in a sentence: not only are tags organized in chunks, but some tags cannot follow other tags. Training using a word-level approach discards this kind of labeling information. We consider a training scheme which takes into account the sentence structure: given the predictions of *all tags* by our network for *all words* in a sentence, and given a score for going from one tag to another tag, we want to encourage valid paths of tags during training, while discouraging all other paths.

We consider the *matrix* of scores $f_\theta([x]_1^T)$ output by the network. As before, we drop the input $[x]_1^T$ for notation simplification. The element $\left[f_\theta\right]_{i,t}$ of the matrix is the score output by the network with parameters θ, for the sentence $[x]_1^T$ and for the i^{th} tag, at the t^{th} word. We introduce a transition score $[A]_{i,j}$ for jumping from i to j tags in successive words, and an initial score $[A]_{i,0}$ for starting from the i^{th} tag. As the transition scores are going to be trained (as are all network parameters θ), we define $\tilde{\theta} = \theta \cup \{[A]_{i,j}\ \forall i,j\}$. The score of a sentence $[x]_1^T$ along a path of tags $[i]_1^T$ is then given by the sum of transition scores and network scores:

$$s([x]_1^T, [i]_1^T, \tilde{\theta}) = \sum_{t=1}^{T} \Big([A]_{[i]_{t-1},[i]_t} + [f_\theta]_{[i]_t,t} \Big)\,. \tag{12}$$

Exactly as for the word-level likelihood (11), where we were normalizing with respect to all *tags* using a softmax (9), we normalize this score over all possible *tag paths* $[j]_1^T$ using a softmax, and we interpret the resulting ratio as a conditional *tag path* probability. Taking the log, the conditional probability of the true path $[y]_1^T$ is therefore given by:

$$\log p([y]_1^T\,|\,[x]_1^T, \tilde{\theta}) = s([x]_1^T, [y]_1^T, \tilde{\theta}) - \underset{\forall [j]_1^T}{\text{logadd}}\, s([x]_1^T, [j]_1^T, \tilde{\theta})\,. \tag{13}$$

While the number of terms in the logadd operation (11) was equal to the number of tags, it grows exponentially with the length of the sentence in (13). Fortunately, one can compute it in linear time with the following standard recursion over t (see Rabiner, 1989), taking advantage of the associativity and distributivity on the semi-ring[10] $(\mathbb{R} \cup \{-\infty\}, \text{logadd}, +)$:

$$\begin{aligned}
\delta_t(k) &\stackrel{\Delta}{=} \operatorname*{logadd}_{\{[j]_1^t \cap [j]_t = k\}} s([x]_1^t, [j]_1^t, \tilde{\theta}) \\
&= \operatorname*{logadd}_i \operatorname*{logadd}_{\{[j]_1^t \cap [j]_{t-1} = i \cap [j]_t = k\}} s([x]_1^t, [j]_1^{t-1}, \tilde{\theta}) + [A]_{[j]_{t-1},k} + [f_\theta]_{k,t} \\
&= \operatorname*{logadd}_i \delta_{t-1}(i) + [A]_{i,k} + [f_\theta]_{k,t} \\
&= [f_\theta]_{k,t} + \operatorname*{logadd}_i \left(\delta_{t-1}(i) + [A]_{i,k} \right) \quad \forall k \,,
\end{aligned} \tag{14}$$

followed by the termination

$$\operatorname*{logadd}_{\forall [j]_1^T} s([x]_1^T, [j]_1^T, \tilde{\theta}) = \operatorname*{logadd}_i \delta_T(i) \,. \tag{15}$$

We can now maximize in (8) the log-likelihood (13) over all the training pairs $([x]_1^T, [y]_1^T)$.

At inference time, given a sentence $[x]_1^T$ to tag, we have to find the best tag path which minimizes the sentence score (12). In other words, we must find

$$\operatorname*{argmax}_{[j]_1^T} s([x]_1^T, [j]_1^T, \tilde{\theta}) \,.$$

The Viterbi algorithm is the natural choice for this inference. It corresponds to performing the recursion (14) and (15), but where the logadd is replaced by a max, and then tracking back the optimal path through each max.

Remark 3 (Graph Transformer Networks) *Our approach is a particular case of the discriminative forward training for graph transformer networks (GTNs) (Bottou et al., 1997; Le Cun et al., 1998). The log-likelihood (13) can be viewed as the difference between the forward score constrained over the valid paths (in our case there is only the labeled path) and the unconstrained forward score (15).*

Remark 4 (Conditional Random Fields) *An important feature of equation (12) is the absence of normalization. Summing the exponentials $e^{[f_\theta]_{i,t}}$ over all possible tags does not necessarily yield the unity. If this was the case, the scores could be viewed as the logarithms of conditional transition probabilities, and our model would be subject to the label-bias problem that motivates Conditional Random Fields (CRFs) (Lafferty et al., 2001). The denormalized scores should instead be likened to the potential functions of a CRF. In fact, a CRF maximizes the same likelihood (13) using a linear model instead of a nonlinear neural network. CRFs have been widely used in the NLP world, such as for POS tagging (Lafferty et al., 2001), chunking (Sha and Pereira, 2003), NER (McCallum and Li, 2003) or SRL (Cohn and Blunsom, 2005). Compared to such CRFs, we take advantage of the nonlinear network to learn appropriate features for each task of interest.*

10. In other words, read logadd as $\oplus$ and $+$ as $\otimes$.

3.4.3 STOCHASTIC GRADIENT

Maximizing (8) with stochastic gradient (Bottou, 1991) is achieved by iteratively selecting a random example (x, y) and making a gradient step:

$$\theta \longleftarrow \theta + \lambda \frac{\partial \log p(y \,|\, x, \theta)}{\partial \theta}, \tag{16}$$

where λ is a chosen learning rate. Our neural networks described in Figure 1 and Figure 2 are a succession of layers that correspond to successive composition of functions. The neural network is finally composed with the word-level log-likelihood (11), or successively composed in the recursion (14) if using the sentence-level log-likelihood (13). Thus, an *analytical* formulation of the derivative (16) can be computed, by applying the differentiation chain rule through the network, and through the word-level log-likelihood (11) or through the recurrence (14).

Remark 5 (Differentiability) *Our cost functions are differentiable almost everywhere. Non-differentiable points arise because we use a "hard" transfer function (5) and because we use a "max" layer (7) in the sentence approach network. Fortunately, stochastic gradient still converges to a meaningful local minimum despite such minor differentiability problems (Bottou, 1991, 1998). Stochastic gradient iterations that hit a non-differentiability are simply skipped.*

Remark 6 (Modular Approach) *The well known "back-propagation" algorithm (LeCun, 1985; Rumelhart et al., 1986) computes gradients using the chain rule. The chain rule can also be used in a modular implementation.*[11] *Our modules correspond to the boxes in Figure 1 and Figure 2. Given derivatives with respect to its outputs, each module can independently compute derivatives with respect to its inputs and with respect to its trainable parameters, as proposed by Bottou and Gallinari (1991). This allows us to easily build variants of our networks. For details about gradient computations, see Appendix A.*

Remark 7 (Tricks) *Many tricks have been reported for training neural networks (LeCun et al., 1998). Which ones to choose is often confusing. We employed only two of them: the initialization and update of the parameters of each network layer were done according to the "fan-in" of the layer, that is the number of inputs used to compute each output of this layer (Plaut and Hinton, 1987). The fan-in for the lookup table (1), the l^{th} linear layer (4) and the convolution layer (6) are respectively 1, n_{hu}^{l-1} and $d_{win} \times n_{hu}^{l-1}$. The initial parameters of the network were drawn from a centered uniform distribution, with a variance equal to the inverse of the square-root of the fan-in. The learning rate in (16) was divided by the fan-in, but stays fixed during the training.*

3.5 Supervised Benchmark Results

For POS, chunking and NER tasks, we report results with the window architecture[12] described in Section 3.3.1. The SRL task was trained using the sentence approach (Section 3.3.2). Results are reported in Table 4, in per-word accuracy (PWA) for POS, and F1 score for all the other tasks. We performed experiments both with the word-level log-likelihood (WLL) and with the sentence-level log-likelihood (SLL). The hyper-parameters of our networks are reported in Table 5. All our

11. See `http://torch5.sf.net`.

12. We found that training these tasks with the more complex sentence approach was computationally expensive and offered little performance benefits. Results discussed in Section 5 provide more insight about this decision.

Approach	**POS** (PWA)	**Chunking** (F1)	**NER** (F1)	**SRL** (F1)
Benchmark Systems	97.24	94.29	89.31	77.92
NN+WLL	96.31	89.13	79.53	55.40
NN+SLL	96.37	90.33	81.47	70.99

Table 4: Comparison in generalization performance of benchmark NLP systems with a vanilla neural network (NN) approach, on POS, chunking, NER and SRL tasks. We report results with both the word-level log-likelihood (WLL) and the sentence-level log-likelihood (SLL). Generalization performance is reported in per-word accuracy rate (PWA) for POS and F1 score for other tasks. The NN results are behind the benchmark results, in Section 4 we show how to improve these models using unlabeled data.

Task	Window/Conv. size	Word dim.	Caps dim.	Hidden units	Learning rate
POS	$d_{win} = 5$	$d^0 = 50$	$d^1 = 5$	$n^1_{hu} = 300$	$\lambda = 0.01$
CHUNK	”	”	”	”	”
NER	”	”	”	”	”
SRL	”	”	”	$n^1_{hu} = 300$ $n^2_{hu} = 500$	”

Table 5: Hyper-parameters of our networks. They were chosen by a minimal validation (see Remark 8), preferring identical parameters for most tasks. We report for each task the window size (or convolution size), word feature dimension, capital feature dimension, number of hidden units and learning rate.

networks were fed with two raw text features: lower case words, and a capital letter feature. We chose to consider lower case words to limit the number of words in the dictionary. However, to keep some upper case information lost by this transformation, we added a “caps” feature which tells if each word was in lowercase, was all uppercase, had first letter capital, or had at least one non-initial capital letter. Additionally, all occurrences of sequences of numbers within a word are replaced with the string “NUMBER”, so for example both the words “PS1” and “PS2” would map to the single word “psNUMBER”. We used a dictionary containing the 100,000 most common words in WSJ (case insensitive). Words outside this dictionary were replaced by a single special “RARE” word.

Results show that neural networks “out-of-the-box” are behind baseline benchmark systems. Although the initial performance of our networks falls short from the performance of the CoNLL challenge winners, it compares honorably with the performance of most competitors. The training criterion which takes into account the sentence structure (SLL) seems to boost the performance for the Chunking, NER and SRL tasks, with little advantage for POS. This result is in line with existing NLP studies comparing sentence-level and word-level likelihoods (Liang et al., 2008). The capacity of our network architectures lies mainly in the word lookup table, which contains $50 \times 100,000$ parameters to train. In the WSJ data, 15% of the most common words appear about 90% of the time. Many words appear only a few times. It is thus very difficult to train properly their corresponding

FRANCE	JESUS	XBOX	REDDISH	SCRATCHED	MEGABITS
454	1973	6909	11724	29869	87025
PERSUADE	THICKETS	DECADENT	WIDESCREEN	ODD	PPA
FAW	SAVARY	DIVO	ANTICA	ANCHIETA	UDDIN
BLACKSTOCK	SYMPATHETIC	VERUS	SHABBY	EMIGRATION	BIOLOGICALLY
GIORGI	JFK	OXIDE	AWE	MARKING	KAYAK
SHAHEED	KHWARAZM	URBINA	THUD	HEUER	MCLARENS
RUMELIA	STATIONERY	EPOS	OCCUPANT	SAMBHAJI	GLADWIN
PLANUM	ILIAS	EGLINTON	REVISED	WORSHIPPERS	CENTRALLY
GOA'ULD	GSNUMBER	EDGING	LEAVENED	RITSUKO	INDONESIA
COLLATION	OPERATOR	FRG	PANDIONIDAE	LIFELESS	MONEO
BACHA	W.J.	NAMSOS	SHIRT	MAHAN	NILGIRIS

Table 6: Word embeddings in the word lookup table of a SRL neural network trained from scratch, with a dictionary of size 100,000. For each column the queried word is followed by its index in the dictionary (higher means more rare) and its 10 nearest neighbors (arbitrarily using the Euclidean metric).

50 dimensional feature vectors in the lookup table. Ideally, we would like semantically similar words to be close in the embedding space represented by the word lookup table: by continuity of the neural network function, tags produced on semantically similar sentences would be similar. We show in Table 6 that it is not the case: neighboring words in the embedding space do not seem to be semantically related.

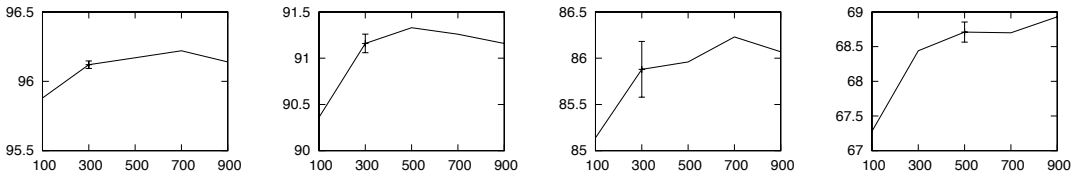
We will focus in the next section on improving these word embeddings by leveraging unlabeled data. We will see our approach results in a performance boost for all tasks.

Remark 8 (Architectures) *In all our experiments in this paper, we tuned the hyper-parameters by trying only a few different architectures by validation. In practice, the choice of hyperparameters such as the number of hidden units, provided they are large enough, has a limited impact on the generalization performance. In Figure 4, we report the F1 score for each task on the validation set, with respect to the number of hidden units. Considering the variance related to the network initialization, we chose the smallest network achieving "reasonable" performance, rather than picking the network achieving the top performance obtained on a single run.*

Remark 9 (Training Time) *Training our network is quite computationally expensive. Chunking and NER take about one hour to train, POS takes few hours, and SRL takes about three days. Training could be faster with a larger learning rate, but we preferred to stick to a small one which works, rather than finding the optimal one for speed. Second order methods (LeCun et al., 1998) could be another speedup technique.*

4. Lots of Unlabeled Data

We would like to obtain word embeddings carrying more syntactic and semantic information than shown in Table 6. Since most of the trainable parameters of our system are associated with the word embeddings, these poor results suggest that we should use considerably more training data.



(a) POS (b) CHUNK (c) NER (d) SRL

Figure 4: F1 score on the *validation* set (y-axis) versus number of hidden units (x-axis) for different tasks trained with the sentence-level likelihood (SLL), as in Table 4. For SRL, we vary in this graph only the number of hidden units in the second layer. The scale is adapted for each task. We show the standard deviation (obtained over 5 runs with different random initialization), for the architecture we picked (300 hidden units for POS, CHUNK and NER, 500 for SRL).

Following our NLP *from scratch* philosophy, we now describe how to dramatically improve these embeddings using large unlabeled data sets. We then use these improved embeddings to initialize the word lookup tables of the networks described in Section 3.5.

4.1 Data Sets

Our first English corpus is the entire English Wikipedia.[13] We have removed all paragraphs containing non-roman characters and all MediaWiki markups. The resulting text was tokenized using the Penn Treebank tokenizer script.[14] The resulting data set contains about 631 million words. As in our previous experiments, we use a dictionary containing the 100,000 most common words in WSJ, with the same processing of capitals and numbers. Again, words outside the dictionary were replaced by the special "RARE" word.

Our second English corpus is composed by adding an extra 221 million words extracted from the Reuters RCV1 (Lewis et al., 2004) data set.[15] We also extended the dictionary to 130,000 words by adding the 30,000 most common words in Reuters. This is useful in order to determine whether improvements can be achieved by further increasing the unlabeled data set size.

4.2 Ranking Criterion versus Entropy Criterion

We used these unlabeled data sets to train *language models* that compute *scores* describing the acceptability of a piece of text. These language models are again large neural networks using the window approach described in Section 3.3.1 and in Figure 1. As in the previous section, most of the trainable parameters are located in the lookup tables.

Similar language models were already proposed by Bengio and Ducharme (2001) and Schwenk and Gauvain (2002). Their goal was to estimate the *probability* of a word given the previous words in a sentence. Estimating conditional probabilities suggests a cross-entropy criterion similar to those described in Section 3.4.1. Because the dictionary size is large, computing the normalization term

13. Available at `http://download.wikimedia.org`. We took the November 2007 version.
14. Available at `http://www.cis.upenn.edu/~treebank/tokenization.html`.
15. Now available at `http://trec.nist.gov/data/reuters/reuters.html`.

can be extremely demanding, and sophisticated approximations are required. More importantly for us, neither work leads to significant word embeddings being reported.

Shannon (1951) has estimated the entropy of the English language between 0.6 and 1.3 bits per character by asking human subjects to guess upcoming characters. Cover and King (1978) give a lower bound of 1.25 bits per character using a subtle gambling approach. Meanwhile, using a simple word trigram model, Brown et al. (1992b) reach 1.75 bits per character. Teahan and Cleary (1996) obtain entropies as low as 1.46 bits per character using variable length character *n*-grams. The human subjects rely of course on all their knowledge of the language and of the world. Can we learn the grammatical structure of the English language and the nature of the world by leveraging the 0.2 bits per character that separate human subjects from simple n-gram models? Since such tasks certainly require high capacity models, obtaining sufficiently small confidence intervals on the test set entropy may require prohibitively large training sets.[16] The entropy criterion lacks dynamical range because its numerical value is largely determined by the most frequent phrases. In order to learn syntax, rare but legal phrases are no less significant than common phrases.

It is therefore desirable to define alternative training criteria. We propose here to use a *pairwise ranking* approach (Cohen et al., 1998). We seek a network that computes a higher score when given a legal phrase than when given an incorrect phrase. Because the ranking literature often deals with information retrieval applications, many authors define complex ranking criteria that give more weight to the ordering of the best ranking instances (see Burges et al., 2007; Clémençon and Vayatis, 2007). However, in our case, we do not want to emphasize the most common phrase over the rare but legal phrases. Therefore we use a simple pairwise criterion.

We consider a *window* approach network, as described in Section 3.3.1 and Figure 1, with parameters θ which outputs a score $f_\theta(x)$ given a window of text $x = [w]_1^{d_{win}}$. We minimize the ranking criterion with respect to θ:

$$\theta \mapsto \sum_{x \in \mathcal{X}} \sum_{w \in \mathcal{D}} \max \left\{ 0 \, , \, 1 - f_\theta(x) + f_\theta(x^{(w)}) \right\}, \tag{17}$$

where $\mathcal{X}$ is the set of all possible text windows with d_{win} words coming from our training corpus, $\mathcal{D}$ is the dictionary of words, and $x^{(w)}$ denotes the text window obtained by replacing the central word of text window $[w]_1^{d_{win}}$ by the word w.

Okanohara and Tsujii (2007) use a related approach to avoiding the entropy criteria using a binary classification approach (correct/incorrect phrase). Their work focuses on using a kernel classifier, and not on learning word embeddings as we do here. Smith and Eisner (2005) also propose a contrastive criterion which estimates the likelihood of the data conditioned to a "negative" neighborhood. They consider various data neighborhoods, including sentences of length d_{win} drawn from $\mathcal{D}^{d_{win}}$. Their goal was however to perform well on some tagging task on fully unsupervised data, rather than obtaining generic word embeddings useful for other tasks.

4.3 Training Language Models

The language model network was trained by stochastic gradient minimization of the ranking criterion (17), sampling a sentence-word pair (s, w) at each iteration.

16. However, Klein and Manning (2002) describe a rare example of realistic unsupervised grammar induction using a cross-entropy approach on binary-branching parsing trees, that is, by forcing the system to generate a hierarchical representation.

Since training times for such large scale systems are counted in weeks, it is not feasible to try many combinations of hyperparameters. It also makes sense to speed up the training time by initializing new networks with the embeddings computed by earlier networks. In particular, we found it expedient to train a succession of networks using increasingly large dictionaries, each network being initialized with the embeddings of the previous network. Successive dictionary sizes and switching times are chosen arbitrarily. Bengio et al. (2009) provides a more detailed discussion of this, the (as yet, poorly understood) "curriculum" process.

For the purposes of model selection we use the process of "breeding". The idea of breeding is instead of trying a full grid search of possible values (which we did not have enough computing power for) to search for the parameters in analogy to breeding biological cell lines. Within each line, child networks are initialized with the embeddings of their parents and trained on increasingly rich data sets with sometimes different parameters. That is, suppose we have k processors, which is much less than the possible set of parameters one would like to try. One chooses k initial parameter choices from the large set, and trains these on the k processors. In our case, possible parameters to adjust are: the learning rate λ, the word embedding dimensions d, number of hidden units n^1_{hu} and input window size d_{win}. One then trains each of these models in an online fashion for a certain amount of time (i.e., a few days), and then selects the best ones using the validation set error rate. That is, breeding decisions were made on the basis of the value of the ranking criterion (17) estimated on a validation set composed of one million words held out from the Wikipedia corpus. In the next breeding iteration, one then chooses another set of k parameters from the possible grid of values that permute slightly the most successful candidates from the previous round. As many of these parameter choices can share weights, we can effectively continue online training retaining some of the learning from the previous iterations.

Very long training times make such strategies necessary for the foreseeable future: if we had been given computers ten times faster, we probably would have found uses for data sets ten times bigger. However, we should say we believe that although we ended up with a particular choice of parameters, many other choices are almost equally as good, although perhaps there are others that are better as we could not do a full grid search.

In the following subsections, we report results obtained with two trained language models. The results achieved by these two models are representative of those achieved by networks trained on the full corpora.

- Language model LM1 has a window size $d_{win} = 11$ and a hidden layer with $n^1_{hu} = 100$ units. The embedding layers were dimensioned like those of the supervised networks (Table 5). Model LM1 was trained on our first English corpus (Wikipedia) using successive dictionaries composed of the 5000, 10,000, 30,000, 50,000 and finally 100,000 most common WSJ words. The total training time was about four weeks.

- Language model LM2 has the same dimensions. It was initialized with the embeddings of LM1, and trained for an additional three weeks on our second English corpus (Wikipedia+Reuters) using a dictionary size of 130,000 words.

4.4 Embeddings

Both networks produce much more appealing word embeddings than in Section 3.5. Table 7 shows the ten nearest neighbors of a few randomly chosen query words for the LM1 model. The syntactic

FRANCE	JESUS	XBOX	REDDISH	SCRATCHED	MEGABITS
454	1973	6909	11724	29869	87025
AUSTRIA	GOD	AMIGA	GREENISH	NAILED	OCTETS
BELGIUM	SATI	PLAYSTATION	BLUISH	SMASHED	MB/S
GERMANY	CHRIST	MSX	PINKISH	PUNCHED	BIT/S
ITALY	SATAN	IPOD	PURPLISH	POPPED	BAUD
GREECE	KALI	SEGA	BROWNISH	CRIMPED	CARATS
SWEDEN	INDRA	PSNUMBER	GREYISH	SCRAPED	KBIT/S
NORWAY	VISHNU	HD	GRAYISH	SCREWED	MEGAHERTZ
EUROPE	ANANDA	DREAMCAST	WHITISH	SECTIONED	MEGAPIXELS
HUNGARY	PARVATI	GEFORCE	SILVERY	SLASHED	GBIT/S
SWITZERLAND	GRACE	CAPCOM	YELLOWISH	RIPPED	AMPERES

Table 7: Word embeddings in the word lookup table of the language model neural network LM1 trained with a dictionary of size 100,000. For each column the queried word is followed by its index in the dictionary (higher means more rare) and its 10 nearest neighbors (using the Euclidean metric, which was chosen arbitrarily).

and semantic properties of the neighbors are clearly related to those of the query word. These results are far more satisfactory than those reported in Table 7 for embeddings obtained using purely supervised training of the benchmark NLP tasks.

4.5 Semi-supervised Benchmark Results

Semi-supervised learning has been the object of much attention during the last few years (see Chapelle et al., 2006). Previous semi-supervised approaches for NLP can be roughly categorized as follows:

- Ad-hoc approaches such as Rosenfeld and Feldman (2007) for relation extraction.
- Self-training approaches, such as Ueffing et al. (2007) for machine translation, and McClosky et al. (2006) for parsing. These methods augment the labeled training set with examples from the unlabeled data set using the labels predicted by the model itself. Transductive approaches, such as Joachims (1999) for text classification can be viewed as a refined form of self-training.
- Parameter sharing approaches such as Ando and Zhang (2005); Suzuki and Isozaki (2008). Ando and Zhang propose a multi-task approach where they jointly train models sharing certain parameters. They train POS and NER models together with a language model (trained on 15 million words) consisting of predicting words given the surrounding tokens. Suzuki and Isozaki embed a generative model (Hidden Markov Model) inside a CRF for POS, Chunking and NER. The generative model is trained on one billion words. These approaches should be seen as a linear counterpart of our work. Using multilayer models vastly expands the parameter sharing opportunities (see Section 5).

Our approach simply consists of initializing the word lookup tables of the supervised networks with the embeddings computed by the language models. Supervised training is then performed as in Section 3.5. In particular the supervised training stage is free to modify the lookup tables. This sequential approach is computationally convenient because it separates the lengthy training of the

Approach	POS (PWA)	CHUNK (F1)	NER (F1)	SRL (F1)
Benchmark Systems	97.24	94.29	89.31	77.92
NN+WLL	96.31	89.13	79.53	55.40
NN+SLL	96.37	90.33	81.47	70.99
NN+WLL+LM1	97.05	91.91	85.68	58.18
NN+SLL+LM1	97.10	93.65	87.58	73.84
NN+WLL+LM2	97.14	92.04	86.96	58.34
NN+SLL+LM2	97.20	93.63	88.67	74.15

Table 8: Comparison in generalization performance of benchmark NLP systems with our (NN) approach on POS, chunking, NER and SRL tasks. We report results with both the word-level log-likelihood (WLL) and the sentence-level log-likelihood (SLL). We report with (LM*n*) performance of the networks trained from the language model embeddings (Table 7). Generalization performance is reported in per-word accuracy (PWA) for POS and F1 score for other tasks.

language models from the relatively fast training of the supervised networks. Once the language models are trained, we can perform multiple experiments on the supervised networks in a relatively short time. Note that our procedure is clearly linked to the (semi-supervised) deep learning procedures of Hinton et al. (2006), Bengio et al. (2007) and Weston et al. (2008).

Table 8 clearly shows that this simple initialization significantly boosts the generalization performance of the supervised networks for each task. It is worth mentioning the larger language model led to even better performance. This suggests that we could still take advantage of even bigger unlabeled data sets.

4.6 Ranking and Language

There is a large agreement in the NLP community that syntax is a necessary prerequisite for semantic role labeling (Gildea and Palmer, 2002). This is why state-of-the-art semantic role labeling systems thoroughly exploit multiple parse trees. The parsers themselves (Charniak, 2000; Collins, 1999) contain considerable prior information about syntax (one can think of this as a kind of informed pre-processing).

Our system does not use such parse trees because we attempt to learn this information from the unlabeled data set. It is therefore legitimate to question whether our ranking criterion (17) has the conceptual capability to capture such a rich hierarchical information. At first glance, the ranking task appears unrelated to the induction of probabilistic grammars that underly standard parsing algorithms. The lack of hierarchical representation seems a fatal flaw (Chomsky, 1956).

However, ranking is closely related to an alternative description of the language structure: *operator grammars* (Harris, 1968). Instead of directly studying the structure of a sentence, Harris defines an algebraic structure on the space of all sentences. Starting from a couple of elementary sentence forms, sentences are described by the successive application of sentence transformation operators. The sentence structure is revealed as a side effect of the successive transformations. Sentence transformations can also have a semantic interpretation.

In the spirit of structural linguistics, Harris describes procedures to discover sentence transformation operators by leveraging the statistical regularities of the language. Such procedures are obviously useful for machine learning approaches. In particular, he proposes a test to decide whether two sentences forms are semantically related by a transformation operator. He first defines a ranking criterion (Harris, 1968, Section 4.1):

> "Starting for convenience with very short sentence forms, say ABC, we choose a particular word choice for all the classes, say B_qC_q, except one, in this case A; for every pair of members A_i, A_j of that word class we ask how the sentence formed with one of the members, that is, $A_iB_qC_q$ compares as to acceptability with the sentence formed with the other member, that is, $A_jB_qC_q$."

These *gradings* are then used to compare sentence forms:

> "It now turns out that, given the graded n-tuples of words for a particular sentence form, we can find other sentences forms of the same word classes in which the same n-tuples of words produce the same grading of sentences."

This is an indication that these two sentence forms exploit common words with the same syntactic function and possibly the same meaning. This observation forms the empirical basis for the construction of operator grammars that describe real-world natural languages such as English.

Therefore there are solid reasons to believe that the ranking criterion (17) has the conceptual potential to capture strong syntactic and semantic information. On the other hand, the structure of our language models is probably too restrictive for such goals, and our current approach only exploits the word embeddings discovered during training.

5. Multi-Task Learning

It is generally accepted that features *trained* for one task can be useful for *related tasks*. This idea was already exploited in the previous section when certain language model features, namely the word embeddings, were used to initialize the supervised networks.

Multi-task learning (MTL) leverages this idea in a more systematic way. Models for all tasks of interests are *jointly trained* with an additional linkage between their trainable parameters in the hope of improving the generalization error. This linkage can take the form of a regularization term in the joint cost function that biases the models towards common representations. A much simpler approach consists in having the models *share certain parameters* defined a priori. Multi-task learning has a long history in machine learning and neural networks. Caruana (1997) gives a good overview of these past efforts.

5.1 Joint Decoding versus Joint Training

Multitask approaches do not necessarily involve joint training. For instance, modern speech recognition systems use Bayes rule to combine the outputs of an acoustic model trained on speech data and a language model trained on phonetic or textual corpora (Jelinek, 1976). This joint decoding approach has been successfully applied to structurally more complex NLP tasks. Sutton and McCallum (2005b) obtain improved results by combining the predictions of independently trained CRF models using a joint decoding process at test time that requires more sophisticated probabilistic

inference techniques. On the other hand, Sutton and McCallum (2005a) obtain results somewhat below the state-of-the-art using joint decoding for SRL and syntactic parsing. Musillo and Merlo (2006) also describe a negative result at the same joint task.

Joint decoding invariably works by considering additional probabilistic dependency paths between the models. Therefore it defines an implicit supermodel that describes all the tasks in the same probabilistic framework. Separately training a submodel only makes sense when the training data blocks these additional dependency paths (in the sense of d-separation, Pearl, 1988). This implies that, without joint training, the additional dependency paths cannot directly involve unobserved variables. Therefore, the natural idea of discovering common internal representations across tasks requires joint training.

Joint training is relatively straightforward when the training sets for the individual tasks contain the same patterns with different labels. It is then sufficient to train a model that computes multiple outputs for each pattern (Suddarth and Holden, 1991). Using this scheme, Sutton et al. (2007) demonstrate improvements on POS tagging and noun-phrase chunking using jointly trained CRFs. However the joint labeling requirement is a limitation because such data is not often available. Miller et al. (2000) achieves performance improvements by jointly training NER, parsing, and relation extraction in a statistical parsing model. The joint labeling requirement problem was weakened using a predictor to fill in the missing annotations.

Ando and Zhang (2005) propose a setup that works around the joint labeling requirements. They define linear models of the form $f_i(x) = w_i^\top \Phi(x) + v_i^\top \Theta\Psi(x)$ where f_i is the classifier for the i-th task with parameters w_i and v_i. Notations $\Phi(x)$ and $\Psi(x)$ represent engineered features for the pattern x. Matrix Θ maps the $\Psi(x)$ features into a low dimensional subspace common across all tasks. Each task is trained using its own examples without a joint labeling requirement. The learning procedure alternates the optimization of w_i and v_i for each task, and the optimization of Θ to minimize the average loss for all examples in all tasks. The authors also consider auxiliary unsupervised tasks for predicting substructures. They report excellent results on several tasks, including POS and NER.

5.2 Multi-Task Benchmark Results

Table 9 reports results obtained by jointly trained models for the POS, CHUNK, NER and SRL tasks using the same setup as Section 4.5. We trained jointly POS, CHUNK and NER using the window approach network. As we mentioned earlier, SRL can be trained only with the sentence approach network, due to long-range dependencies related to the verb predicate. We thus performed additional experiments, where all four tasks were trained using the sentence approach network. In both cases, all models share the lookup table parameters (2). The parameters of the first linear layers (4) were shared in the window approach case (see Figure 5), and the first the convolution layer parameters (6) were shared in the sentence approach networks.

For the window approach, best results were obtained by enlarging the first hidden layer size to $n^1_{hu} = 500$ (chosen by validation) in order to account for its shared responsibilities. We used the same architecture as SRL for the sentence approach network. The word embedding dimension was kept constant $d^0 = 50$ in order to reuse the language models of Section 4.5.

Training was achieved by minimizing the loss averaged across all tasks. This is easily achieved with stochastic gradient by alternatively picking examples for each task and applying (16) to all the parameters of the corresponding model, including the shared parameters. Note that this gives each task equal weight. Since each task uses the training sets described in Table 1, it is worth noticing

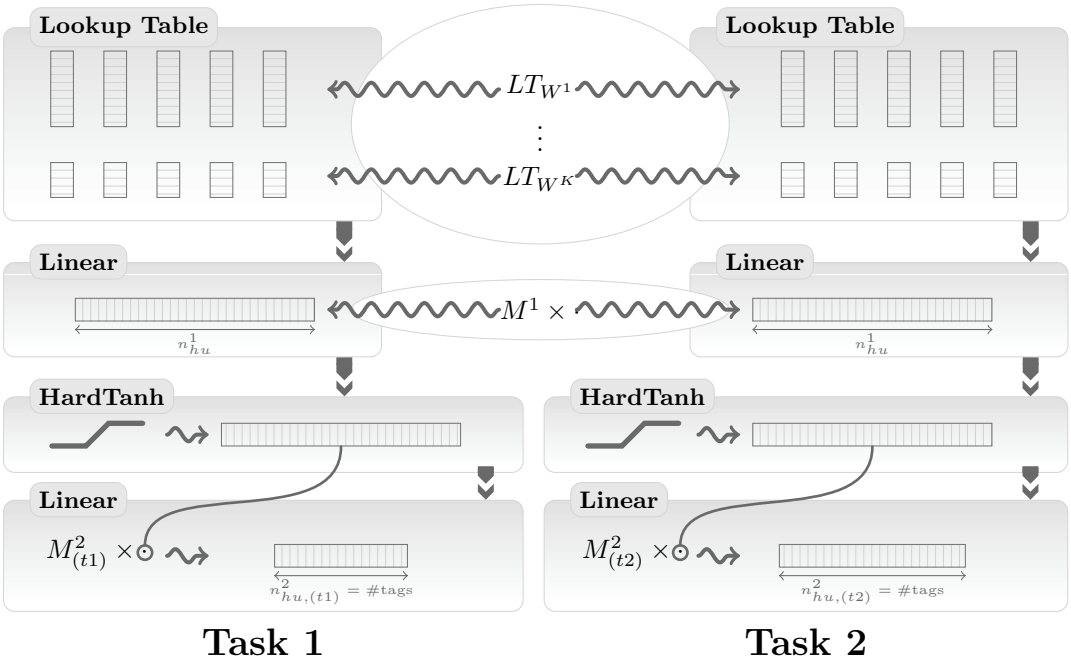


Figure 5: Example of multitasking with NN. Task 1 and Task 2 are two tasks trained with the window approach architecture presented in Figure 1. Lookup tables as well as the first hidden layer are shared. The last layer is task specific. The principle is the same with more than two tasks.

that examples can come from quite different data sets. The generalization performance for each task was measured using the traditional testing data specified in Table 1. Fortunately, none of the training and test sets overlap across tasks.

It is worth mentioning that MTL can produce a single *unified network* that performs well for all these tasks using the sentence approach. However this unified network only leads to marginal improvements over using a separate network for each task: the most important MTL task appears to be the unsupervised learning of the word embeddings. As explained before, simple computational considerations led us to train the POS, Chunking, and NER tasks using the window approach. The baseline results in Table 9 also show that using the sentence approach for the POS, Chunking, and NER tasks yields no performance improvement (or degradation) over the window approach. The next section shows we can leverage known correlations between tasks in more direct manner.

6. The Temptation

Results so far have been obtained by staying (almost[17]) true to our *from scratch* philosophy. We have so far avoided specializing our architecture for any task, disregarding a lot of useful *a priori*

17. We did some basic preprocessing of the raw input words as described in Section 3.5, hence the "almost" in the title of this article. A completely from scratch approach would presumably not know anything about words at all and would work from letters only (or, taken to a further extreme, from speech or optical character recognition, as humans do).

Approach	**POS** (PWA)	**CHUNK** (F1)	**NER** (F1)	**SRL** (F1)
Benchmark Systems	97.24	94.29	89.31	77.92
	Window Approach			
NN+SLL+LM2	97.20	93.63	88.67	–
NN+SLL+LM2+MTL	97.22	94.10	88.62	–
	Sentence Approach			
NN+SLL+LM2	97.12	93.37	88.78	74.15
NN+SLL+LM2+MTL	97.22	93.75	88.27	74.29

Table 9: Effect of multi-tasking on our neural architectures. We trained POS, CHUNK NER in a MTL way, both for the window and sentence network approaches. SRL was only included in the sentence approach joint training. As a baseline, we show previous results of our window approach system, as well as additional results for our sentence approach system, when trained separately on each task. Benchmark system performance is also given for comparison.

Approach	**POS** (PWA)	**CHUNK** (F1)	**NER** (F1)	**SRL**
Benchmark Systems	97.24	94.29	89.31	77.92
NN+SLL+LM2	97.20	93.63	88.67	74.15
NN+SLL+LM2+Suffix2	97.29	–	–	–
NN+SLL+LM2+Gazetteer	–	–	89.59	–
NN+SLL+LM2+POS	–	94.32	88.67	–
NN+SLL+LM2+CHUNK	–	–	–	74.72

Table 10: Comparison in generalization performance of benchmark NLP systems with our neural networks (NNs) using increasing task-specific engineering. We report results obtained with a network trained without the extra task-specific features (Section 5) and with the extra task-specific features described in Section 6. The POS network was trained with two character word suffixes; the NER network was trained using the small CoNLL 2003 gazetteer; the CHUNK and NER networks were trained with additional POS features; and finally, the SRL network was trained with additional CHUNK features.

NLP knowledge. We have shown that, thanks to large unlabeled data sets, our generic neural networks can still achieve close to state-of-the-art performance by discovering useful features. This section explores what happens when we increase the level of task-specific engineering in our systems by incorporating some common techniques from the NLP literature. We often obtain further improvements. These figures are useful to quantify how far we went by leveraging large data sets instead of relying on a priori knowledge.

6.1 Suffix Features

Word suffixes in many western languages are strong predictors of the syntactic function of the word and therefore can benefit the POS system. For instance, Ratnaparkhi (1996) uses inputs representing word suffixes and prefixes up to four characters. We achieve this in the POS task by adding discrete word features (Section 3.2.1) representing the last two characters of every word. The size of the suffix dictionary was 455. This led to a small improvement of the POS performance (Table 10, row NN+SLL+LM2+Suffix2). We also tried suffixes obtained with the Porter (1980) stemmer and obtained the same performance as when using two character suffixes.

6.2 Gazetteers

State-of-the-art NER systems often use a large dictionary containing well known named entities (e.g., Florian et al., 2003). We restricted ourselves to the gazetteer provided by the CoNLL challenge, containing 8,000 locations, person names, organizations, and miscellaneous entities. We trained a NER network with 4 additional word features indicating (feature "on" or "off") whether the word is found in the gazetteer under one of these four categories. The gazetteer includes not only words, but also chunks of words. If a sentence chunk is found in the gazetteer, then all words in the chunk have their corresponding gazetteer feature turned to "on". The resulting system displays a clear performance improvement (Table 10, row NN+SLL+LM2+Gazetteer), slightly outperforming the baseline. A plausible explanation of this large boost over the network using only the language model is that gazetteers include word chunks, while we use only the word representation of our language model. For example, "united" and "bicycle" seen separately are likely to be non-entities, while "united bicycle" might be an entity, but catching it would require higher level representations of our language model.

6.3 Cascading

When one considers related tasks, it is reasonable to assume that tags obtained for one task can be useful for taking decisions in the other tasks. Conventional NLP systems often use features obtained from the output of other preexisting NLP systems. For instance, Shen and Sarkar (2005) describe a chunking system that uses POS tags as input; Florian et al. (2003) describes a NER system whose inputs include POS and CHUNK tags, as well as the output of two other NER classifiers. State-of-the-art SRL systems exploit parse trees (Gildea and Palmer, 2002; Punyakanok et al., 2005), related to CHUNK tags, and built using POS tags (Charniak, 2000; Collins, 1999).

Table 10 reports results obtained for the CHUNK and NER tasks by adding discrete word features (Section 3.2.1) representing the POS tags. In order to facilitate comparisons, instead of using the more accurate tags from our POS network, we use for each task the POS tags provided by the corresponding CoNLL challenge. We also report results obtained for the SRL task by adding word features representing the CHUNK tags (also provided by the CoNLL challenge). We consistently obtain moderate improvements.

6.4 Ensembles

Constructing ensembles of classifiers is a proven way to trade computational efficiency for generalization performance (Bell et al., 2007). Therefore it is not surprising that many NLP systems achieve state-of-the-art performance by combining the outputs of multiple classifiers. For instance, Kudo

Approach		POS (PWA)	CHUNK (F1)	NER (F1)
Benchmark Systems		97.24	94.29	89.31
NN+SLL+LM2+POS	worst	97.29	93.99	89.35
NN+SLL+LM2+POS	mean	97.31	94.17	89.65
NN+SLL+LM2+POS	best	97.35	94.32	89.86
NN+SLL+LM2+POS	voting ensemble	97.37	94.34	89.70
NN+SLL+LM2+POS	joined ensemble	97.30	94.35	89.67

Table 11: Comparison in generalization performance for POS, CHUNK and NER tasks of the networks obtained using by combining ten training runs with different initialization.

and Matsumoto (2001) use an ensemble of classifiers trained with different tagging conventions (see Section 3.3.3). Winning a challenge is of course a legitimate objective. Yet it is often difficult to figure out which ideas are most responsible for the state-of-the-art performance of a large ensemble.

Because neural networks are nonconvex, training runs with different initial parameters usually give different solutions. Table 11 reports results obtained for the CHUNK and NER task after ten training runs with random initial parameters. Voting the ten network outputs on a per tag basis ("voting ensemble") leads to a small improvement over the average network performance. We have also tried a more sophisticated ensemble approach: the ten network output scores (before sentence-level likelihood) were combined with an additional linear layer (4) and then fed to a new sentence-level likelihood (13). The parameters of the combining layers were then trained on the existing training set, while keeping the ten networks fixed ("joined ensemble"). This approach did not improve on simple voting.

These ensembles come of course at the expense of a ten fold increase of the running time. On the other hand, multiple training times could be improved using smart sampling strategies (Neal, 1996).

We can also observe that the performance variability among the ten networks is not very large. The local minima found by the training algorithm are usually good local minima, thanks to the oversized parameter space and to the noise induced by the stochastic gradient procedure (LeCun et al., 1998). In order to reduce the variance in our experimental results, we always use the same initial parameters for networks trained on the same task (except of course for the results reported in Table 11.)

6.5 Parsing

Gildea and Palmer (2002) and Punyakanok et al. (2005) offer several arguments suggesting that syntactic parsing is a necessary prerequisite for the SRL task. The CoNLL 2005 SRL benchmark task provides parse trees computed using *both* the Charniak (2000) and Collins (1999) parsers. State-of-the-art systems often exploit additional parse trees such as the k top ranking parse trees (Koomen et al., 2005; Haghighi et al., 2005).

In contrast our SRL networks so far do not use parse trees at all. They rely instead on internal representations transferred from a language model trained with an objective function that captures

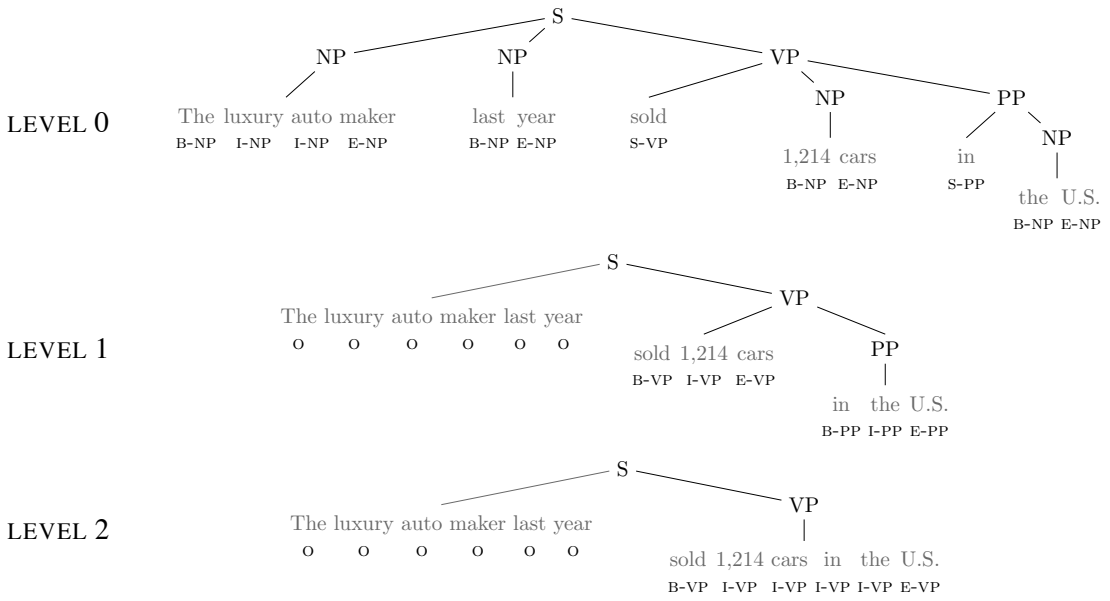


Figure 6: Charniak parse tree for the sentence *"The luxury auto maker last year sold 1,214 cars in the U.S."*. Level 0 is the original tree. Levels 1 to 4 are obtained by successively collapsing terminal tree branches. For each level, words receive tags describing the segment associated with the corresponding leaf. All words receive tag "O" at level 3 in this example.

a lot of syntactic information (see Section 4.6). It is therefore legitimate to question whether this approach is an acceptable lightweight replacement for parse trees.

We answer this question by providing parse tree information as additional input features to our system.[18] We have limited ourselves to the Charniak parse tree provided with the CoNLL 2005 data. Considering that a node in a syntactic parse tree assigns a label to a segment of the parsed sentence, we propose a way to feed (partially) this labeled segmentation to our network, through additional lookup tables. Each of these lookup tables encode *labeled* segments of each parse tree level (up to a certain depth). The labeled segments are fed to the network following a IOBES tagging scheme (see Sections 3.3.3 and 3.2.1). As there are 40 different phrase labels in WSJ, each additional tree-related lookup tables has 161 entries ($40 \times 4 + 1$) corresponding to the IBES segment tags, plus the extra O tag.

We call level 0 the information associated with the leaves of the original Charniak parse tree. The lookup table for level 0 encodes the corresponding IOBES phrase tags for each words. We obtain levels 1 to 4 by repeatedly trimming the leaves as shown in Figure 6. We labeled "O" words belonging to the root node "S", or all words of the sentence if the root itself has been trimmed.

Experiments were performed using the LM2 language model using the same network architectures (see Table 5) and using additional lookup tables of dimension 5 for each parse tree level. Table 12 reports the performance improvements obtained by providing increasing levels of parse

18. In a more recent work (Collobert, 2011), we propose an extension of this approach for the generation of full syntactic parse trees, using a recurrent version of our architecture.

Approach	SRL	
	(valid)	(test)
Benchmark System (six parse trees)	77.35	77.92
Benchmark System (top Charniak parse tree only)	74.76	–
NN+SLL+LM2	72.29	74.15
NN+SLL+LM2+Charniak (level 0 only)	74.44	75.65
NN+SLL+LM2+Charniak (levels 0 & 1)	74.50	75.81
NN+SLL+LM2+Charniak (levels 0 to 2)	75.09	76.05
NN+SLL+LM2+Charniak (levels 0 to 3)	75.12	75.89
NN+SLL+LM2+Charniak (levels 0 to 4)	75.42	76.06
NN+SLL+LM2+CHUNK	–	74.72
NN+SLL+LM2+PT0	–	75.49

Table 12: Generalization performance on the SRL task of our NN architecture compared with the benchmark system. We show performance of our system fed with different levels of depth of the Charniak parse tree. We report previous results of our architecture with no parse tree as a baseline. Koomen et al. (2005) report test and validation performance using six parse trees, as well as validation performance using only the top Charniak parse tree. For comparison purposes, we hence also report validation performance. Finally, we report our performance with the CHUNK feature, and compare it against a level 0 feature PT0 obtained by our network.

tree information. Level 0 alone increases the F1 score by almost 1.5%. Additional levels yield diminishing returns. The top performance reaches 76.06% F1 score. This is not too far from the state-of-the-art system which we note uses six parse trees instead of one. Koomen et al. (2005) also report a 74.76% F1 score on the validation set using only the Charniak parse tree. Using the first three parse tree levels, we reach this performance on the validation set. These results corroborate findings in the NLP literature (Gildea and Palmer, 2002; Punyakanok et al., 2005) showing that parsing is important for the SRL task.

We also reported in Table 12 our previous performance obtained with the CHUNK feature (see Table 10). It is surprising to observe that adding chunking features into the semantic role labeling network performs significantly worse than adding features describing the level 0 of the Charniak parse tree (Table 12). Indeed, if we ignore the label prefixes "BIES" defining the segmentation, the parse tree leaves (at level 0) and the chunking have identical labeling. However, the parse trees identify leaf sentence segments that are often smaller than those identified by the chunking tags, as shown by Hollingshead et al. (2005).[19] Instead of relying on Charniak parser, we chose to train a second chunking network to identify the segments delimited by the leaves of the Penn Treebank parse trees (level 0). Our network achieved 92.25% F1 score on this task (we call it PT0), while we evaluated Charniak performance as 91.94% on the same task. As shown in Table 12, feeding our

19. As in Hollingshead et al. (2005), consider the sentence and chunk labels "(NP They) (VP are starting to buy) (NP growth stocks)". The parse tree can be written as "(S (NP They) (VP are (VP starting (S (VP to (VP buy (NP growth stocks)))))))". The tree leaves segmentation is thus given by "(NP They) (VP are) (VP starting) (VP to) (VP buy) (NP growth stocks)".

own PT0 prediction into the SRL system gives similar performance to using Charniak predictions, and is consistently better than the CHUNK feature.

6.6 Word Representations

We have described how we induced useful word embeddings by applying our architecture to a language modeling task trained using a large amount of unlabeled text data. These embeddings improve the generalization performance on all tasks (Section 4.) The literature describes other ways to induce word representations. Mnih and Hinton (2007) proposed a related language model approach inspired from Restricted Boltzmann Machines. However, word representations are perhaps more commonly inferred from n-gram language modeling rather than purely continuous language models. One popular approach is the Brown clustering algorithm (Brown et al., 1992a), which builds hierarchical word clusters by maximizing the bigram's mutual information. The induced word representation has been used with success in a wide variety of NLP tasks, including POS (Schütze, 1995), NER (Miller et al., 2004; Ratinov and Roth, 2009), or parsing (Koo et al., 2008). Other related approaches exist, like phrase clustering (Lin and Wu, 2009) which has been shown to work well for NER. Finally, Huang and Yates (2009) have recently proposed a smoothed language modeling approach based on a Hidden Markov Model, with success on POS and Chunking tasks.

While a comparison of all these word representations is beyond the scope of this paper, it is rather fair to question the quality of our word embeddings compared to a popular NLP approach. In this section, we report a comparison of our word embeddings against Brown clusters, when used as features into our neural network architecture. We report as baseline previous results where our word embeddings are *fine-tuned* for each task. We also report performance when our embeddings are kept fixed during task-specific training. Since *convex* machine learning algorithms are common practice in NLP, we finally report performances for the convex version of our architecture.

For the convex experiments, we considered the linear version of our neural networks (instead of having several linear layers interleaved with a non-linearity). While we always picked the sentence approach for SRL, we had to consider the window approach in this particular convex setup, as the sentence approach network (see Figure 2) includes a Max layer. Having only one linear layer in our neural network is not enough to make our architecture convex: all lookup-tables (for each discrete feature) must also be *fixed*. The word-lookup table is simply fixed to the embeddings obtained from our language model LM2. All other discrete feature lookup-tables (caps, POS, Brown Clusters...) are fixed to a standard *sparse* representation. Using the notation introduced in Section 3.2.1, if LT_{W^k} is the lookup-table of the k^{th} discrete feature, we have $W^k \in \mathbb{R}^{|\mathcal{D}^k| \times |\mathcal{D}^k|}$ and the representation of the discrete input w is obtained with:

$$LT_{W^k}(w) = \langle W^k \rangle^1_w = \left(0, \cdots 0, \underset{\text{at index } w}{1}, 0, \cdots 0\right)^{\mathrm{T}}. \tag{18}$$

Training our architecture in this convex setup with the sentence-level likelihood (13) corresponds to training a CRF. In that respect, these convex experiments show the performance of our word embeddings in a classical NLP framework.

Following the Ratinov and Roth (2009) and Koo et al. (2008) setups, we generated 1,000 Brown clusters using the implementation[20] from Liang (2005). To make the comparison fair, the clusters were first induced on the concatenation of Wikipedia and Reuters data sets, as we did in Section 4

20. Available at `http://www.eecs.berkeley.edu/~pliang/software`.

Approach	**POS** (PWA)	**CHUNK** (F1)	**NER** (F1)	**SRL** (F1)
	Non-convex Approach			
LM2 (non-linear NN)	97.29	94.32	89.59	76.06
LM2 (non-linear NN, fixed embeddings)	97.10	94.45	88.79	72.24
Brown Clusters (non-linear NN, 130K words)	96.92	94.35	87.15	72.09
Brown Clusters (non-linear NN, all words)	96.81	94.21	86.68	71.44
	Convex Approach			
LM2 (linear NN, fixed embeddings)	96.69	93.51	86.64	59.11
Brown Clusters (linear NN, 130K words)	96.56	94.20	86.46	51.54
Brown Clusters (linear NN, all words)	96.28	94.22	86.63	56.42

Table 13: Generalization performance of our neural network architecture trained with our language model (LM2) word embeddings, and with the word representations derived from the Brown Clusters. As before, all networks are fed with a capitalization feature. Additionally, POS is using a word suffix of size 2 feature, CHUNK is fed with POS, NER uses the CoNLL 2003 gazetteer, and SRL is fed with levels 1–5 of the Charniak parse tree, as well as a verb position feature. We report performance with both convex and non-convex architectures (300 hidden units for all tasks, with an additional 500 hidden units layer for SRL). We also provide results for Brown Clusters induced with a 130K word dictionary, as well as Brown Clusters induced with all words of the given tasks.

for training our largest language model LM2, using a 130K word dictionary. This dictionary covers about 99% of the words in the training set of each task. To cover the last 1%, we augmented the dictionary with the missing words (reaching about 140K words) and induced Brown Clusters using the concatenation of WSJ, Wikipedia, and Reuters.

The Brown clustering approach is hierarchical and generates a binary tree of clusters. Each word in the vocabulary is assigned to a node in the tree. Features are extracted from this tree by considering the path from the root to the node containing the word of interest. Following Ratinov & Roth, we picked as features the path prefixes of size 4, 6, 10 and 20. In the non-convex experiments, we fed these four Brown Cluster features to our architecture using four different lookup tables, replacing our word lookup table. The size of the lookup tables was chosen to be 12 by validation. In the convex case, we used the classical sparse representation (18), as for any other discrete feature.

We first report in Table 13 generalization performance of our best non-convex networks trained with our LM2 language model and with Brown Cluster features. Our embeddings perform at least as well as Brown Clusters. Results are more mitigated in a convex setup. For most tasks, going non-convex is better for both word representation types. In general, "fine-tuning" our embeddings for each task also gives an extra boost. Finally, using a better word coverage with Brown Clusters ("all words" instead of "130K words" in Table 13) did not help.

More complex features could be possibly combined instead of using a non-linear model. For instance, Turian et al. (2010) performed a comparison of Brown Clusters and embeddings trained in the same spirit as ours[21], with additional features combining labels and tokens. We believe this

21. However they did not reach our embedding performance. There are several differences in how they trained their models that might explain this. Firstly, they may have experienced difficulties because they train 50-dimensional

Task	Features
POS	Suffix of size 2
CHUNK	POS
NER	CoNLL 2003 gazetteer
PT0	POS
SRL	PT0, verb position

Table 14: Features used by SENNA implementation, for each task. In addition, all tasks use "low caps word" and "caps" features.

Task		**Benchmark**	**SENNA**
Part of Speech (POS)	(Accuracy)	97.24 %	97.29 %
Chunking (CHUNK)	(F1)	94.29 %	94.32 %
Named Entity Recognition (NER)	(F1)	89.31 %	89.59 %
Parse Tree level 0 (PT0)	(F1)	91.94 %	92.25 %
Semantic Role Labeling (SRL)	(F1)	77.92 %	75.49 %

Table 15: Performance of the engineered sweet spot (SENNA) on various tagging tasks. The PT0 task replicates the sentence segmentation of the parse tree leaves. The corresponding benchmark score measures the quality of the Charniak parse tree leaves relative to the Penn Treebank gold parse trees.

type of comparison should be taken with care, as combining a given feature with different word representations might not have the same effect on each word representation.

6.7 Engineering a Sweet Spot

We implemented a standalone version of our architecture, written in the C language. We gave the name "SENNA" (Semantic/syntactic Extraction using a Neural Network Architecture) to the resulting system. The parameters of each architecture are the ones described in Table 5. All the networks were trained separately on each task using the sentence-level likelihood (SLL). The word embeddings were initialized to LM2 embeddings, and then fine-tuned for each task. We summarize features used by our implementation in Table 14, and we report performance achieved on each task in Table 15. The runtime version[22] contains about 2500 lines of C code, runs in less than 150MB of memory, and needs less than a millisecond per word to compute all the tags. Table 16 compares the tagging speeds for our system and for the few available state-of-the-art systems: the Toutanova et al. (2003) POS tagger[23], the Shen et al. (2007) POS tagger[24] and the Koomen et al. (2005) SRL

embeddings for 269K distinct words using a comparatively small training set (RCV1, 37M words), unlikely to contain enough instances of the rare words. Secondly, they predict the correctness of the final word of each window instead of the center word (Turian et al., 2010), effectively restricting the model to unidirectional prediction. Finally, they do not fine tune their embeddings after unsupervised training.

22. Available at `http://ml.nec-labs.com/senna`.
23. Available at `http://nlp.stanford.edu/software/tagger.shtml`. We picked the 3.0 version (May 2010).
24. Available at `http://www.cis.upenn.edu/~xtag/spinal`.

POS System	RAM (MB)	Time (s)
Toutanova et al. (2003)	800	64
Shen et al. (2007)	2200	833
SENNA	32	4

SRL System	RAM (MB)	Time (s)
Koomen et al. (2005)	3400	6253
SENNA	124	51

Table 16: Runtime speed and memory consumption comparison between state-of-the-art systems and our approach (SENNA). We give the runtime in seconds for running both the POS and SRL taggers on their respective testing sets. Memory usage is reported in megabytes.

system.[25] All programs were run on a single 3GHz Intel core. The POS taggers were run with Sun Java 1.6 with a large enough memory allocation to reach their top tagging speed. The beam size of the Shen tagger was set to 3 as recommended in the paper. Regardless of implementation differences, it is clear that our neural networks run considerably faster. They also require much less memory. Our POS and SRL tagger runs in 32MB and 120MB of RAM respectively. The Shen and Toutanova taggers slow down significantly when the Java machine is given less than 2.2GB and 800MB of RAM respectively, while the Koomen tagger requires at least 3GB of RAM.

We believe that a number of reasons explain the speed advantage of our system. First, our system only uses rather simple input features and therefore avoids the nonnegligible computation time associated with complex handcrafted features. Secondly, most network computations are *dense* matrix-vector operations. In contrast, systems that rely on a great number of *sparse* features experience memory latencies when traversing the sparse data structures. Finally, our compact implementation is self-contained. Since it does not rely on the outputs of disparate NLP system, it does not suffer from communication latency issues.

7. Critical Discussion

Although we believe that this contribution represents a step towards the "NLP from scratch" objective, we are keenly aware that both our goal and our means can be criticized.

The main criticism of our goal can be summarized as follows. Over the years, the NLP community has developed a considerable expertise in engineering effective NLP features. Why should they forget this painfully acquired expertise and instead painfully acquire the skills required to train large neural networks? As mentioned in our introduction, we observe that no single NLP task really covers the goals of NLP. Therefore we believe that task-specific engineering (i.e., that does not generalize to other tasks) is not desirable. But we also recognize how much our neural networks owe to previous NLP task-specific research.

The main criticism of our means is easier to address. Why did we choose to rely on a twenty year old technology, namely multilayer neural networks? We were simply attracted by their ability to discover hidden representations using a stochastic learning algorithm that scales linearly with

25. Available at `http://l2r.cs.uiuc.edu/~cogcomp/asoftware.php?skey=SRL`.

the number of examples. Most of the neural network technology necessary for our work has been described ten years ago (e.g., Le Cun et al., 1998). However, if we had decided ten years ago to train the language model network LM2 using a vintage computer, training would only be nearing completion today. Training algorithms that scale linearly are most able to benefit from such tremendous progress in computer hardware.

8. Conclusion

We have presented a multilayer neural network architecture that can handle a number of NLP tasks with both speed and accuracy. The design of this system was determined by our desire to avoid task-specific engineering as much as possible. Instead we rely on large unlabeled data sets and let the training algorithm discover internal representations that prove useful for all the tasks of interest. Using this strong basis, we have engineered a fast and efficient "all purpose" NLP tagger that we hope will prove useful to the community.

Acknowledgments

We acknowledge the persistent support of NEC for this research effort. We thank Yoshua Bengio, Samy Bengio, Eric Cosatto, Vincent Etter, Hans-Peter Graf, Ralph Grishman, and Vladimir Vapnik for their useful feedback and comments.

Appendix A. Neural Network Gradients

We consider a neural network $f_\theta(\cdot)$, with parameters θ. We maximize the likelihood (8), or minimize ranking criterion (17), with respect to the parameters θ, using stochastic gradient. By negating the likelihood, we now assume it all corresponds to minimize a cost $C(f_\theta(\cdot))$, with respect to θ.

Following the classical "back-propagation" derivations (LeCun, 1985; Rumelhart et al., 1986) and the modular approach shown in Bottou (1991), any feed-forward neural network with L layers, like the ones shown in Figure 1 and Figure 2, can be seen as a composition of functions $f^l_\theta(\cdot)$, corresponding to each layer l:

$$f_\theta(\cdot) = f^L_\theta(f^{L-1}_\theta(\ldots f^1_\theta(\cdot)\ldots))$$

Partitioning the parameters of the network with respect to each layers $1 \leq l \leq L$, we write:

$$\theta = (\theta^1, \ldots, \theta^l, \ldots, \theta^L).$$

We are now interested in computing the gradients of the cost with respect to each θ^l. Applying the chain rule (generalized to vectors) we obtain the classical backpropagation recursion:

$$\frac{\partial C}{\partial \theta^l} = \frac{\partial f^l_\theta}{\partial \theta^l} \frac{\partial C}{\partial f^l_\theta} \tag{19}$$

$$\frac{\partial C}{\partial f^{l-1}_\theta} = \frac{\partial f^l_\theta}{\partial f^{l-1}_\theta} \frac{\partial C}{\partial f^l_\theta}. \tag{20}$$

In other words, we first initialize the recursion by computing the gradient of the cost with respect to the last layer output $\partial C / \partial f^L_\theta$. Then each layer l computes the gradient respect to its own parameters

with (19), given the gradient coming from its output $\partial C/\partial f_\theta^l$. To perform the backpropagation, it also computes the gradient with respect to its own inputs, as shown in (20). We now derive the gradients for each layer we used in this paper.

A.1 Lookup Table Layer

Given a matrix of parameters $\theta^1 = W^1$ and word (or discrete feature) indices $[w]_1^T$, the layer outputs the matrix:

$$f_\theta^l([w]_l^T) = \left(\begin{array}{cccc} \langle W \rangle^1_{[w]_1} & \langle W \rangle^1_{[w]_2} & \cdots & \langle W \rangle^1_{[w]_T} \end{array} \right).$$

The gradients of the weights $\langle W \rangle_i^1$ are given by:

$$\frac{\partial C}{\partial \langle W \rangle_i^1} = \sum_{\{1 \leq t \leq T / [w]_t = i\}} \langle \frac{\partial C}{\partial f_\theta^l} \rangle_i^1$$

This sum equals zero if the index i in the lookup table does not corresponds to a word in the sequence. In this case, the i^{th} column of W does not need to be updated. As a Lookup Table Layer is always the first layer, we do not need to compute its gradients with respect to the inputs.

A.2 Linear Layer

Given parameters $\theta^l = (W^l, b^l)$, and an input *vector* f_θ^{l-1} the output is given by:

$$f_\theta^l = W^l f_\theta^{l-1} + b^l. \tag{21}$$

The gradients with respect to the parameters are then obtained with:

$$\frac{\partial C}{\partial W^l} = \left[\frac{\partial C}{\partial f_\theta^l} \right] \left[f_\theta^{l-1} \right]^{\text{T}} \text{ and } \frac{\partial C}{\partial b^l} = \frac{\partial C}{\partial f_\theta^l}, \tag{22}$$

and the gradients with respect to the inputs are computed with:

$$\frac{\partial C}{\partial f_\theta^{l-1}} = \left[W^l \right]^{\text{T}} \frac{\partial C}{\partial f_\theta^l}. \tag{23}$$

A.3 Convolution Layer

Given a input *matrix* f_θ^{l-1}, a Convolution Layer $f_\theta^l(\cdot)$ applies a Linear Layer operation (21) successively on each window $\langle f_\theta^{l-1} \rangle_t^{d_{win}}$ $(1 \leq t \leq T)$ of size d_{win}. Using (22), the gradients of the parameters are thus given by summing over all windows:

$$\frac{\partial C}{\partial W^l} = \sum_{t=1}^{T} \left[\langle \frac{\partial C}{\partial f_\theta^l} \rangle_t^1 \right] \left[\langle f_\theta^{l-1} \rangle_t^{d_{win}} \right]^{\text{T}} \text{ and } \frac{\partial C}{\partial b^l} = \sum_{t=1}^{T} \langle \frac{\partial C}{\partial f_\theta^l} \rangle_t^1.$$

After initializing the input gradients $\partial C/\partial f_\theta^{l-1}$ to zero, we iterate (23) over all windows for $1 \leq t \leq T$, leading the *accumulation*[26]:

$$\langle \frac{\partial C}{\partial f_\theta^{l-1}} \rangle_t^{d_{win}} \mathrel{+}= \left[W^l \right]^{\text{T}} \langle \frac{\partial C}{\partial f_\theta^l} \rangle_t^1.$$

26. We denote "+=" any accumulation operation.

A.4 Max Layer

Given a *matrix* f_θ^{l-1}, the Max Layer computes

$$\left[f_\theta^l\right]_i = \max_t \left[\langle f_\theta^{l-1}\rangle_t^1\right]_i \text{ and } a_i = \operatorname*{argmax}_t \left[\langle f_\theta^{l-1}\rangle_t^1\right]_i \ \forall i,$$

where a_i stores the index of the largest value. We only need to compute the gradient with respect to the inputs, as this layer has no parameters. The gradient is given by

$$\left[\langle \frac{\partial C}{\partial f_\theta^{l-1}}\rangle_t^1\right]_i = \begin{cases} \left[\langle \frac{\partial C}{\partial f_\theta^l}\rangle_t^1\right]_i & \text{if } t = a_i \\ 0 & \text{otherwise} \end{cases}.$$

A.5 HardTanh Layer

Given a *vector* f_θ^{l-1}, and the definition of the HardTanh (5) we get

$$\left[\frac{\partial C}{\partial f_\theta^{l-1}}\right]_i = \begin{cases} 0 & \text{if } \left[f_\theta^{l-1}\right]_i < -1 \\ \left[\frac{\partial C}{\partial f_\theta^l}\right]_i & \text{if } -1 <= \left[f_\theta^{l-1}\right]_i <= 1 \\ 0 & \text{if } \left[f_\theta^{l-1}\right]_i > 1 \end{cases},$$

if we ignore non-differentiability points.

A.6 Word-Level Log-Likelihood

The network outputs a score $[f_\theta]_i$ for each tag indexed by i. Following (11), if y is the true tag for a given example, the stochastic score to minimize can be written as

$$C(f_\theta) = \operatorname*{logadd}_j [f_\theta]_j - [f_\theta]_y$$

Considering the definition of the logadd (10), the gradient with respect to f_θ is given by

$$\frac{\partial C}{\partial [f_\theta]_i} = \frac{e^{[f_\theta]_i}}{\sum_k e^{[f_\theta]_k}} - 1_{i=y} \quad \forall i.$$

A.7 Sentence-Level Log-Likelihood

The network outputs a matrix where each element $\left[f_\theta\right]_{i,t}$ gives a score for tag i at word t. Given a tag sequence $[y]_1^T$ and a input sequence $[x]_1^T$, we maximize the likelihood (13), which corresponds to minimizing the score

$$C(f_\theta, A) = \underbrace{\operatorname*{logadd}_{\forall [j]_1^T} s([x]_1^T, [j]_1^T, \tilde{\theta})}_{C_{logadd}} - s([x]_1^T, [y]_1^T, \tilde{\theta}),$$

with

$$s([x]_1^T, [y]_1^T, \tilde{\theta}) = \sum_{t=1}^T \left([A]_{[y]_{t-1}, [y]_t} + [f_\theta]_{[y]_t, t} \right).$$

We first initialize all gradients to zero

$$\frac{\partial C}{\partial \left[f_\theta\right]_{i,t}} = 0 \; \forall i,t \;\text{ and }\; \frac{\partial C}{\partial \left[A\right]_{i,j}} = 0 \quad \forall i,j\,.$$

We then *accumulate* gradients over the second part of the cost $-s([x]_1^T, [y]_1^T, \tilde{\theta})$, which gives:

$$\begin{array}{l} \dfrac{\partial C}{\partial \left[f_\theta\right]_{[y]_t,t}} \mathrel{+}= 1 \\ \dfrac{\partial C}{\partial \left[A\right]_{[y]_{t-1},[y]_t}} \mathrel{+}= 1 \end{array} \qquad \forall t\,.$$

We now need to accumulate the gradients over the first part of the cost, that is C_{logadd}. We differentiate C_{logadd} by applying the chain rule through the recursion (14). First we initialize our recursion with

$$\frac{\partial C_{logadd}}{\partial \delta_T(i)} = \frac{e^{\delta_T(i)}}{\sum_k e^{\delta_T(k)}} \quad \forall i\,.$$

We then compute iteratively:

$$\frac{\partial C_{logadd}}{\partial \delta_{t-1}(i)} = \sum_j \frac{\partial C_{logadd}}{\partial \delta_t(j)} \frac{e^{\delta_{t-1}(i)+[A]_{i,j}}}{\sum_k e^{\delta_{t-1}(k)+[A]_{k,j}}}\,,$$

where at each step t of the recursion we accumulate of the gradients with respect to the inputs f_θ, and the transition scores $[A]_{i,j}$:

$$\begin{array}{ll} \dfrac{\partial C}{\partial \left[f_\theta\right]_{i,t}} \mathrel{+}= \dfrac{\partial C_{logadd}}{\partial \delta_t(i)} \dfrac{\partial \delta_t(i)}{\partial \left[f_\theta\right]_{i,t}} & = \dfrac{\partial C_{logadd}}{\partial \delta_t(i)} \\ \dfrac{\partial C}{\partial \left[A\right]_{i,j}} \mathrel{+}= \dfrac{\partial C_{logadd}}{\partial \delta_t(j)} \dfrac{\partial \delta_t(j)}{\partial \left[A\right]_{i,j}} & = \dfrac{\partial C_{logadd}}{\partial \delta_t(j)} \dfrac{e^{\delta_{t-1}(i)+[A]_{i,j}}}{\sum_k e^{\delta_{t-1}(k)+[A]_{k,j}}}\,. \end{array}$$

A.8 Ranking Criterion

We use the ranking criterion (17) for training our language model. In this case, given a "positive" example x and a "negative" example $x^{(w)}$, we want to minimize:

$$C(f_\theta(x), f_\theta(x^w)) = \max\left\{0\,,\, 1 - f_\theta(x) + f_\theta(x^{(w)})\right\}.$$

Ignoring the non-differentiability of $max(0,\cdot)$ in zero, the gradient is simply given by:

$$\begin{pmatrix} \frac{\partial C}{\partial f_\theta(x)} \\ \frac{\partial C}{\partial f_\theta(x^w)} \end{pmatrix} = \begin{cases} \begin{pmatrix} -1 \\ 1 \end{pmatrix} & \text{if } 1 - f_\theta(x) + f_\theta(x^{(w)}) > 0 \\ \begin{pmatrix} 0 \\ 0 \end{pmatrix} & \text{otherwise} \end{cases}.$$

References

R. K. Ando and T. Zhang. A framework for learning predictive structures from multiple tasks and unlabeled data. *Journal of Machine Learning Research (JMLR)*, 6:1817–1953, 2005.

R. M. Bell, Y. Koren, and C. Volinsky. The BellKor solution to the Netflix Prize. Technical report, AT&T Labs, 2007. `http://www.research.att.com/~volinsky/netflix`.

Y. Bengio and R. Ducharme. A neural probabilistic language model. In *Advances in Neural Information Processing Systems (NIPS 13)*, 2001.

Y. Bengio, P. Lamblin, D. Popovici, and H. Larochelle. Greedy layer-wise training of deep networks. In *Advances in Neural Information Processing Systems (NIPS 19)*, 2007.

Y. Bengio, J. Louradour, R. Collobert, and J. Weston. Curriculum learning. In *International Conference on Machine Learning (ICML)*, 2009.

L. Bottou. Stochastic gradient learning in neural networks. In *Proceedings of Neuro-Nîmes*. EC2, 1991.

L. Bottou. Online algorithms and stochastic approximations. In David Saad, editor, *Online Learning and Neural Networks*. Cambridge University Press, Cambridge, UK, 1998.

L. Bottou and P. Gallinari. A framework for the cooperation of learning algorithms. In *Advances in Neural Information Processing Systems (NIPS 3)*. 1991.

L. Bottou, Y. LeCun, and Yoshua Bengio. Global training of document processing systems using graph transformer networks. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 489–493, 1997.

J. S. Bridle. Probabilistic interpretation of feedforward classification network outputs, with relationships to statistical pattern recognition. In F. Fogelman Soulié and J. Hérault, editors, *Neurocomputing: Algorithms, Architectures and Applications*, pages 227–236. NATO ASI Series, 1990.

P. F. Brown, P. V. deSouza, R. L. Mercer, V. J. D. Pietra, and J C. Lai. Class-based n-gram models of natural language. *Computational Linguistics*, 18(4):467–479, 1992a.

P. F. Brown, V. J. Della Pietra, R. L. Mercer, S. A. Della Pietra, and J. C. Lai. An estimate of an upper bound for the entropy of english. *Computational Linguistics*, 18(1):31–41, 1992b.

C. J. C. Burges, R. Ragno, and Quoc Viet Le. Learning to rank with nonsmooth cost functions. In *Advances in Neural Information Processing Systems (NIPS 19)*, pages 193–200. 2007.

R. Caruana. Multitask Learning. *Machine Learning*, 28(1):41–75, 1997.

O. Chapelle, B. Schlkopf, and A. Zien. *Semi-Supervised Learning*. Adaptive computation and machine learning. MIT Press, Cambridge, Mass., USA, September 2006.

E. Charniak. A maximum-entropy-inspired parser. In *Conference of the North American Chapter of the Association for Computational Linguistics & Human Language Technologies (NAACL-HLT)*, pages 132–139, 2000.

H. L. Chieu. Named entity recognition with a maximum entropy approach. In *Conference on Natural Language Learning (CoNLL)*, pages 160–163, 2003.

N. Chomsky. Three models for the description of language. *IRE Transactions on Information Theory*, 2(3):113–124, September 1956.

S. Clémençon and N. Vayatis. Ranking the best instances. *Journal of Machine Learning Research (JMLR)*, 8:2671–2699, 2007.

W. W. Cohen, R. E. Schapire, and Y. Singer. Learning to order things. *Journal of Artificial Intelligence Research (JAIR)*, 10:243–270, 1998.

T. Cohn and P. Blunsom. Semantic role labelling with tree conditional random fields. In *Conference on Computational Natural Language (CoNLL)*, 2005.

M. Collins. *Head-Driven Statistical Models for Natural Language Parsing*. PhD thesis, University of Pennsylvania, 1999.

R. Collobert. *Large Scale Machine Learning*. PhD thesis, Université Paris VI, 2004.

R. Collobert. Deep learning for efficient discriminative parsing. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2011.

T. Cover and R. King. A convergent gambling estimate of the entropy of english. *IEEE Transactions on Information Theory*, 24(4):413–421, July 1978.

R. Florian, A. Ittycheriah, H. Jing, and T. Zhang. Named entity recognition through classifier combination. In *Conference of the North American Chapter of the Association for Computational Linguistics & Human Language Technologies (NAACL-HLT)*, pages 168–171, 2003.

D. Gildea and D. Jurafsky. Automatic labeling of semantic roles. *Computational Linguistics*, 28(3): 245–288, 2002.

D. Gildea and M. Palmer. The necessity of parsing for predicate argument recognition. *Meeting of the Association for Computational Linguistics (ACL)*, pages 239–246, 2002.

J. Giménez and L. Màrquez. SVMTool: A general POS tagger generator based on support vector machines. In *Conference on Language Resources and Evaluation (LREC)*, 2004.

A. Haghighi, K. Toutanova, and C. D. Manning. A joint model for semantic role labeling. In *Conference on Computational Natural Language Learning (CoNLL)*, June 2005.

Z. S. Harris. *Mathematical Structures of Language*. John Wiley & Sons Inc., 1968.

D. Heckerman, D. M. Chickering, C. Meek, R. Rounthwaite, and C. Kadie. Dependency networks for inference, collaborative filtering, and data visualization. *Journal of Machine Learning Research (JMLR)*, 1:49–75, 2001.

G. E. Hinton, S. Osindero, and Y.-W. Teh. A fast learning algorithm for deep belief nets. *Neural Computation*, 18(7):1527–1554, July 2006.

K. Hollingshead, S. Fisher, and B. Roark. Comparing and combining finite-state and context-free parsers. In *Conference on Human Language Technology and Empirical Methods in Natural Language Processing (HLT-EMNLP)*, pages 787–794, 2005.

F. Huang and A. Yates. Distributional representations for handling sparsity in supervised sequence-labeling. In *Meeting of the Association for Computational Linguistics (ACL)*, pages 495–503, 2009.

F. Jelinek. Continuous speech recognition by statistical methods. *Proceedings of the IEEE*, 64(4): 532–556, 1976.

T. Joachims. Transductive inference for text classification using support vector machines. In *International Conference on Machine learning (ICML)*, 1999.

D. Klein and C. D. Manning. Natural language grammar induction using a constituent-context model. In *Advances in Neural Information Processing Systems (NIPS 14)*, pages 35–42. 2002.

T. Koo, X. Carreras, and M. Collins. Simple semi-supervised dependency parsing. In *Meeting of the Association for Computational Linguistics (ACL)*, pages 595–603, 2008.

P. Koomen, V. Punyakanok, D. Roth, and W. Yih. Generalized inference with multiple semantic role labeling systems (shared task paper). In *Conference on Computational Natural Language Learning (CoNLL)*, pages 181–184, 2005.

T. Kudo and Y. Matsumoto. Chunking with support vector machines. In *Conference of the North American Chapter of the Association for Computational Linguistics & Human Language Technologies (NAACL-HLT)*, pages 1–8, 2001.

T. Kudoh and Y. Matsumoto. Use of support vector learning for chunk identification. In *Conference on Natural Language Learning (CoNLL) and Second Learning Language in Logic Workshop (LLL)*, pages 142–144, 2000.

J. Lafferty, A. McCallum, and F. Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *International Conference on Machine Learning (ICML)*, 2001.

Y. Le Cun, L. Bottou, Y. Bengio, and P. Haffner. Gradient based learning applied to document recognition. *Proceedings of IEEE*, 86(11):2278–2324, 1998.

Y. LeCun. A learning scheme for asymmetric threshold networks. In *Proceedings of Cognitiva*, pages 599–604, Paris, France, 1985.

Y. LeCun, L. Bottou, G. B. Orr, and K.-R. Müller. Efficient backprop. In G.B. Orr and K.-R. Müller, editors, *Neural Networks: Tricks of the Trade*, pages 9–50. Springer, 1998.

D. D. Lewis, Y. Yang, T. G. Rose, and F. Li. Rcv1: A new benchmark collection for text categorization research. *Journal of Machine Learning Research (JMLR)*, 5:361–397, 2004.

P. Liang. Semi-supervised learning for natural language. Master's thesis, Massachusetts Institute of Technology, 2005.

P. Liang, H. Daumé, III, and D. Klein. Structure compilation: trading structure for features. In *International Conference on Machine learning (ICML)*, pages 592–599, 2008.

D. Lin and X. Wu. Phrase clustering for discriminative learning. In *Meeting of the Association for Computational Linguistics (ACL)*, pages 1030–1038, 2009.

N. Littlestone. Learning quickly when irrelevant attributes abound: A new linear-threshold algorithm. In *Machine Learning*, pages 285–318, 1988.

A. McCallum and Wei Li. Early results for named entity recognition with conditional random fields, feature induction and web-enhanced lexicons. In *Conference of the North American Chapter of the Association for Computational Linguistics & Human Language Technologies (NAACL-HLT)*, pages 188–191, 2003.

D. McClosky, E. Charniak, and M. Johnson. Effective self-training for parsing. *Conference of the North American Chapter of the Association for Computational Linguistics & Human Language Technologies (NAACL-HLT)*, 2006.

R. McDonald, K. Crammer, and F. Pereira. Flexible text segmentation with structured multilabel classification. In *Conference on Human Language Technology and Empirical Methods in Natural Language Processing (HLT-EMNLP)*, pages 987–994, 2005.

S. Miller, H. Fox, L. Ramshaw, and R. Weischedel. A novel use of statistical parsing to extract information from text. *Applied Natural Language Processing Conference (ANLP)*, 2000.

S. Miller, J. Guinness, and A. Zamanian. Name tagging with word clusters and discriminative training. In *Conference of the North American Chapter of the Association for Computational Linguistics & Human Language Technologies (NAACL-HLT)*, pages 337–342, 2004.

A Mnih and G. E. Hinton. Three new graphical models for statistical language modelling. In *International Conference on Machine Learning (ICML)*, pages 641–648, 2007.

G. Musillo and P. Merlo. Robust Parsing of the Proposition Bank. *ROMAND 2006: Robust Methods in Analysis of Natural language Data*, 2006.

R. M. Neal. *Bayesian Learning for Neural Networks*. Number 118 in Lecture Notes in Statistics. Springer-Verlag, New York, 1996.

D. Okanohara and J. Tsujii. A discriminative language model with pseudo-negative samples. *Meeting of the Association for Computational Linguistics (ACL)*, pages 73–80, 2007.

M. Palmer, D. Gildea, and P. Kingsbury. The proposition bank: An annotated corpus of semantic roles. *Computational Linguistics*, 31(1):71–106, 2005.

J. Pearl. *Probabilistic Reasoning in Intelligent Systems*. Morgan Kaufman, San Mateo, 1988.

D. C. Plaut and G. E. Hinton. Learning sets of filters using back-propagation. *Computer Speech and Language*, 2:35–61, 1987.

M. F. Porter. An algorithm for suffix stripping. *Program*, 14(3):130–137, 1980.

S. Pradhan, W. Ward, K. Hacioglu, J. Martin, and D. Jurafsky. Shallow semantic parsing using support vector machines. *Conference of the North American Chapter of the Association for Computational Linguistics & Human Language Technologies (NAACL-HLT)*, 2004.

S. Pradhan, K. Hacioglu, W. Ward, J. H. Martin, and D. Jurafsky. Semantic role chunking combining complementary syntactic views. In *Conference on Computational Natural Language Learning (CoNLL)*, pages 217–220, 2005.

V. Punyakanok, D. Roth, and W. Yih. The necessity of syntactic parsing for semantic role labeling. In *International Joint Conference on Artificial Intelligence (IJCAI)*, pages 1117–1123, 2005.

L. R. Rabiner. A tutorial on hidden Markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286, 1989.

L. Ratinov and D. Roth. Design challenges and misconceptions in named entity recognition. In *Conference on Computational Natural Language Learning (CoNLL)*, pages 147–155. Association for Computational Linguistics, 2009.

A. Ratnaparkhi. A maximum entropy model for part-of-speech tagging. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 133–142, 1996.

B. Rosenfeld and R. Feldman. Using Corpus Statistics on Entities to Improve Semi-supervised Relation Extraction from the Web. *Meeting of the Association for Computational Linguistics (ACL)*, pages 600–607, 2007.

D. E. Rumelhart, G. E. Hinton, and R. J. Williams. Learning internal representations by back-propagating errors. In D.E. Rumelhart and J. L. McClelland, editors, *Parallel Distributed Processing: Explorations in the Microstructure of Cognition*, volume 1, pages 318–362. MIT Press, 1986.

H. Schütze. Distributional part-of-speech tagging. In *Meeting of the Association for Computational Linguistics (ACL)*, pages 141–148, 1995.

H. Schwenk and J. L. Gauvain. Connectionist language modeling for large vocabulary continuous speech recognition. In *International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pages 765–768, 2002.

F. Sha and F. Pereira. Shallow parsing with conditional random fields. In *Conference of the North American Chapter of the Association for Computational Linguistics & Human Language Technologies (NAACL-HLT)*, pages 134–141, 2003.

C. E. Shannon. Prediction and entropy of printed english. *Bell Systems Technical Journal*, 30: 50–64, 1951.

H. Shen and A. Sarkar. Voting between multiple data representations for text chunking. *Advances in Artificial Intelligence*, pages 389–400, 2005.

L. Shen, G. Satta, and A. K. Joshi. Guided learning for bidirectional sequence classification. In *Meeting of the Association for Computational Linguistics (ACL)*, 2007.

N. A. Smith and J. Eisner. Contrastive estimation: Training log-linear models on unlabeled data. In *Meeting of the Association for Computational Linguistics (ACL)*, pages 354–362, 2005.

S. C. Suddarth and A. D. C. Holden. Symbolic-neural systems and the use of hints for developing complex systems. *International Journal of Man-Machine Studies*, 35(3):291–311, 1991.

X. Sun, L.-P. Morency, D. Okanohara, and J. Tsujii. Modeling latent-dynamic in shallow parsing: a latent conditional model with improved inference. In *International Conference on Computational Linguistics (COLING)*, pages 841–848, 2008.

C. Sutton and A. McCallum. Joint parsing and semantic role labeling. In *Conference on Computational Natural Language (CoNLL)*, pages 225–228, 2005a.

C. Sutton and A. McCallum. Composition of conditional random fields for transfer learning. *Conference on Human Language Technology and Empirical Methods in Natural Language Processing (HLT-EMNLP)*, pages 748–754, 2005b.

C. Sutton, A. McCallum, and K. Rohanimanesh. Dynamic Conditional Random Fields: Factorized Probabilistic Models for Labeling and Segmenting Sequence Data. *Journal of Machine Learning Research (JMLR)*, 8:693–723, 2007.

J. Suzuki and H. Isozaki. Semi-supervised sequential labeling and segmentation using giga-word scale unlabeled data. In *Conference of the North American Chapter of the Association for Computational Linguistics & Human Language Technologies (NAACL-HLT)*, pages 665–673, 2008.

W. J. Teahan and J. G. Cleary. The entropy of english using ppm-based models. In *Data Compression Conference (DCC)*, pages 53–62. IEEE Computer Society Press, 1996.

K. Toutanova, D. Klein, C. D. Manning, and Y. Singer. Feature-rich part-of-speech tagging with a cyclic dependency network. In *Conference of the North American Chapter of the Association for Computational Linguistics & Human Language Technologies (NAACL-HLT)*, 2003.

J. Turian, L. Ratinov, and Y. Bengio. Word representations: A simple and general method for semi-supervised learning. In *Meeting of the Association for Computational Linguistics (ACL)*, pages 384–392, 2010.

N. Ueffing, G. Haffari, and A. Sarkar. Transductive learning for statistical machine translation. In *Meeting of the Association for Computational Linguistics (ACL)*, pages 25–32, 2007.

A. Waibel, T. Hanazawa, G. Hinton, K. Shikano, and K.J. Lang. Phoneme recognition using time-delay neural networks. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 37(3): 328–339, 1989.

J. Weston, F. Ratle, and R. Collobert. Deep learning via semi-supervised embedding. In *International Conference on Machine learning (ICML)*, pages 1168–1175, 2008.

Journal of Machine Learning Research 12 (2011) 2539-2561 Submitted 5/10; Revised 6/11; Published 9/11

Weisfeiler-Lehman Graph Kernels

Nino Shervashidze NINO.SHERVASHIDZE@TUEBINGEN.MPG.DE
Machine Learning & Computational Biology Research Group
Max Planck Institutes Tübingen
Spemannstr. 38
72076 Tübingen, Germany

Pascal Schweitzer PASCAL@MPI-INF.MPG.DE
Max Planck Institute for Informatics
Campus E1 4
66123 Saarbrücken, Germany

Erik Jan van Leeuwen E.J.VAN.LEEUWEN@II.UIB.NO
Department of Informatics
University of Bergen
Postboks 7803
N-5020 Bergen, Norway

Kurt Mehlhorn MEHLHORN@MPI-INF.MPG.DE
Max Planck Institute for Informatics
Campus E1 4
66123 Saarbrücken, Germany

Karsten M. Borgwardt KARSTEN.BORGWARDT@TUEBINGEN.MPG.DE
Machine Learning & Computational Biology Research Group
Max Planck Institutes Tübingen
Spemannstr. 38
72076 Tübingen, Germany

Editor: Francis Bach

Abstract

In this article, we propose a family of efficient kernels for large graphs with discrete node labels. Key to our method is a rapid feature extraction scheme based on the Weisfeiler-Lehman test of isomorphism on graphs. It maps the original graph to a sequence of graphs, whose node attributes capture topological and label information. A family of kernels can be defined based on this Weisfeiler-Lehman sequence of graphs, including a highly efficient kernel comparing subtree-like patterns. Its runtime scales only linearly in the number of edges of the graphs and the length of the Weisfeiler-Lehman graph sequence. In our experimental evaluation, our kernels outperform state-of-the-art graph kernels on several graph classification benchmark data sets in terms of accuracy and runtime. Our kernels open the door to large-scale applications of graph kernels in various disciplines such as computational biology and social network analysis.

Keywords: graph kernels, graph classification, similarity measures for graphs, Weisfeiler-Lehman algorithm

1. Introduction

Graph-structured data is becoming more and more abundant: examples are social networks, protein or gene regulation networks, chemical pathways and protein structures, or the growing body of research in program flow analysis. To analyze and understand this data, one needs data analysis and machine learning methods that can handle large-scale graph data sets. For instance, a typical problem of learning on graphs arises in chemoinformatics: In this problem one is given a large set of chemical compounds, represented as node- and edge-labeled graphs, that have a certain function (e.g., mutagenicity or toxicity) and another set of molecules that do not have this function. The task then is to accurately predict whether a new, previously unseen molecule will exhibit this function or not. A common assumption made in this problem is that molecules with similar structure have similar functional properties. The problem of measuring the similarity of graphs is therefore at the core of learning on graphs.

There exist many graph similarity measures based on graph isomorphism or related concepts such as subgraph isomorphism or the largest common subgraph. Possibly the most natural measure of similarity of graphs is to check whether the graphs are topologically identical, that is, isomorphic. This gives rise to a binary similarity measure, which equals 1 if the graphs are isomorphic, and 0 otherwise. Despite the idea of checking graph isomorphism being so intuitive, no efficient algorithms are known for it. The graph isomorphism problem is in NP, but has been neither proven NP-complete nor found to be solved by a polynomial-time algorithm (Garey and Johnson, 1979, Chapter 7).

Subgraph isomorphism checking is the analogue of graph isomorphism checking in a setting in which the two graphs have different sizes. Unlike the graph isomorphism problem, the problem of subgraph isomorphism has been proven to be NP-complete (Garey and Johnson, 1979, Section 3.2.1). A slightly less restrictive measure of similarity can be defined based on the size of the largest common subgraph in two graphs, but unfortunately the problem of finding the largest common subgraph of two graphs is NP-complete as well (Garey and Johnson, 1979, Section 3.3).

Besides being computationally expensive or even intractable, similarity measures based on graph isomorphism and its variants are too restrictive in the sense that graphs have to be exactly identical or contain large identical subgraphs in order to be deemed similar by these measures. More flexible similarity measures, based on inexact matching of graphs, have been proposed in the literature. Graph comparison methods based on graph edit distances (Bunke and Allermann, 1983; Neuhaus and Bunke, 2005) are expressive similarity measures respecting the topology, as well as node and edge labels of graphs, but they are hard to parameterize and involve solving NP-complete problems as intermediate steps. Another type of graph similarity measures, optimal assignment kernels (Fröhlich et al., 2005), arise from finding the best match between substructures of graphs. However, these kernels are not positive semidefinite in general (Vert, 2008).

Recently proposed group theoretical approaches for representing graphs, the skew spectrum (Kondor and Borgwardt, 2008) and the graphlet spectrum (Kondor et al., 2009) can also be used for defining similarity measures on graphs that are computable in polynomial time. However, the skew spectrum is restricted to unlabeled graphs, while the graphlet spectrum can be difficult to parameterize on general labeled graphs.

Graph kernels have recently evolved into a rapidly developing branch of learning on structured data. They respect and exploit graph topology, but restrict themselves to comparing substructures of graphs that are computable in polynomial time. Graph kernels bridge the gap between

graph-structured data and a large spectrum of machine learning algorithms called kernel methods (Schölkopf and Smola, 2002), that include algorithms such as support vector machines, kernel regression, or kernel PCA (see Hofmann et al., 2008, for a recent review of kernel algorithms).

Informally, a kernel is a function of two objects that quantifies their similarity. Mathematically, it corresponds to an inner product in a reproducing kernel Hilbert space (Schölkopf and Smola, 2002). Graph kernels are instances of the family of so-called R-convolution kernels by Haussler (1999). *R*-convolution is a generic way of defining kernels on discrete compound objects by comparing all pairs of decompositions thereof. Therefore, a new type of decomposition of a graph results in a new graph kernel.

Given a decomposition relation R that decomposes a graph into any of its subgraphs and the remaining part of the graph, the associated R-convolution kernel will compare all subgraphs in two graphs. However, this *all subgraphs* kernel is at least as hard to compute as deciding if graphs are isomorphic (Gärtner et al., 2003). Therefore one usually restricts graph kernels to compare only specific types of subgraphs that are computable in polynomial runtime.

1.1 Review of Graph Kernels

Before we review graph kernels from the literature, we clarify our terminology. We define a graph G as a triplet (V, E, ℓ), where V is the set of vertices, E is the set of undirected edges, and $\ell : V \to \Sigma$ is a function that assigns labels from an alphabet Σ to nodes in the graph.[1] The neighbourhood $\mathcal{N}(v)$ of a node v is the set of nodes to which v is connected by an edge, that is $\mathcal{N}(v) = \{v' | (v, v') \in E\}$. For simplicity, we assume that every graph has n nodes, m edges, and a maximum degree of d. The size of G is defined as the cardinality of V.

A walk is a sequence of nodes in a graph, in which consecutive nodes are connected by an edge. A path is a walk that consists of distinct nodes only. A *(rooted) subtree* is a subgraph of a graph, which has no cycles, but a designated root node. A subtree of G can thus be seen as a connected subset of distinct nodes of G with an underlying tree structure. The height of a subtree is the maximum distance between the root and any other node in the subtree. Just as the notion of walk extends the notion of path by allowing nodes to be equal, the notion of subtrees can be extended to *subtree patterns* (also called *tree-walks*, Bach, 2008), which can have nodes that are equal (see Figure 1). These repetitions of the same node are then treated as distinct nodes, such that the pattern is still a cycle-free tree. Note that all subtree kernels compare subtree *patterns* in two graphs, not (strict) subtrees.

Several different graph kernels have been defined in machine learning which can be categorized into three classes: graph kernels based on walks (Kashima et al., 2003; Gärtner et al., 2003) and paths (Borgwardt and Kriegel, 2005), graph kernels based on limited-size subgraphs (Horváth et al., 2004; Shervashidze et al., 2009), and graph kernels based on subtree patterns (Ramon and Gärtner, 2003; Mahé and Vert, 2009).

The first class, graph kernels on walks and paths, compute the number of matching pairs of random walks (resp. paths) in two graphs. The standard formulation of the random walk kernel, based on the direct product graph of two graphs, is computable in $O(n^6)$ for a pair of graphs (Gärtner et al., 2003). However, the same problem can be stated in terms of Kronecker products that can be exploited to bring down the runtime complexity to $O(n^3)$ (Vishwanathan et al., 2010). For a

1. An extension of this definition and of our results to graphs with discrete edge labels is straightforward, but omitted for clarity of presentation.

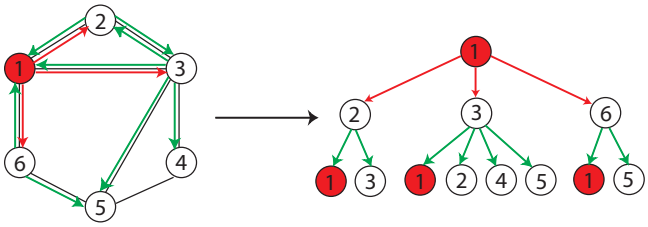


Figure 1: A subtree pattern of height 2 rooted at the node 1. Note the repetitions of nodes in the unfolded subtree pattern on the right.

computer vision application, Harchaoui and Bach (2007) have proposed a dynamic programming-based approach to speed up the computation of the random walk kernel, but at the cost of considering walks of fixed size. Suard et al. (2005) and Vert et al. (2009) present other applications of random walk kernels in computer vision. Mahé et al. (2004) have proposed extensions of marginalized graph kernels (Kashima et al., 2003) for a chemoinformatics application: here the authors relabel vertices of graphs using the Morgan index (Morgan, 1965), which increases the specificity of labels by augmenting them with information on the number of walks starting at a node, and thereby also helps reduce the runtime, as fewer vertices will match. The shortest path kernel by Borgwardt and Kriegel (2005) counts pairs of shortest paths having the same source and sink labels and the same length in two graphs. The runtime of this kernel scales as $O(n^4)$.

The second class, graph kernels based on limited-size subgraphs, includes kernels based on so-called *graphlets*, which represent graphs as counts of all types of subgraphs of size $k \in \{3,4,5\}$. There exist efficient computation schemes for these kernels based on sampling or exploitation of the low maximum degree of graphs (Shervashidze et al., 2009), but these apply to unlabeled graphs only. Cyclic pattern kernels (Horváth et al., 2004) count pairs of matching cyclic patterns in two graphs. Computing this kernel for a general graph is unfortunately NP-hard, however there exist special cases where the kernel can be efficiently computed. The kernel, recently proposed by Costa and De Grave (2010), can also be classified in this category: It counts identical pairs of rooted subgraphs containing nodes up to a certain distance from the root, the roots of which are located at a certain distance from each other, in two graphs.

The first kernel from the third class, subtree kernels, was defined by Ramon and Gärtner (2003). Intuitively, to compare graphs G and G', this kernel iteratively compares all matchings between neighbours of two nodes v from G and v' from G'. In other words, for all pairs of nodes v from G and v' from G', it counts all pairs of matching substructures in subtree patterns rooted at v and v'. The runtime complexity of the subtree kernel for a data set of N graphs is $O(N^2n^2h\,4^d)$. For a detailed description of this kernel see Section 3.2.2.

The subtree kernels by Mahé and Vert (2009) and Bach (2008) refine the Ramon-Gärtner kernel for applications in chemoinformatics and hand-written digit recognition. Both Mahé and Vert (2009) and Bach (2008) propose to consider α-ary subtrees with at most α children per node. This restricts the set of matchings to matchings of up to α nodes, but the runtime complexity is still exponential

in this parameter α, which both papers describe as feasible on small graphs (with approximately 20 nodes on average) with many distinct node labels.

It is a general limitation of all the aforementioned graph kernels that they scale poorly to large, labeled graphs with more than 100 nodes: In the worst case, none of them scale better than $O(n^3)$. The efficient comparison of large, labeled graphs remained an unsolved challenge for almost a decade. We present a general definition of graph kernels that encompasses many previously known graph kernels, and instances of which are efficient to compute for both unlabeled and discretely labeled graphs with thousands of nodes next. Moreover, in terms of prediction accuracy in graph classification tasks its instances are competitive with or outperform other state-of-the-art graph kernels.

The remainder of this article is structured as follows. In Section 2, we describe the Weisfeiler-Lehman isomorphism test that our main contribution is based on. In Section 3, we describe what we call the Weisfeiler-Lehman graphs and our proposed general graph kernels based on them, followed by some examples. In Section 4, we compare these kernels to each other, as well as to a set of five other state-of-the-art graph kernels. We report results on kernel computation runtime and classification accuracy on graph benchmark data sets. Section 5 summarizes our contributions.

2. The Weisfeiler-Lehman Test of Isomorphism

Our graph kernels use concepts from the Weisfeiler-Lehman test of isomorphism (Weisfeiler and Lehman, 1968), more specifically its 1-dimensional variant, also known as "naive vertex refinement". Assume we are given two graphs G and G' and we would like to test whether they are isomorphic. The 1-dimensional Weisfeiler-Lehman test proceeds in iterations, which we index by i and which comprise the steps given in Algorithm 1.

The key idea of the algorithm is to augment the node labels by the sorted set of node labels of neighbouring nodes, and compress these augmented labels into new, short labels. These steps are then repeated until the node label sets of G and G' differ, or the number of iterations reaches n. See Figure 2, a-d, for an illustration of these steps (note however, that the two graphs in the figure would directly be identified as non-isomorphic by the Weisfeiler-Lehman test, as their label sets are already different in the beginning).

Sorting the set of multisets allows for a straightforward definition and implementation of f for the compression of labels in step 4: one keeps a counter variable for f that records the number of distinct strings that f has compressed before. f assigns the current value of this counter to a string if an identical string has been compressed before, but when one encounters a new string, one increments the counter by one and f assigns its value to the new string. The sorted order of the set of multisets guarantees that all identical strings are mapped to the same number, because they occur in a consecutive block. However, note that the sorting of the set of multisets is not required for defining f. Any other injective mapping will give equivalent results. The alphabet Σ has to be sufficiently large for f to be injective. For two graphs, $|\Sigma| = 2n$ suffices.

The Weisfeiler-Lehman algorithm terminates after step 4 of iteration i if $\{l_i(v)|v \in V\} \neq \{l_i(v')|$ $v' \in V'\}$, that is, if the sets of newly created labels are not identical in G and G'. The graphs are then not isomorphic. If the sets are identical after n iterations, it means that either G and G' are isomorphic, or the algorithm has not been able to determine that they are not isomorphic (see Cai et al., 1992, for examples of graphs that cannot be distinguished by this algorithm or its higher-dimensional variants). As a side note, we mention that the 1-dimensional Weisfeiler-Lehman al-

Algorithm 1 One iteration of the 1-dim. Weisfeiler-Lehman test of graph isomorphism

1: Multiset-label determination
- For $i = 0$, set $M_i(v) := l_0(v) = \ell(v)$. [2]
- For $i > 0$, assign a multiset-label $M_i(v)$ to each node v in G and G' which consists of the multiset $\{l_{i-1}(u) | u \in \mathcal{N}(v)\}$.

2: Sorting each multiset
- Sort elements in $M_i(v)$ in ascending order and concatenate them into a string $s_i(v)$.
- Add $l_{i-1}(v)$ as a prefix to $s_i(v)$ and call the resulting string $s_i(v)$.

3: Label compression
- Sort all of the strings $s_i(v)$ for all v from G and G' in ascending order.
- Map each string $s_i(v)$ to a new compressed label, using a function $f : \Sigma^* \to \Sigma$ such that $f(s_i(v)) = f(s_i(w))$ if and only if $s_i(v) = s_i(w)$.

4: Relabeling
- Set $l_i(v) := f(s_i(v))$ for all nodes in G and G'.

gorithm has been shown to be a valid isomorphism test for almost all graphs (Babai and Kucera, 1979).

Note that in Algorithm 1 we used the same node labeling functions $\ell, l_0, \ldots, l_h$ for both G and G' in order not to overload the notation. We will continue using this notation throughout the paper and assume without loss of generality that the domain of these functions $\ell, l_0, \ldots, l_h$ is the set of all nodes in our data set of graphs, which corresponds to $V \cup V'$ in the case of Algorithm 1.

2.1 Complexity

The runtime complexity of the 1-dimensional Weisfeiler-Lehman algorithm with h iterations is $O(hm)$. Defining the multisets in step 1 for all nodes is an $O(m)$ operation. Sorting each multiset is an $O(m)$ operation for all nodes. This efficiency can be achieved by using counting sort, which is an instance of bucket sort, due to the limited range of the elements of the multiset. The elements of each multiset are a subset of $\{f(s_i(v)) | v \in V\}$. For a fixed i, the cardinality of this set is upper-bounded by n, which means that we can sort all multisets in $O(m)$ by the following procedure: We assign the elements of all multisets to their corresponding buckets, recording which multiset they came from. By reading through all buckets in ascending order, we can then extract the sorted multisets for all nodes in a graph. The runtime is $O(m)$ as there are $O(m)$ elements in the multisets of a graph in iteration i. Sorting the resulting strings is of time complexity $O(m)$ via radix sort (see Mehlhorn, 1984, Vol. 1, Section II.2.1). The label compression requires one pass over all strings and their characters, that is $O(m)$. Hence all these steps result in a total runtime of $O(hm)$ for h iterations.

2.2 Link with Subtree Patterns

Note that the compressed labels $l_i(v)$ correspond to subtree patterns of height i rooted at v (see Figure 1 for an illustration of subtree patterns).

2. For unlabeled graphs, node labels $M_0(v) := l_0(v)$ can be initialized with letters corresponding one to one to node degrees $|\mathcal{N}(v)|$.

3. The General Weisfeiler-Lehman Kernels

In this section, we first define the Weisfeiler-Lehman graph sequence and the general graph kernels based on them. We then present three instances of this kernel, the Weisfeiler-Lehman subtree kernel (Section 3.2), the Weisfeiler-Lehman edge kernel (Section 3.3), and the Weisfeiler-Lehman shortest path kernel (Section 3.4).

3.1 The Weisfeiler-Lehman Kernel Framework

In each iteration i of the Weisfeiler-Lehman algorithm (see Algorithm 1), we get a new labeling $l_i(v)$ for all nodes v. Recall that this labeling is concordant in G and G', meaning that if nodes in G and G' have identical multiset labels, and only in this case, they will get identical new labels. Therefore, we can imagine one iteration of Weisfeiler-Lehman relabeling as a function $r((V,E,l_i)) = (V,E,l_{i+1})$ that transforms all graphs in the same manner. Note that r depends on the set of graphs that we consider.

Definition 1 *Define the* Weisfeiler-Lehman graph *at height i of the graph* $G = (V,E,\ell) = (V,E,l_0)$ *as the graph* $G_i = (V,E,l_i)$. *We call the sequence of Weisfeiler-Lehman graphs*

$$\{G_0, G_1, \ldots, G_h\} = \{(V,E,l_0),(V,E,l_1),\ldots,(V,E,l_h)\},$$

where $G_0 = G$ *and* $l_0 = \ell$, *the* Weisfeiler-Lehman sequence *up to height h of G.*

G_0 is the original graph, $G_1 = r(G_0)$ is the graph resulting from the first relabeling, and so on. Note that neither V, nor E ever change in this sequence, but we define it as a sequence of graphs rather than a sequence of labeling functions for the sake of clarity of definitions that follow.

Definition 2 *Let k be any kernel for graphs, that we will call the* base kernel. *Then the Weisfeiler-Lehman kernel with h iterations with the base kernel k is defined as*

$$k_{WL}^{(h)}(G,G') = k(G_0,G'_0) + k(G_1,G'_1) + \ldots + k(G_h,G'_h), \tag{1}$$

where h is the number of Weisfeiler-Lehman iterations and $\{G_0,\ldots,G_h\}$ *and* $\{G'_0,\ldots,G'_h\}$ *are the Weisfeiler-Lehman sequences of G and* G' *respectively.*

Theorem 3 *Let the base kernel k be any positive semidefinite kernel on graphs. Then the corresponding Weisfeiler-Lehman kernel* $k_{WL}^{(h)}$ *is positive semidefinite.*

Proof Let ϕ be the feature mapping corresponding to the kernel k:

$$k(G_i,G'_i) = \langle \phi(G_i), \phi(G'_i) \rangle.$$

We have

$$k(G_i,G'_i) = k(r^i(G),r^i(G')) = \langle \phi(r^i(G)), \phi(r^i(G')) \rangle.$$

Let us define the feature mapping $\psi(G)$ as $\phi(r^i(G))$. Then we have

$$k(G_i,G'_i) = \langle \psi(G), \psi(G') \rangle,$$

hence k is a kernel on G and G' and $k_{WL}^{(h)}$ is positive semidefinite as a sum of positive semidefinite kernels.

■

This definition provides a framework for applying all graph kernels that take into account discrete node labels to different levels of the node-labeling of graphs, from the original labeling to more and more fine-grained labelings for growing h. This enriches the set of extracted features. For example, while the shortest path kernel counts pairs of shortest paths with the same distance between identically labeled source and sink nodes on the original graphs, it will count pairs of shortest paths with the same distance between the roots of identical subtree patterns of height 1 on Weisfeiler-Lehman graphs with $h = 1$.

For some base kernels one might be able to exploit the fact that the graph structure does not change over the Weisfeiler-Lehman sequence to do some computations only once instead of repeating it h times. One example of such a base kernel is the shortest path kernel: As shortest paths in a graph G are the same as shortest paths in corresponding Weisfeiler-Lehman graphs G_i, we can precompute them. One should bear in mind that for graph kernels k that depend on the size of the alphabet of node labels, computing $k(G_i, G_i')$ will accordingly get increasingly expensive, or, in some cases, cheaper, as a function of growing i.

Note that it is possible to put nonnegative real weights α_i on $k(G_i, G_i')$, $i = \{0, 1, \ldots, h\}$, to obtain a more general definition of the Weisfeiler-Lehman kernel:

$$k_{WL}^{(h)}(G, G') = \alpha_0 k(G_0, G_0') + \alpha_1 k(G_1, G_1') + \ldots + \alpha_h k(G_h, G_h').$$

In this case, $k_{WL}^{(h)}$ will still be positive semidefinite, as a positive linear combination of positive semidefinite kernels.

3.1.1 NOTE ON COMPUTING WEISFEILER-LEHMAN KERNELS IN PRACTICE

In the inductive learning setting, we compute the kernel on the training set of graphs. For any test graph that we subsequently need to classify, we have to map it to the feature space spanned by original and compressed labels occurred in the training set. For this purpose, we will need to maintain record of the data structures that hold the mappings $l_i(v) := f(s_i(v))$ for each iteration i and each distinct $s_i(v)$. This requires $O(Nmh)$ memory in the worst case.

In contrast, in the transductive setting, where the test set is already known, we can compute the kernel matrix on the whole data set (training and test set) without having to keep the mappings mentioned above.

3.2 The Weisfeiler-Lehman Subtree Kernel

In this section we present the Weisfeiler-Lehman subtree kernel (Shervashidze and Borgwardt, 2009), which is a natural instance of Definition 2.

Definition 4 *Let G and G' be graphs. Define $\Sigma_i \subseteq \Sigma$ as the set of letters that occur as node labels at least once in G or G' at the end of the i-th iteration of the Weisfeiler-Lehman algorithm. Let Σ_0 be the set of original node labels of G and G'. Assume all Σ_i are pairwise disjoint. Without loss of generality, assume that every $\Sigma_i = \{\sigma_{i1}, \ldots, \sigma_{i|\Sigma_i|}\}$ is ordered. Define a map $c_i : \{G, G'\} \times \Sigma_i \to \mathbb{N}$ such that $c_i(G, \sigma_{ij})$ is the number of occurrences of the letter σ_{ij} in the graph G.*

The Weisfeiler-Lehman subtree kernel on two graphs G and G' with h iterations is defined as:

$$k^{(h)}_{WLsubtree}(G,G') = \langle \phi^{(h)}_{WLsubtree}(G), \phi^{(h)}_{WLsubtree}(G') \rangle, \tag{2}$$

where

$$\phi^{(h)}_{WLsubtree}(G) = (c_0(G,\sigma_{01}),\ldots,c_0(G,\sigma_{0|\Sigma_0|}),\ldots,c_h(G,\sigma_{h1}),\ldots,c_h(G,\sigma_{h|\Sigma_h|})),$$

and

$$\phi^{(h)}_{WLsubtree}(G') = (c_0(G',\sigma_{01}),\ldots,c_0(G',\sigma_{0|\Sigma_0|}),\ldots,c_h(G',\sigma_{h1}),\ldots,c_h(G',\sigma_{h|\Sigma_h|})).$$

That is, the Weisfeiler-Lehman subtree kernel counts common *original and compressed labels* in two graphs. See Figure 2 for an illustration.

Theorem 5 *The Weisfeiler-Lehman subtree kernel on a pair of graphs G and G' can be computed in time $O(hm)$.*

Proof This follows directly from the definition of the Weisfeiler-Lehman subtree kernel and the runtime complexity of the Weisfeiler-Lehman test, as described in Section 2. ■

The following theorem shows that (2) is indeed a special case of the general Weisfeiler-Lehman kernel (1).

Theorem 6 *Let the base kernel k be a function counting pairs of matching node labels in two graphs:*

$$k(G,G') = \sum_{v\in V}\sum_{v'\in V'} \delta(\ell(v),\ell(v')),$$

where δ is the Dirac kernel, that is, it is 1 when its arguments are equal and 0 otherwise. Then $k^{(h)}_{WL}(G,G') = k^{(h)}_{WLsubtree}(G,G')$ for all G,G'.

Proof It is easy to notice that for each $i \in \{0,1,\ldots,h\}$ we have

$$\sum_{v\in V}\sum_{v'\in V'} \delta(l_i(v),l'_i(v')) = \sum_{j=1}^{|\Sigma_i|} c_i(G,\sigma_{ij})c_i(G',\sigma_{ij}).$$

Adding up these sums for all $i \in \{0,1,\ldots,h\}$ gives us $k^{(h)}_{WL}(G,G') = k^{(h)}_{WLsubtree}(G,G')$. ■

3.2.1 COMPUTING THE WEISFEILER-LEHMAN SUBTREE KERNEL ON MANY GRAPHS

To compute the Weisfeiler-Lehman subtree kernel on N graphs, we propose Algorithm 2, which improves over the naive, N^2-fold application of the kernel from Definition 4. We now process all N graphs simultaneously and conduct the steps given in Algorithm 2 on each graph G in each of h iterations.

As before, Σ is assumed to be sufficiently large to allow f to be injective. In the case of N graphs and h iterations, a Σ of size $Nn(h+1)$ suffices.

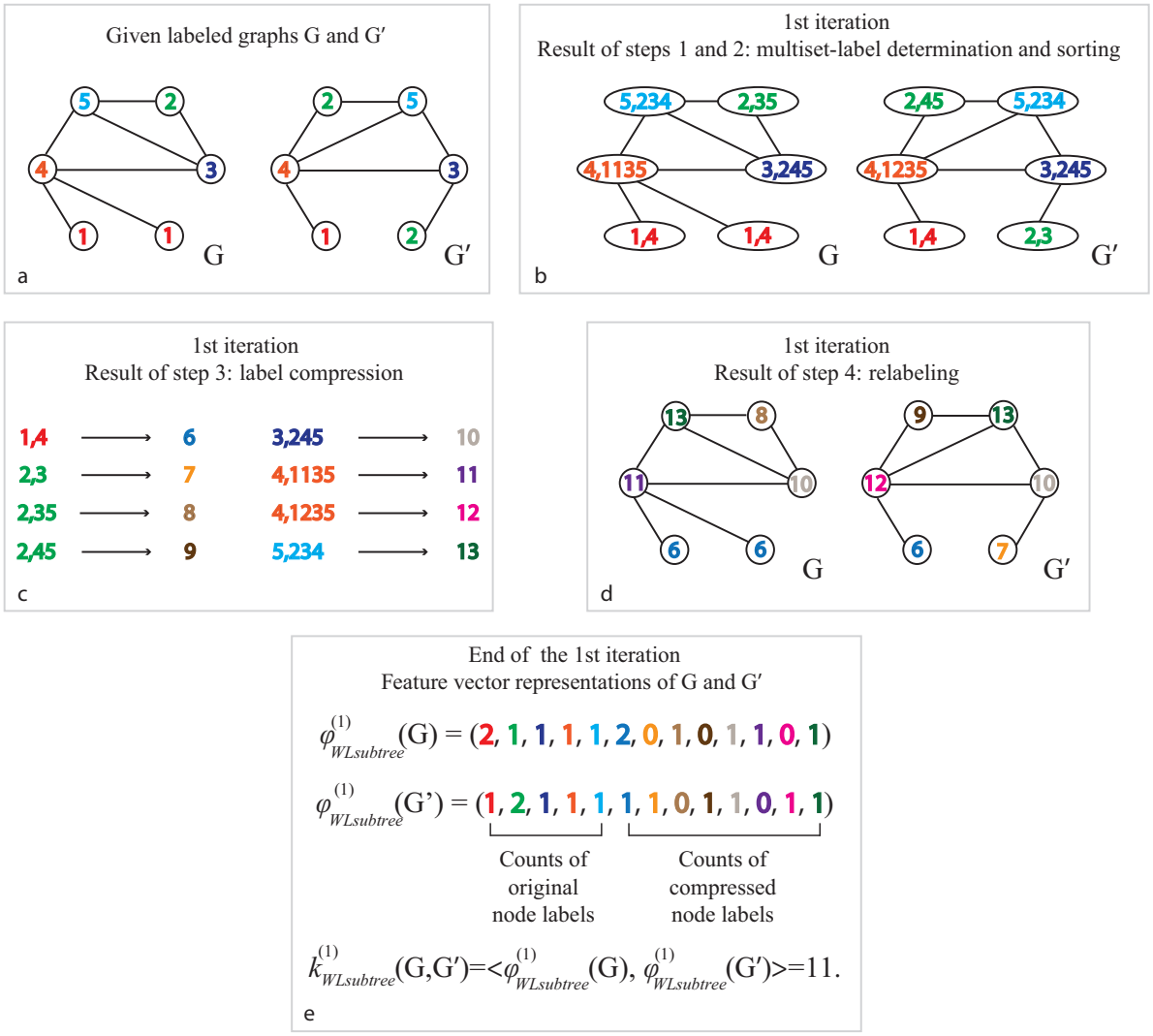


Figure 2: Illustration of the computation of the Weisfeiler-Lehman subtree kernel with $h = 1$ for two graphs. Here $\{1,2,\ldots,13\} \in \Sigma$ are considered as letters. Note that compressed labels denote subtree patterns: For instance, if a node has label 8, this means that there is a subtree pattern of height 1 rooted at this node, where the root has label 2 and its neighbours have labels 3 and 5.

One way of implementing f is to sort all neighbourhood strings using radix sort, as done in step 4 in Algorithm 1. The resulting complexity of this step would be linear in the sum of the size of the current alphabet and the total length of strings, that is $O(Nn+Nm) = O(Nm)$. An alternative implementation of f would be by means of a perfect hash function.

Algorithm 2 One iteration of the Weisfeiler-Lehman subtree kernel computation on N graphs

1: Multiset-label determination
- Assign a multiset-label $M_i(v)$ to each node v in G which consists of the multiset $\{l_{i-1}(u)|u \in \mathcal{N}(v)\}$.

2: Sorting each multiset
- Sort elements in $M_i(v)$ in ascending order and concatenate them into a string $s_i(v)$.
- Add $l_{i-1}(v)$ as a prefix to $s_i(v)$.

3: Label compression
- Map each string $s_i(v)$ to a compressed label using a hash function $f : \Sigma^* \to \Sigma$ such that $f(s_i(v)) = f(s_i(w))$ if and only if $s_i(v) = s_i(w)$.

4: Relabeling
- Set $l_i(v) := f(s_i(v))$ for all nodes in G.

Theorem 7 *For N graphs, the Weisfeiler-Lehman subtree kernel with h iterations on all pairs of these graphs can be computed in $O(Nhm + N^2hn)$.*

Proof Naive application of the kernel from Definition 4 for computing an $N \times N$ kernel matrix would require a runtime of $O(N^2hm)$. One can improve upon this runtime complexity by computing $\phi^{(h)}_{WLsubtree}$ explicitly for each graph and only then taking pairwise inner products.

Step 1, the multiset-label determination, still requires $O(Nm)$. Step 2, the sorting of the elements in each multiset, can be done via a joint bucket sort (counting sort) of all strings, requiring $O(Nn + Nm)$ time.

The effort of computing $\phi^{(h)}_{WLsubtree}$ on all N graphs in h iterations is then $O(Nhm)$, assuming that $m > n$. To get all pairwise kernel values, we have to multiply all feature vectors, which requires a runtime of $O(N^2hn)$, as each graph G has at most hn non-zero entries in $\phi^{(h)}_{WLsubtree}(G)$. In Section 4.1, we empirically show that the first term Nhm dominates the overall runtime in practice. ■

While our Weisfeiler-Lehman subtree kernel matches neighbourhoods of nodes in a graph exactly, one could also think of other strategies of comparing node neighbourhoods, and still retain the favourable runtime of our graph kernel. In research that was published in parallel to ours, Hido and Kashima (2009) present such an alternative kernel based on node neighbourhoods which uses hash functions and logical operations on bit-representations of node labels and which also scales linearly in the number of edges. The Morgan index (Morgan, 1965) is another way of summarizing information contained in the neighbourhood of a node, and has been used by Mahé et al. (2004) in the context of graph kernels.

3.2.2 THE RAMON-GÄRTNER SUBTREE KERNEL

Description. The first subtree kernel on graphs was defined by Ramon and Gärtner (2003). The Ramon-Gärtner subtree kernel with subtree height h compares all pairs of nodes from graphs $G = (V, E, \ell)$ and $G' = (V', E', \ell)$ by iteratively comparing their neighbourhoods:

$$k^{(h)}_{RG}(G, G') = \sum_{v \in V} \sum_{v' \in V'} k_{RG,h}(v, v'),$$

where

$$k_{RG,h}(v,v') = \begin{cases} \delta(\ell(v),\ell(v')), & \text{if } h=0 \\ \lambda_v \lambda_{v'} \delta(\ell(v),\ell(v')) \sum_{R\in\mathcal{M}(v,v')} \prod_{(w,w')\in R} k_{RG,h-1}(w,w'), & \text{if } h>0, \end{cases}$$

δ is an indicator function that equals 1 if its arguments are equal, 0 otherwise, λ_v and $\lambda_{v'}$ are weights associated with nodes v and v', and

$$\mathcal{M}(v,v') = \Big\{ R \subseteq \mathcal{N}(v) \times \mathcal{N}(v') \Big| (\forall (u,u'),(w,w') \in R : u = w \Leftrightarrow u' = w') \\ \wedge (\forall (u,u') \in R : \ell(u) = \ell(u')) \Big\}. \tag{3}$$

Said differently, $\mathcal{M}(v,v')$ is the set of exact matchings of subsets of the neighbourhoods of v and v'. Each element R of $\mathcal{M}(v,v')$ is a set of pairs of nodes from the neighbourhoods of $v \in V$ and $v' \in V'$ such that nodes in each pair have identical labels and no node is contained in more than one pair. Thus, intuitively, k_{RG} iteratively considers all matchings $\mathcal{M}(v,v')$ between neighbours of two identically labeled nodes v from G and v' from G'. Taking the parameters λ_v and $\lambda_{v'}$ equal to a single parameter λ results in weighting each pattern by λ raised to the power of the number of nodes in the pattern.

Complexity. The runtime complexity of the subtree kernel for a pair of graphs is $O(n^2 h 4^d)$, including a comparison of all pairs of nodes (n^2), and a pairwise comparison of all matchings in their neighbourhoods in $O(4^d)$, which is repeated in h iterations. h is a multiplicative factor, not an exponent, since one can implement the subtree kernel via dynamic programming, starting with k_1 and computing k_h from k_{h-1}. For a data set of N graphs, the resulting runtime complexity is then in $O(N^2 n^2 h 4^d)$.

3.2.3 LINK TO THE WEISFEILER-LEHMAN SUBTREE KERNEL

The Weisfeiler-Lehman subtree kernel can be defined in a recursive fashion which elucidates its relation to the Ramon-Gärtner kernel.

Theorem 8 *The kernel $k_{rec}^{(h)}$ defined as*

$$k_{rec}^{(h)}(G,G') = \sum_{i=0}^{h} \sum_{v\in V} \sum_{v'\in V'} k_{rec,i}(v,v'), \tag{4}$$

where

$$k_{rec,i}(v,v') = \begin{cases} \delta(\ell(v),\ell(v')), & \text{if } i=0 \\ k_{rec,i-1}(v,v') \max_{R\in\mathcal{M}(v,v')} \prod_{(w,w')\in R} k_{rec,i-1}(w,w'), & \text{if } i>0 \text{ and } \mathcal{M}\neq\emptyset \\ 0, & \text{if } i>0 \text{ and } \mathcal{M}=\emptyset, \end{cases} \tag{5}$$

δ *is the indicator function again, and*

$$\mathcal{M}(v,v') = \Big\{ R \subseteq \mathcal{N}(v) \times \mathcal{N}(v') \Big| |R| = |\mathcal{N}(v)| = |\mathcal{N}(v')| \\ \wedge (\forall (u,u'),(w,w') \in R : u = w \Leftrightarrow u' = w') \wedge (\forall (u,u') \in R : \ell(u) = \ell(u')) \Big\}, \tag{6}$$

is equivalent to the Weisfeiler-Lehman subtree kernel $k_{WLsubtree}^{(h)}$.

In other words, $\mathcal{M}(v,v')$ is the set of exact matchings of the neighbourhoods of v and v'. It is nonempty only in the case where the neighbourhoods of v and v' have exactly the same size and the multisets of labels of their neighbours $\{\ell(u)|u \in \mathcal{N}(v)\}$ and $\{\ell(u')|u' \in \mathcal{N}(v')\}$ are identical. Note that $k_{rec,i}(v,v')$ only takes binary values: it evaluates to 1 if the subtree patterns of height i rooted at v and v' are identical, and to 0 otherwise.

Proof We prove this theorem by induction over h.

Induction initialisation $h = 0$:

$$k^{(0)}_{WLsubtree} = \langle \phi^{(0)}_{WLsubtree}(G), \phi^{(0)}_{WLsubtree}(G) \rangle = \sum_{j=1}^{|\Sigma_0|} c_0(G,\sigma_{0j}) c_0(G',\sigma_{0j}) =$$
$$= \sum_{v \in V} \sum_{v' \in V'} \delta(\ell(v),\ell(v')) = k^{(0)}_{rec},$$

where Σ_0 is the initial alphabet of node labels and $c_0(G,\sigma_{0j})$ is the number of occurrences of the letter σ_{0j} as a node label in G. The equality follows from the definitions of $k^{(h)}_{rec}$ and $k^{(h)}_{WLsubtree}$.

Induction step $h \to h+1$: Assume that $k^{(h)}_{WLsubtree} = k^{(h)}_{rec}$. Then

$$k^{(h+1)}_{rec} = \sum_{v \in V} \sum_{v' \in V'} k_{rec,h+1}(v,v') + \sum_{i=0}^{h} \sum_{v \in V} \sum_{v' \in V'} k_{rec,i}(v,v') = \tag{7}$$
$$= \sum_{j=1}^{|\Sigma_{h+1}|} c_{h+1}(G,\sigma_{h+1,j}) c_{h+1}(G',\sigma_{h+1,j}) + k^{(h)}_{WLsubtree} = k^{(h+1)}_{WLsubtree}, \tag{8}$$

where the equality of (7) and (8) follows from the fact that $k_{rec,h+1}(v,v') = 1$ if and only if the labels and neigbourhoods of v and v' are identical, that is, if $f(s_{h+1}(v)) = f(s_{h+1}(v'))$. ■

Theorem 8 highlights the following differences between the Weisfeiler-Lehman and the Ramon-Gärtner subtree kernels: In Equation (4), Weisfeiler-Lehman considers all subtrees up to height h, whereas the Ramon-Gärtner kernel looks at subtrees of exactly height h. In Equations (5) and (6), the Weisfeiler-Lehman subtree kernel checks whether the neighbourhoods of v and v' match exactly, while the Ramon-Gärtner kernel considers all pairs of matching subsets of the neighbourhoods of v and v' in Equation (3). In our experiments, we examine the empirical differences between these two kernels in terms of runtime and prediction accuracy on classification benchmark data sets (see Section 4.2).

3.3 The Weisfeiler-Lehman Edge Kernel

The Weisfeiler-Lehman edge kernel is another instance of the Weisfeiler-Lehman kernel framework. In the case of graphs with unweighted edges, we consider the base kernel that counts matching pairs of edges with identically labeled endpoints (incident nodes) in two graphs. In other words, the base kernel is defined as

$$k_E = \langle \phi_E(G), \phi_E(G') \rangle,$$

where $\phi_E(G)$ is a vector of numbers of occurrences of pairs (a,b), $a,b \in \Sigma$, which represent ordered labels of endpoints of an edge in G. Denoting (a,b) and (a',b') the ordered labels of endpoints of edges e and e' respectively, and δ the Dirac kernel, k_E can equivalently be expressed as

$\sum_{e\in E}\sum_{e'\in E'}\delta(a,a')\delta(b,b')$. If the edges are weighted by a function w that assigns weights, the base kernel k_E can be defined as $\sum_{e\in E}\sum_{e'\in E'}\delta(a,a')\delta(b,b')k_w(w(e),w(e'))$, where k_w is a kernel comparing edge weights.

Following (1), we have

$$k^{(h)}_{WL\,edge} = k_E(G_0,G'_0)+k_E(G_1,G'_1)+\ldots+k_E(G_h,G'_h).$$

3.3.1 NOTE ON COMPUTATIONAL COMPLEXITY

If the edges are not weighted or labeled, the number of possible edge features in each iteration equals the number of distinct ordered pairs (a,b), that is, $\frac{|\Sigma_i|(|\Sigma_i|+1)}{2}$. It is easy to notice by looking at the Algorithm 1 that for each $i\in\{0,\ldots,h-1\}$, we have $|\Sigma_i|\leq|\Sigma_{i+1}|$. Therefore, if we compute the edge kernel by first explicitly computing $\phi_E(G)$ for each G in the data set, the computation will become increasingly expensive in each iteration i of the Weisfeiler-Lehman relabeling.

If edges are weighted and we use any general kernel to compare their weights, computing the feature map explicitly may not be possible or practical any more. In this case, the kernel can be computed by comparing edges pairwise in each pair of graphs. Assuming that the kernel on a pair of weights can be computed in $O(1)$, this results in $O(N^2m^2)$ operations per Weisfeiler-Lehman iteration.

Computing the feature map explicitly can also become problematic if the alphabet size gets prohibitively large. In this case, one can either compute the kernel via pairwise comparisons of edges in each pair of graphs as above ($O(N^2m^2)$ per iteration), or via the construction of the explicit feature map for each pair of graphs separately, potentially yielding smaller alphabets Σ_i than considering the whole data set of N graphs at once.

3.4 The Weisfeiler-Lehman Shortest Path Kernel

Another example of the general Weisfeiler-Lehman kernels that we consider is the Weisfeiler-Lehman shortest path kernel. Here we use a node-labeled shortest path kernel (Borgwardt and Kriegel, 2005) as the base kernel.

In the particular case of graphs with unweighted edges, we consider the base kernel k_{SP} of the form $k_{SP}(G,G') = \langle\phi_{SP}(G),\phi_{SP}(G')\rangle$, where $\phi_{SP}(G)$ (resp. $\phi_{SP}(G')$) is a vector whose components are numbers of occurrences of triplets of the form (a,b,p) in G (resp. G'), where $a,b\in\Sigma$ are ordered endpoint labels of a shortest path and $p\in\mathbb{N}_0$ is the shortest path length.

According to (1), we have

$$k^{(h)}_{WL\,shortest\ path} = k_{SP}(G_0,G'_0)+k_{SP}(G_1,G'_1)+\ldots+k_{SP}(G_h,G'_h).$$

3.4.1 NOTE ON COMPUTATIONAL COMPLEXITY

Computing shortest paths between all pairs of nodes in a graph can be done in $O(n^3)$ using the Floyd-Warshall algorithm. Consequently, for N graphs, the complexity is of $O(Nn^3)$. This step does not have to be repeated for every Weisfeiler-Lehman iteration, as the topology of a graph does not change across the Weisfeiler-Lehman sequence. In case edges are not weighted, shortest paths are determined in terms of geodesic distance and path lengths are integers. Denote the number of distinct shortest path lengths occurring in the data set of graphs as P.

Let us first consider the Dirac (δ) kernel on the shortest path lengths, which means that the similarity of two paths in two graphs equals 1 if they have exactly the same length and identically labeled endpoints and 0 otherwise. Then, in iteration i of the Weisfeiler-Lehman relabeling, we can bound the number of features, triplets (a,b,p) where $a,b \in |\Sigma_i|$ are ordered start and end node labels and $p \in \mathbb{N}_0$ the shortest path length, by $\frac{|\Sigma_i|(|\Sigma_i|+1)}{2}P$. As $|\Sigma_i| \leq |\Sigma_{i+1}|$ for each $i \in \{0,\ldots,h-1\}$, if we compute the shortest path kernel by first explicitly computing $\phi_{SP}(G)$ for each G in the data set, the computation will get increasingly expensive in each iteration, as in the case of edge kernels (Section 3.3).

Similarly to the Weisfeiler-Lehman edge kernel, in a more general setting where we do not assume that edges are unweighted and use any kernel (not necessarily the Dirac kernel) on shortest path lengths, or if the alphabet size gets prohibitively large, computing the feature map explicitly may become impossible or difficult. In this case, we can compute the kernel by comparing shortest path lengths pairwise in two graphs. Therefore, the runtime of computing $k_{SP}(G_i, G_i')$ will not depend on i any more. It will scale as $O(n^4)$ for each pair of graphs as we have to compare all pairs of the $O(n^2)$ shortest path lengths, and $O(N^2n^4)$ for the whole data set.

3.5 Other Weisfeiler-Lehman Kernels

In a similar fashion, we can plug other base graph kernels into our Weisfeiler-Lehman graph kernel framework. As node labels are the only aspect that differentiate Weisfeiler-Lehman graphs at different *resolutions* (determined by the number of iterations), a clear requirement that the base kernel has to satisfy for the Weisfeiler-Lehman kernel to make sense is to exploit the labels on nodes. A non-exhaustive list of possible base kernels not mentioned in previous sections includes the labeled version of the graphlet kernel (Shervashidze et al., 2009), the random walk kernel (Gärtner et al., 2003; Vishwanathan et al., 2010), and the subtree kernel by Ramon and Gärtner (2003).

4. Experiments

In this section, we first empirically study the runtime behaviour of the Weisfeiler-Lehman subtree kernel on synthetic graphs (Section 4.1). Next, we compare the Weisfeiler-Lehman subtree kernel, the Weisfeiler-Lehman edge kernel, and the Weisfeiler-Lehman shortest path kernel to state-of-the-art graph kernels in terms of kernel computation runtime and classification accuracy on graph benchmark data sets (Section 4.2).

4.1 Runtime Behaviour of Weisfeiler-Lehman Subtree Kernel

Here we experimentally examine the runtime performance of the Weisfeiler-Lehman subtree kernel.

4.1.1 METHODS

We empirically compared the runtime behaviour of our two variants of the Weisfeiler-Lehman subtree (WL) kernel. The first variant computes kernel values pairwise in $O(N^2hm)$. The second variant computes the kernel values in $O(Nhm+N^2hn)$ on the data set simultaneously. We will refer to the former variant as the "pairwise" WL, and the latter as "global" WL.

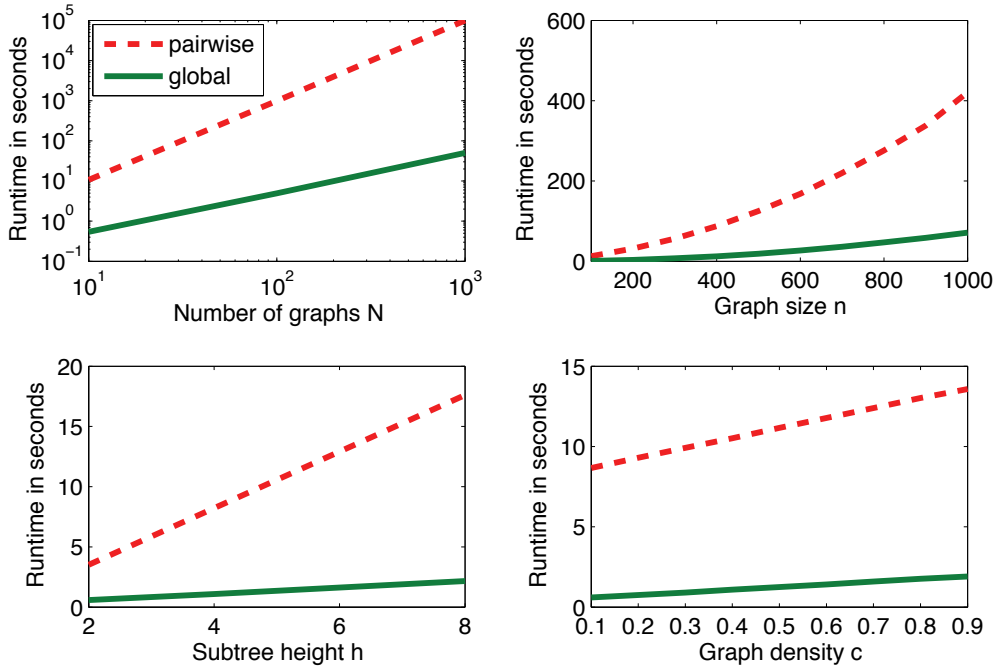


Figure 3: Runtime in seconds for kernel matrix computation on synthetic graphs using the pairwise (red, dashed) and the global (green, solid) computation schemes for the Weisfeiler-Lehman subtree kernel (Default values: data set size $N = 10$, graph size $n = 100$, subtree height $h = 4$, graph density $c = 0.4$).

4.1.2 EXPERIMENTAL SETUP

We assessed the behaviour on randomly generated graphs with respect to four parameters: data set size N, graph size n, subtree height h and graph density c. The density of an undirected graph of n nodes without self-loops is defined as the number of its edges divided by $n(n-1)/2$, the maximal number of edges. We kept 3 out of 4 parameters fixed at their default values and varied the fourth parameter. The default values we used were 10 for N, 100 for n, 4 for h and 0.4 for the graph density c. In more detail, we varied N in range $\{10, 100, 1000\}$, n in $\{100, 200, \ldots, 1000\}$, h in $\{2, 4, 8\}$ and c in $\{0.1, 0.2, \ldots, 0.9\}$.

For each individual experiment, we generated N graphs with n nodes, and inserted edges randomly until the number of edges reached $\lfloor cn(n-1)/2 \rfloor$. We then computed the pairwise and the global WL kernel on these synthetic graphs. We report CPU runtimes in seconds in Figure 3, as measured in Matlab R2008a on an Apple MacPro with 3.0GHz Intel 8-Core with 16GB RAM.

4.1.3 RESULTS

Empirically, we observe that the pairwise kernel scales quadratically with data set size N. Interestingly, the global kernel scales linearly with N for the considered range of N. The N^2 sparse vector

multiplications that have to be performed for kernel computation with global WL do not dominate runtime here. This result on synthetic data indicates that the global WL kernel has attractive scalability properties for large data sets.

When varying the number of nodes n per graph, we observe that the runtime of both WL kernels scales quadratically with n, and the global WL is much faster than the pairwise WL for large graphs. This agrees with the fact that our kernels scale linearly with the number of edges per graph, m, which is $0.4\frac{n(n-1)}{2}$ in this experiment.

We observe a different picture for the height h of the subtree patterns. The runtime of both kernels grows linearly with h, but the global WL is more efficient in terms of runtime.

Varying the graph density c, both methods show again a linearly increasing runtime, although the runtime of the global WL kernel is much lower than the runtime of the pairwise WL.

Across all different graph properties, the global WL kernel from Section 3.2.1 requires less runtime than the pairwise WL kernel from Section 3.2. Hence the global WL kernel is the variant of our Weisfeiler-Lehman subtree kernel that we use on the following graph classification tasks.

4.2 Graph Classification

We compared the performance of the WL subtree kernel, the WL edge kernel and the WL shortest path kernel to several other state-of-the-art graph kernels in terms of runtime and classification accuracy on graph benchmark data sets.

4.2.1 DATA SETS

We employed the following data sets in our experiments: MUTAG, NCI1, NCI109, ENZYMES and D&D. MUTAG (Debnath et al., 1991) is a data set of 188 mutagenic aromatic and heteroaromatic nitro compounds labeled according to whether or not they have a mutagenic effect on the Gram-negative bacterium *Salmonella typhimurium*. NCI1 and NCI109 represent two balanced subsets of data sets of chemical compounds screened for activity against non-small cell lung cancer and ovarian cancer cell lines, respectively (Wale and Karypis, 2006, and `http://pubchem.ncbi.nlm.nih.gov`). ENZYMES is a data set of protein tertiary structures obtained from Borgwardt et al. (2005) consisting of 600 enzymes from the BRENDA enzyme database (Schomburg et al., 2004). In this case the task is to correctly assign each enzyme to one of the 6 EC top-level classes. D&D is a data set of 1178 protein structures (Dobson and Doig, 2003). Each protein is represented by a graph, in which the nodes are amino acids and two nodes are connected by an edge if they are less than 6 Ångstroms apart. The prediction task is to classify the protein structures into enzymes and non-enzymes. Note that nodes are labeled in all data sets.

Figure 4 shows the distributions of node numbers, edge numbers, and degrees in these data sets.

All of these data sets, as well as Matlab scripts for computing kernels used in our experiments, can be downloaded from `http://mlcb.is.tuebingen.mpg.de/Mitarbeiter/Nino/WL/`.

4.2.2 EXPERIMENTAL SETUP

On these data sets, we compared our Weisfeiler-Lehman subtree, Weisfeiler-Lehman edge, and Weisfeiler-Lehman shortest path kernels to the Ramon-Gärtner kernel ($\lambda = 1$), as well as to several state-of-the-art graph kernels for large graphs. Due to the large number of graph kernels in the literature, we could not compare to every single graph kernel, but to representative instances of the major families of graph kernels.

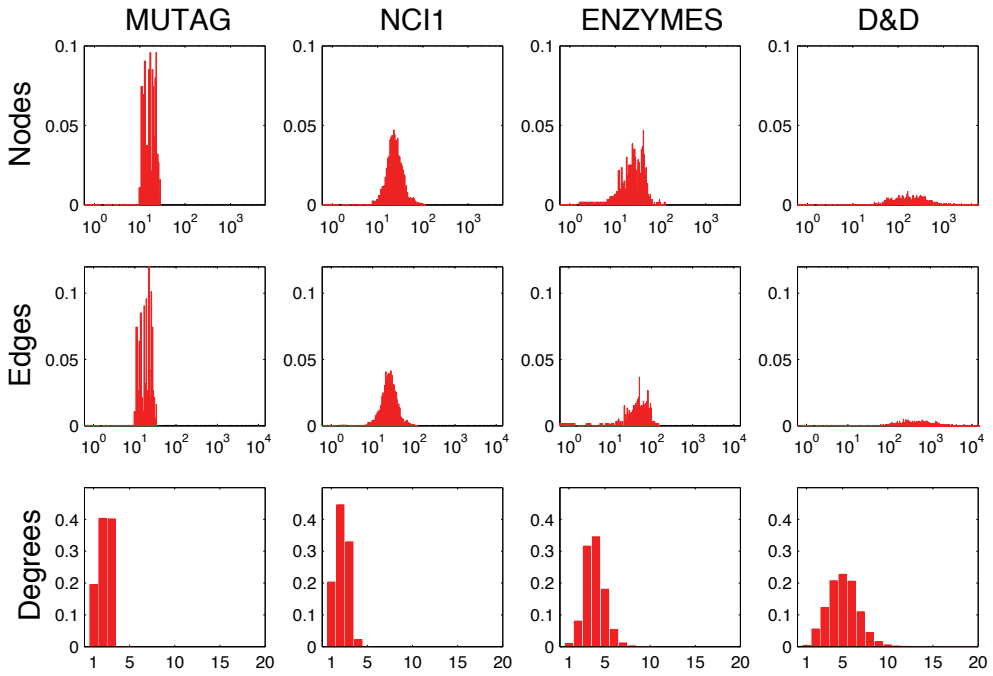


Figure 4: The rows illustrate the distributions of node number, edge number, and degree in data sets MUTAG, NCI1, ENZYMES and D&D. We omitted NCI109, as its node number, edge number, and degree distributions are similar to those of NCI1.

From the family of kernels based on walks, we compared our new kernels to the fast geometric random walk kernel by Vishwanathan et al. (2010) that counts common labeled walks, and to the p-random walk kernel that compares random walks up to length p in two graphs (a special case of random walk kernels Kashima et al., 2003; Gärtner et al., 2003).

From the family of kernels based on limited-size subgraphs, we chose an extension of the graphlet kernel by Shervashidze et al. (2009) that counts common induced labeled connected subgraphs of size 3.

From the family of kernels based on paths, we compared to the shortest path kernel by Borgwardt and Kriegel (2005) that counts pairs of labeled nodes with identical shortest path length.

Note that whenever possible, we used fast computation schemes based on explicitly computing the feature map (similar to that in Algorithm 2) before taking the inner product, in order to speed up kernel computation. In particular, we used this technique for computing shortest path and graphlet kernels. For connected 3-node graphlet kernels it is rather intuitive to imagine the explicit feature map: First, we have only 4 types of different graphlets with 3 nodes. Second, for each type of graphlet we can determine the number of possible labelings of the three nodes as a function of the size of the node label alphabet. In the case of the shortest path kernel, the explicit feature map may or may not exist. In our experiments, as edges were not weighted, we used the number of edges in a

path as a measure of its length. Moreover, we used the Dirac kernel on shortest path distances. This allowed us to explicitly compute the feature map corresponding to the shortest path kernel for each graph in all data sets. We were able to compute the explicit feature maps corresponding to the WL edge and WL shortest path up to and including $h = 3$ and $h = 2$ respectively on all data sets except the largest one, D&D (which also has the largest original node label alphabet), because of the large number of compressed labels. In the case of this data set, we used the pairwise edge (resp. shortest path) comparison scheme described in Sections 3.3 and 3.4.

We performed 10-fold cross-validation of C-Support Vector Machine Classification using LIBSVM (Chang and Lin, 2001), using 9 folds for training and 1 for testing. All parameters of the SVM were optimised on the training data set only. To exclude random effects of fold assignments, we repeated the whole experiment 10 times. We report average prediction accuracies and standard deviations in Tables 1 and 2.

We chose h for our Weisfeiler-Lehman subtree kernel by cross-validation on the training data set for $h \in \{0,1,\ldots,10\}$, which means that we computed 11 different WL subtree kernel matrices in each experiment. In the case of the WL edge and WL shortest path kernels, h was chosen by cross-validation for $h \in \{0,1,2,3\}$ and $h \in \{0,1,2\}$ respectively. We reported the total runtime of these computations (*not* the average per kernel matrix).

Note that all kernel matrices in Table 2 which needed more than 3 days to be computed on one machine were computed on a cluster by distributing different blocks of the kernel matrix to be computed to different nodes. The reported runtime is the sum of the runtimes required to obtain each block.

Proceeding in the same fashion as in the case of the Weisfeiler-Lehman subtree kernel, we computed the Ramon-Gärtner subtree and Weisfeiler-Lehman shortest path kernels for $h \in \{0,1,2\}$ and the p-random walk kernel for $p \in \{1,\ldots,10\}$. We computed the random walk kernel for λ chosen from the set $\{10^{-2}, 10^{-3}, \ldots, 10^{-6}\}$ for smaller data sets and did not observe a large variation in the resulting accuracy. For this reason and because of the relatively high runtime needed to compute this kernel on larger data sets (see Table 2), we set λ as the largest power of 10 smaller than the inverse of the squared maximum degree in the data set.

4.2.3 RESULTS

In terms of runtime, the Weisfeiler-Lehman subtree kernel could easily scale up even to graphs with thousands of nodes. On D&D, subtree-patterns of height up to 10 were computed in 11 minutes, while no other comparison method could handle this data set in less than half an hour. The shortest path kernel, the WL edge kernel and the WL shortest path kernel were competitive to the WL subtree kernel on smaller graphs (MUTAG, NCI1, NCI109, ENZYMES), but on D&D their runtime degenerated to more than 23 hours for the shortest path kernel, to 3 days for the WL edge kernel, and to more than a year for the WL shortest path kernel. The Ramon and Gärtner kernel was computable on MUTAG in approximately 40 minutes, but it finished computation in more than a month on ENZYMES and the computation took even longer time on larger data sets. The random walk kernel was competitive on MUTAG and ENZYMES in terms of runtime, but took more than a week on each of the NCI data sets and more than a month on D&D. The fact that the random walk kernel was competitive on the smallest of our data sets, MUTAG, is not surprising, as on this data set one could also afford using kernels with exponential runtime, such as the all paths kernel (Gärtner et al., 2003). The graphlet kernel was faster than our WL subtree kernel on MUTAG and

Method/Data Set	MUTAG	NCI1	NCI109	ENZYMES	D & D
WL subtree	82.05 (±0.36)	82.19 (± 0.18)	82.46 (±0.24)	52.22 (±1.26)	79.78 (±0.36)
WL edge	81.06 (±1.95)	84.37 (±0.30)	84.49 (±0.20)	53.17 (±2.04)	77.95 (±0.70)
WL shortest path	83.78 (±1.46)	84.55 (±0.36)	83.53 (±0.30)	59.05 (±1.05)	79.43 (±0.55)
Ramon & Gärtner	85.72 (±0.49)	61.86 (±0.27)	61.67 (±0.21)	13.35 (±0.87)	57.27 (±0.07)
p-random walk	79.19 (±1.09)	58.66 (±0.28)	58.36 (±0.94)	27.67 (±0.95)	66.64 (±0.83)
Random walk	80.72 (±0.38)	64.34 (±0.27)	63.51 (± 0.18)	21.68 (±0.94)	71.70 (±0.47)
Graphlet count	75.61 (±0.49)	66.00 (±0.07)	66.59 (±0.08)	32.70 (±1.20)	78.59 (±0.12)
Shortest path	87.28 (±0.55)	73.47 (±0.11)	73.07 (±0.11)	41.68 (±1.79)	78.45 (±0.26)

Table 1: Prediction accuracy (± standard deviation) on graph classification benchmark data sets

the NCI data sets, and about a factor of 3 slower on D&D. However, this efficiency came at a price, as the kernel based on size-3 graphlets turned out to lead to poor accuracy levels on four data sets.

Data Set	MUTAG	NCI1	NCI109	ENZYMES	D & D
Maximum # nodes	28	111	111	126	5748
Average # nodes	17.93	29.87	29.68	32.63	284.32
# labels	7	37	38	3	82
Number of graphs	188	4110	4127	600	1178
WL subtree	6”	7’20”	7’21”	20”	11’0”
WL edge	3”	1’5”	58”	11”	3 days
WL shortest path	2”	2’20”	2’23”	1’3”	484 days
Ramon & Gärtner	40’6”	81 days	81 days	38 days	103 days
p-random walk	4’42”	5 days	5 days	10’	4 days
Random walk	12”	9 days	9 days	12’19”	48 days
Graphlet count	3”	1’27”	1’27”	25”	30’21”
Shortest path	2”	4’38”	4’39”	5”	23h 17’2”

Table 2: CPU runtime for kernel computation on graph classification benchmark data sets

On NCI1, NCI109, ENZYMES and D&D, the kernels from the Weisfeiler-Lehman framework reached the highest accuracy. While on NCI1, NCI109, and D&D the results of all three WL kernels were competitive with each other, on ENZYMES the WL shortest path kernel dramatically improved over the other two WL kernels. On D&D the shortest path and graphlet kernels yielded similarly good results, while on NCI1 and NCI109 the Weisfeiler-Lehman subtree kernel improved by more than 8% the best accuracy attained by other methods. On MUTAG, the WL kernels reached the third, the fourth and the fifth best accuracy levels among all methods considered.

The labeled size-3 graphlet kernel achieved low accuracy levels, except on D&D. The random walk and the *p*-random walk kernels, as well as the Ramon-Gärtner kernel, were less competitive to kernels that performed the best on data sets other than MUTAG.

It is worth mentioning that in the case of WL edge and WL shortest path kernels, the values 2 and 3 of h were almost always chosen by the cross-validation procedure, meaning that the kernels comparing edges and shortest paths on Weisfeiler-Lehman graphs of positive height systematically improved the accuracy of the base kernel (corresponding to $h = 0$).

To summarize, the WL subtree kernel turned out to be competitive in terms of runtime on all smaller data sets, fastest on the large protein data set, and its accuracy levels were competitive on all data sets. The WL edge kernel performed slightly better than the WL subtree kernel on three out of five data sets in terms of accuracy. The WL shortest path kernel achieved the highest accuracy level on two out of five data sets, and was competitive on the remaining data sets.

5. Conclusions

We have defined a general framework for constructing graph kernels on graphs with unlabeled or discretely labeled nodes. Instances of our framework include a fast subtree kernel that combines scalability with the ability to deal with node labels. Our kernels are competitive in terms of accuracy with state-of-the-art kernels on several classification benchmark data sets, even reaching the highest accuracy level on four out of five data sets. Moreover, in terms of runtime on large graphs, instances of our kernel outperform other kernels, even the efficient computation schemes for random walk kernels (Vishwanathan et al., 2010) and graphlet kernels (Shervashidze et al., 2009) that were recently developed.

Our new kernels open the door to applications of graph kernels on large graphs in bioinformatics, for instance, protein function prediction via detailed graph models of protein structure on the amino acid level, or on gene networks for phenotype prediction. An exciting algorithmic question for further studies will be to consider kernels on graphs with continuous or high-dimensional node labels and their efficient computation.

Acknowledgments

The authors would like to thank Ulrike von Luxburg for useful comments. N. S. was funded by the DFG project "Kernels for Large, Labeled Graphs (LaLa)".

References

L. Babai and L. Kucera. Canonical labelling of graphs in linear average time. In *Proceedings Symposium on Foundations of Computer Science*, pages 39–46, 1979.

F. R. Bach. Graph kernels between point clouds. In *Proceedings of the International Conference on Machine Learning*, pages 25–32, 2008.

K. M. Borgwardt and H.-P. Kriegel. Shortest-path kernels on graphs. In *Proceedings of the International Conference on Data Mining*, pages 74–81, 2005.

K. M. Borgwardt, C. S. Ong, S. Schönauer, S. V. N. Vishwanathan, A. J. Smola, and H. P. Kriegel. Protein function prediction via graph kernels. *Bioinformatics*, 21(Suppl 1):i47–i56, Jun 2005.

H. Bunke and G. Allermann. Inexact graph matching for structural pattern recognition. *Pattern Recognition Letters*, 1:245–253, 1983.

J.-Y. Cai, M. Fürer, and N. Immerman. An optimal lower bound on the number of variables for graph identification. *Combinatorica*, 12(4):389–410, 1992.

C.-C. Chang and C.-J. Lin. *LIBSVM: A Library For Support Vector Machines*, 2001. Software available at http://www.csie.ntu.edu.tw/∼cjlin/libsvm.

F. Costa and K. De Grave. Fast neighborhood subgraph pairwise distance kernel. In *Proceedings of the International Conference on Machine Learning*, pages 255–262, 2010.

A. K. Debnath, R. L. Lopez de Compadre, G. Debnath, A. J. Shusterman, and C. Hansch. Structure-activity relationship of mutagenic aromatic and heteroaromatic nitro compounds. correlation with molecular orbital energies and hydrophobicity. *J. Med. Chem.*, 34:786–797, 1991.

P. D. Dobson and A. J. Doig. Distinguishing enzyme structures from non-enzymes without alignments. *J. Mol. Biol.*, 330(4):771–783, Jul 2003.

H. Fröhlich, J. Wegner, F. Sieker, and A. Zell. Optimal assignment kernels for attributed molecular graphs. In *Proceedings of the International Conference on Machine Learning*, pages 225–232, Bonn, Germany, 2005.

M. R. Garey and D. S. Johnson. *Computers and Intractability, A Guide to the Theory of NP-Completeness*. W.H. Freeman and Company, New York, 1979.

T. Gärtner, P. A. Flach, and S. Wrobel. On graph kernels: Hardness results and efficient alternatives. In *Proceedings of the Annual Conference on Computational Learning Theory*, pages 129–143, 2003.

Z. Harchaoui and F. Bach. Image classification with segmentation graph kernels. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2007.

D. Haussler. Convolutional kernels on discrete structures. Technical Report UCSC-CRL-99 - 10, Computer Science Department, UC Santa Cruz, 1999.

S. Hido and H. Kashima. A linear-time graph kernel. In *Proceedings of the International Conference on Data Mining*, pages 179–188, 2009.

T. Hofmann, B. Schölkopf, and A. J. Smola. Kernel methods in machine learning. *Annals of Statistics*, 36(3):1171–1220, 2008.

T. Horváth, T. Gärtner, and S. Wrobel. Cyclic pattern kernels for predictive graph mining. In *Proceedings of the ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 158–167, 2004.

H. Kashima, K. Tsuda, and A. Inokuchi. Marginalized kernels between labeled graphs. In *Proceedings of the International Conference on Machine Learning*, Washington, DC, United States, 2003.

I. R. Kondor and K. M. Borgwardt. The skew spectrum of graphs. In *Proceedings of the International Conference on Machine Learning*, pages 496–503, 2008.

I. R. Kondor, N. Shervashidze, and K. M. Borgwardt. The graphlet spectrum. In *Proceedings of the International Conference on Machine Learning*, pages 529–536, 2009.

P. Mahé and J.-P. Vert. Graph kernels based on tree patterns for molecules. *Machine Learning*, 75 (1):3–35, 2009.

P. Mahé, N. Ueda, T. Akutsu, J. Perret, and J. Vert. Extensions of marginalized graph kernels. In *Proceedings of the International Conference on Machine Learning*, pages 552–559, Alberta, Canada, 2004.

K. Mehlhorn. *Data Structures and Efficient Algorithms*. Springer, 1984.

H. L. Morgan. The generation of unique machine description for chemical structures - a technique developed at chemical abstracts service. *Journal of Chemical Documentation*, 5(2):107–113, 1965.

M. Neuhaus and H. Bunke. Self-organizing maps for learning the edit costs in graph matching. *IEEE Transactions on Systems, Man, and Cybernetics, Part B*, 35(3):503–514, 2005.

J. Ramon and T. Gärtner. Expressivity versus efficiency of graph kernels. Technical report, First International Workshop on Mining Graphs, Trees and Sequences (held with ECML/PKDD'03), 2003.

B. Schölkopf and A. J. Smola. *Learning with Kernels*. MIT Press, Cambridge, MA, 2002.

I. Schomburg, A. Chang, C. Ebeling, M. Gremse, C. Heldt, G. Huhn, and D. Schomburg. Brenda, the enzyme database: updates and major new developments. *Nucleic Acids Research*, 32D:431–433, 2004.

N. Shervashidze and K. M. Borgwardt. Fast subtree kernels on graphs. In Y. Bengio, D. Schuurmans, J. Lafferty, C. K. I. Williams, and A. Culotta, editors, *Proceedings of the Conference on Advances in Neural Information Processing Systems*, pages 1660–1668, 2009.

N. Shervashidze, S.V.N. Vishwanathan, T. Petri, K. Mehlhorn, and K.M. Borgwardt. Efficient graphlet kernels for large graph comparison. In David van Dyk and Max Welling, editors, *Proceedings of the International Conference on Artificial Intelligence and Statistics*, 2009.

F. Suard, V. Guigue, A. Rakotomamonjy, and A. Benshrair. Pedestrian detection using stereo-vision and graph kernels. In *IEEE Symposium on Intelligent Vehicles*, 2005.

J.-P. Vert. The optimal assignment kernel is not positive definite. *CoRR*, abs/0801.4061, 2008.

J.-P. Vert, T. Matsui, S. Satoh, and Y. Uchiyama. High-level feature extraction using svm with walk-based graph kernel. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 1121–1124, 2009.

S. V. N. Vishwanathan, N. N. Schraudolph, I. R. Kondor, and K. M. Borgwardt. Graph kernels. *Journal of Machine Learning Research*, 11:1201–1242, 2010.

N. Wale and G. Karypis. Comparison of descriptor spaces for chemical compound retrieval and classification. In *Proceedings of the International Conference on Data Mining*, pages 678–689, Hong Kong, 2006.

B. Weisfeiler and A. A. Lehman. A reduction of a graph to a canonical form and an algebra arising during this reduction. *Nauchno-Technicheskaya Informatsia, Ser. 2*, 9, 1968.

Journal of Machine Learning Research 12 (2011) 2563-2581 Submitted 7/10; Revised 2/11; Published 9/11

Kernel Analysis of Deep Networks

Grégoire Montavon GMONTAVON@CS.TU-BERLIN.DE
Mikio L. Braun MIKIO@CS.TU-BERLIN.DE
Klaus-Robert Müller* KLAUS-ROBERT.MUELLER@TU-BERLIN.DE
Machine Learning Group
Technische Unversität Berlin
Franklinstr. 28/29
10587 Berlin, Germany

Editor: Yoshua Bengio

Abstract

When training deep networks it is common knowledge that an efficient and well generalizing representation of the problem is formed. In this paper we aim to elucidate what makes the emerging representation successful. We analyze the layer-wise evolution of the representation in a deep network by building a sequence of deeper and deeper kernels that subsume the mapping performed by more and more layers of the deep network and measuring how these increasingly complex kernels fit the learning problem. We observe that deep networks create increasingly better representations of the learning problem and that the structure of the deep network controls how fast the representation of the task is formed layer after layer.

Keywords: deep networks, kernel principal component analysis, representations

1. Introduction

Finding an appropriate representation of data is a central problem in machine learning. The representation should ideally distill the relevant information about a learning problem in a compact manner, such that it becomes possible to learn the data from a small number of examples.

Deep networks (e.g., Rumelhart et al., 1986; Hinton et al., 2006) have shown promise by automatically extracting representations from raw data. Through their deep multi-layered architecture, simpler and more accurate representations of the learning problem can be built layer after layer. Their depth makes possible the creation of abstractions that are important in order to learn the desired well-generalizing representation. Also, their flexibility offers the possibility to systematically and structurally incorporate prior knowledge, for example, by constraining the connectivity of the deep network (e.g., LeCun, 1989; Lang et al., 1990), by learning multiple tasks at the same time (Caruana, 1997; Collobert and Weston, 2008) or by regularizing the solution with unlabeled samples (Salakhutdinov and Hinton, 2007; Weston et al., 2008). Such prior knowledge can significantly improve the generalization ability of deep networks, leading to state-of-the-art performance on several complex real-world data sets.

While a considerable amount of work has been dedicated to learning efficiently deep architectures (Orr and Müller, 1998; Hinton et al., 2006; Bengio et al., 2006), leading to simple and efficient training algorithms, these learning machines still lack of analytic understanding. Recently,

∗. Also at the Institute for Pure & Applied Mathematics, University of California, Los Angeles, Los Angeles, CA 90095.

a significant amount of research has focused on improving our theoretical understanding of deep networks, in particular, understanding the benefits of unsupervised pretraining (Erhan et al., 2010), understanding what are the main difficulties when training deep networks (Larochelle et al., 2009) and studying the invariance of representations built in deep networks (Goodfellow et al., 2009). However, quantifying how good hidden representations are and measuring how the representation evolves layer after layer are still open questions. Overall, deep networks are thus generally assumed to be powerful and flexible learning machines that are however not well understood theoretically (Bengio, 2009).

In parallel to the development of deep networks, kernel methods (Müller et al., 2001; Schölkopf and Smola, 2002) offer an elegant framework that decouples learning algorithms from data representations. The kernel operator $k(x,x')$—a central concept of the kernel framework—measures the similarity between two points x and x' of the input distribution, yielding an implicit kernel feature map $x \mapsto \phi(x)$ (Schölkopf et al., 1999) that ideally implements all the prior knowledge of the learning problem contained in the kernel operator. This decoupling between learning algorithms and data representations opens the door to a whole world of generic learning machines and data analysis tools such as support vector machines (Cortes and Vapnik, 1995), kernel discriminant analysis (Mika et al., 1999; Baudat and Anouar, 2000; Mika et al., 2003) and kernel principal component analysis (Schölkopf et al., 1998) that can be applied independently of the data set. The kernel framework has also been used as an abstraction tool for modeling complex real systems such as the visual cortex (Smale et al., 2010).

The goal of this paper is to study in the light of the kernel framework how exactly the representation is built in a deep network, in particular, how the representation evolves as we map the input through more and more layers of the deep network. Here, the kernel framework is not used as an effective learning machine, but as an abstraction tool for modeling the deep network. Our analysis takes a trained deep network $f(x) = f_L \circ \cdots \circ f_1(x)$ as input, defines a sequence of "deep kernels"

$$\begin{aligned} k_0(x,x') &= k_{RBF}(x,x'), \\ k_1(x,x') &= k_{RBF}(f_1(x), f_1(x')), \\ &\vdots \\ k_L(x,x') &= k_{RBF}(f_L \circ \cdots \circ f_1(x), f_L \circ \cdots \circ f_1(x')) \end{aligned}$$

that subsume the mapping performed by more and more layers of the deep network and outputs how good the representations yielded by these deeper and deeper kernels are. We quantify for each kernel how good the representation with respect to the learning problem is by measuring how much task-relevant information is contained in the leading principal components of the kernel feature space. This method is based on the theoretical results of Braun (2006) and Braun et al. (2008) which show that eigenvalues and projections to eigenspaces of the kernel matrix have small approximation errors, even for already a small number of samples.

This analysis allows us for the first time to observe and quantify the evolution of the representation in deep networks. We use our analysis to test two hypotheses on deep networks:

> Hypothesis 1: *as the input is propagated through more and more layers of the deep network, simpler and more accurate representations of the learning problem are obtained.*

Indeed, as the input is mapped through more and more layers, abstractions learned by the deep network are likely to change the perception of whether a task is simple or not. For example, in

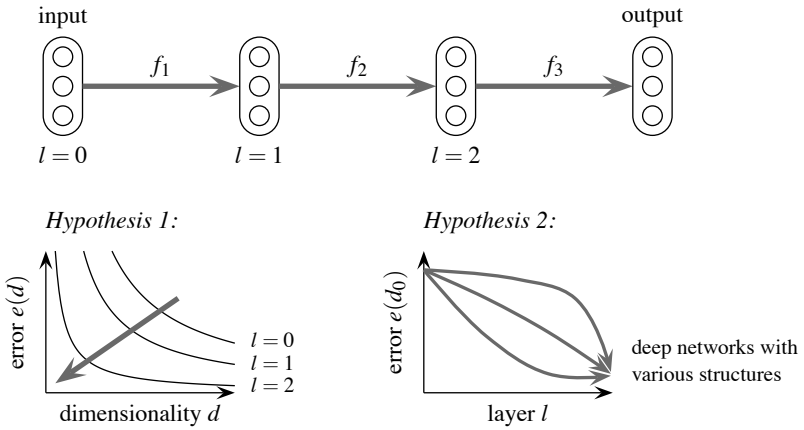


Figure 1: Illustration of our analysis. Curves on the left plot relate the simplicity (dimensionality) and accuracy (error) of the representation of the learning problem at each layer of the deep network. The dimensionality is measured as the number of kernel principal components on which the representation is projected. The thick gray arrows indicate the forward path of the deep network. Hypothesis 1 states that as deeper and deeper kernels are built, simpler and more accurate representations of the learning problem are obtained. Hypothesis 2 states that the structure of the deep network controls the way the solution is formed layer after layer.

the context of image classification, classifying between cat and dog would appear simpler in the last layers of the deep network than in the first layers since irrelevant factors of variation such as occlusion and orientation would be progressively filtered out by the hierarchy of abstractions built in the deep network.

Hypothesis 2: *the structure of the deep network controls how fast the representation of the task is formed layer after layer.*

It has been empirically corroborated that carefully regularizing the training process by means of specific learning rates, weight penalties, initial weights, shared weights or restricted connectivity can greatly improve the generalization of deep networks (LeCun, 1989; Orr and Müller, 1998; Hinton et al., 2006). We hypothesize that a common aspect of these various regularization techniques is to control the layer-wise evolution of the representation through the deep network. On the other hand, a simple unregularized deep network may make inefficient use of the representational power of deep networks, distributing the discrimination steps across layers in a suboptimal way.

These two hypotheses are illustrated in Figure 1. Testing them are, to our opinion, of significant importance as they might shed light on the nature of deep learning and on the way complex problems are to be solved. This paper completes our conference paper (Montavon et al., 2010) by extending the discussion on the interest of analyzing deep networks within the kernel framework and by extending the empirical study to more data sets and larger deep networks.

1.1 Related Work

The concept of building kernels imitating the structure of deep architectures—or more simply, building "deep kernels"—is not new. Cho and Saul (2009) already expressed deep architectures as kernels in order to solve a convex optimization problem and achieve large margin discrimination in a deep network. This approach differs from our work in the sense that their deep kernel is not used as an analysis tool for trained deep networks but as part of an effective learning machine.

The concept was also developed in Smale et al. (2010) where the authors give a recursive definition of the neural response as hierarchy of simple kernels operating on subparts of the sensory input and in Wibisono et al. (2010) where a principal component analysis is performed on top of these deep kernels in order to measure invariance properties of deep networks. While the last authors focus mostly on the representation of data in static deep architectures made of predefined features, we are considering instead trainable deep architectures.

Although not directly using the kernel framework, Goodfellow et al. (2009) also analyze the layer-wise evolution of the representation in deep networks, showing that deep networks trained in an unsupervised fashion build increasing levels of invariance with respect to several engineered transformations of the input and to temporal transformations in video data.

2. Theory

Before being able to observe the layer-wise evolution of the representation in deep networks, we first need quantify how good a representation is with respect to the learning problem. The representation is said to be *good* if simple and accurate models of the learning problem can be built on top of it. We measure it by means of an analysis based on kernel principal component analysis that determines how much of the relevant problem subspace is contained in the leading kernel principal components, more precisely, how well the learning problem can be solved from the leading kernel principal components. The analysis extends naturally to deep networks by building a sequence of kernels that subsume the mapping performed by more and more layers of the deep network and repeating the analysis for these deeper and deeper kernels.

2.1 Quantifying How Good a Representation Is

In this section, we are interested in quantifying how good a representation is with respect to the learning problem. The representation is *good* when it is possible to build models of the learning problem on top of it that are both *simple* and *accurate*.

A first technical difficulty is to quantify how simple a model is. Indeed, the notion of simplicity is highly subjective (Bousquet et al., 2004) and typically depends on which prior knowledge on the learning problem is taken for granted. For example, visual recognition tasks are very simple for humans, but very complex for simple learning algorithms such as a local learning machine. In this example, humans possess a form of prior on how the image should look like (e.g., we know how to classify real images from artificial images) and a machinery to make sense more easily of this complex data.

We choose to model this prior by isolating it into a kernel operator that measures how similar two data points drawn from the input distribution are. For example, a local predictor could be modeled with a Gaussian kernel while a more intelligent human-like predictor should be modeled with a more complex kernel encoding translation invariance, rotation invariance, etc. Then, the

induced kernel feature map $x \mapsto \phi(x)$ encodes implicitly all the prior defined in the kernel with the advantage that linear models can be built on top of it (Schölkopf et al., 1999).

A second technical difficulty comes from the fact that accuracy and simplicity are not always measurable in practice: accuracy of a model can only be estimated up to a certain precision from the finite data set and estimating the simplicity depends on whether we consider, for example, the number of parameters of a model, its entropy or its algorithmic complexity. For these reasons, we need to restrict ourselves to a class of models whose simplicity and accuracy can be easily measured and that are expressive enough to solve the learning problem.

We choose to use the kernel principal component analysis (kernel PCA, Schölkopf et al., 1998) as a basis for building measurably simple models of the learning problem. Our method consists of projecting the input distribution on the d first components (in terms of variance) of the kernel feature space and fitting a linear model on this low-rank representation. The number of components d controls the simplicity of the model. When d is small, the model is simple. When d is large, the model is complex. The accuracy can in turn be obtained by measuring the prediction error $e(d)$ of a linear predictor on top of the d-component kernel representation. We refer to the parameter d as the dimensionality of the model and $e(d)$ as the prediction error obtained with the d-component model. The curve $e(d)$ gives a complete picture of how good a representation is with respect to the learning problem. Figure 2 gives some examples of curves $e(d)$ and explains how these curves can be interpreted.

An advantage of the kernel PCA method is that there exists a theoretical framework and convergence bounds for the estimation of spectral properties from a limited number of samples drawn from the input distribution. In the case of fixed kernels, Braun et al. (2008) show that the projections to kernel principal components obtained with a finite and typically small number of samples n are close with essentially multiplicative errors to those that would be obtained in the asymptotic case where $n \to \infty$. This result can be naturally extended to a finite set of kernels. These convergence properties are desirable since the data distribution is unknown and only a finite number of observations are available for our analysis. Appendix A gives some additional information on the convergence of kernel principal components.

A second advantage of the kernel PCA method is the high flexibility that it offers with respect to the nature of the learning problem. Kernel PCA is not only independent of the input representation due to the kernel embedding, but also independent of the output representation. Indeed, kernel PCA simply acts as a regularizer on the kernel feature space that limits the complexity of the subsequent learning machine. Therefore any discriminative model can be used on top of the regularized representation, allowing to treat various classes of problems such as binary classification, multi-class classification or regression within the same framework.

To summarize, the kernel framework combines the four requirements of our analysis: (1) the kernel operator expresses and isolates the subjective notion of simplicity, (2) the complexity of the model is controlled by projecting the input distribution on a limited number of kernel principal components, (3) convergence bounds allow to effectively measure the accuracy of the model and (4) various models can be built on top of the leading kernel principal components in order to express the various types of learning problems (regression, classification, ...) that arise in real applications.

We present below the computation steps required to estimate how good a kernel k and its associated feature map $x \mapsto \phi(x)$ are with respect to a learning problem $p(x,y)$. Let $\{(x_1,y_1),\ldots,(x_n,y_n)\}$ be a data set of n points drawn independently from $p(x,y)$. Let $X = (x_1,\ldots,x_n)$ and $Y = (y_1,\ldots,y_n)$ be the matrices associated to the inputs and labels of the data set. We compute the kernel matrix K

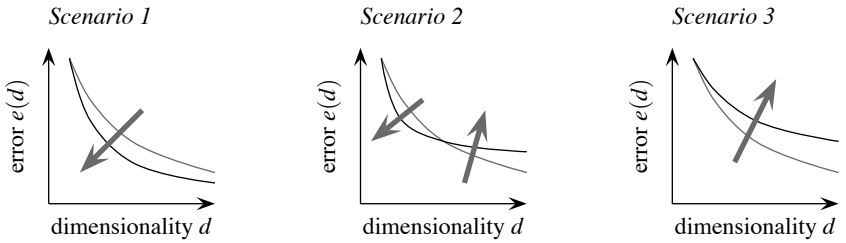


Figure 2: Effect of converting a representation of the learning problem $p(x,y)$ (gray curve) to a new representation of the learning problem $p(f(x),y)$ (black curve) where the input x is mapped to $f(x)$. We can distinguish three scenarios: (Scenario 1) the mapping produces a better representation from which more accurate models are obtained for every dimensionality—this is the desired behavior of deep networks,—(Scenario 2) the mapping concentrates the label information in the leading kernel principal components but also loses some information—lossy feature extractors typically fall into that category—and (Scenario 3) the mapping makes the learning problem more complex—this would be the result of introducing noise or throwing away label information.

associated to the data set:

$$K = \begin{pmatrix} k(x_1,x_1) & \dots & k(x_1,x_n) \\ \vdots & & \vdots \\ k(x_n,x_1) & \dots & k(x_n,x_n) \end{pmatrix}.$$

The kPCA components $u_1,\dots,u_n$ are obtained by performing an eigendecomposition of K where eigenvectors $u_1,\dots,u_n$ have unit length and eigenvalues $\lambda_1,\dots,\lambda_n$ are sorted by decreasing magnitude:

$$K = (u_1|\dots|u_n)\cdot \mathrm{diag}(\lambda_1,\dots,\lambda_n)\cdot (u_1|\dots|u_n)^\top.$$

Let $\hat{U} = (u_1|\dots|u_d)$ and $\hat{\Lambda} = \mathrm{diag}(\lambda_1,\dots,\lambda_d)$ be a d-dimensional approximation of the eigendecomposition. The space spanned by this basis approximates the space spanned by the d leading components of the infinite-dimensional kernel feature space associated to the probability distribution $p(x)$. In this space, the learning problem can be solved by a standard linear or logistic regression model. For regression problems, we fit a linear model $\beta^\star$ that maps the d leading components to the output:

$$\beta^\star = \mathrm{argmin}_\beta ||\hat{U}\beta - Y||_F^2 = \hat{U}^\top Y. \tag{1}$$

For classification problems, instead of fitting the model directly on the outputs, we fit the model on the log-likelihood of classes

$$\beta^\star = \mathrm{argmin}_\beta \prod_{i=1}^{n} \mathrm{softmax}([\hat{U}\beta]_i)_{y_i} \tag{2}$$

where $\mathrm{softmax}(z) = e^z / \sum_j e^{z_j}$ converts a vector z into a probability distribution over classes. Note that the optimization criterion only consists of the empirical risk minimization term and lacks a

regularization term. Indeed, the regularization is implicitly carried out by the projection on the d leading principal components. The problem is therefore well-posed only when $d \ll n$. Once the model $\beta^\star$ is computed, the estimated outputs can be calculated as

$$\hat{Y} = \hat{U}\beta^\star$$

for regression problems and as

$$\hat{y}_i = \text{argmax}([\hat{U}\beta^\star]_i) \qquad 1 \leq i \leq n$$

for classification problems. The training error is estimated as

$$e(d) = \frac{1}{n}\sum_{i=1}^{n} ||\hat{y}_i - y_i||^2 \tag{3}$$

for regression problems and as

$$e(d) = \frac{1}{n}\sum_{i=1}^{n} 1_{\hat{y}_i \neq y_i} \tag{4}$$

for classification problems. The test error can be obtained by cross-validating the linear model on random partitions of $(x_1, \ldots, x_n)$. Training and test error can be used as approximation bounds for the asymptotic case $n \to \infty$ where the model $\beta^\star$ would minimize the error on the real distribution $p(x,y)$. In the next sections, the upper and lower approximation bounds are respectively depicted as solid and dotted lines in Figure 5, 6 and 7.

2.2 Application to Deep Networks

In this section, we describe how the analysis of representations presented above can be used to measure the layer-wise forming of the representation in deep networks. Let $f(x) = f_L \circ \cdots \circ f_1(x)$ be a trained deep network of L layers. Our analysis consists of defining a sequence of "deep kernels"

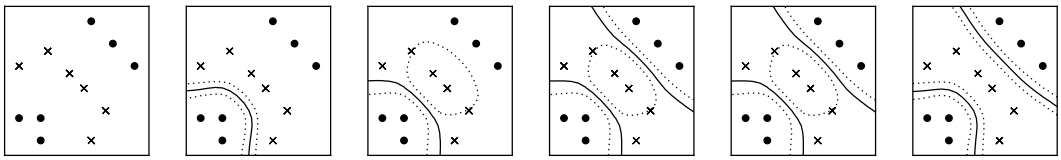
$$\begin{aligned} k_0(x,x') &= k_{RBF}(x,x'), \\ k_1(x,x') &= k_{RBF}(f_1(x), f_1(x')), \\ &\vdots \\ k_L(x,x') &= k_{RBF}(f_L \circ \cdots \circ f_1(x), f_L \circ \cdots \circ f_1(x')) \end{aligned}$$

that subsume the mapping performed by more and more layers of the deep network and repeating for each kernel the analysis presented in Section 2.1. Algorithm 1 summarizes the main computational steps of our analysis. The kernel k_{RBF} is the standard Gaussian kernel defined as $k_{RBF}(x,x') = \exp(-||x-x'||^2/2\sigma^2)$.

The main prior encoded by Gaussian kernels is the smoothness of the task of interest in the input space (Smola et al., 1998). Gaussian kernels are appropriate when two neighboring samples (in terms of Euclidean distance) are likely to have the same class. It remains to see how the concept of simplicity encoded by the Gaussian kernel can be understood from the perspective of the induced prediction model. Figure 3 shows that the simplicity of the model can be related to the number of allowed local variations in the input space. When d increases, more variations of the

Algorithm 1: Main computational steps of our layer-wise analysis of deep networks. At every layer of the deep network, the same analysis is performed, returning a list of curves $e(d)$ capturing the evolution of the representation in the deep network.

Input: A data set $\{(x_1,y_1),\ldots,(x_n,y_n)\}$
A deep network $f: x \mapsto f_L \circ \cdots \circ f_1(x)$
Output: The curves $e(d)$ for each layer l
for $l \in \{1,\ldots,L\}$ **do**
  for $\sigma \in \Sigma$ **do**
    $k(x,x') = k_{RBF(\sigma)}(f_l \circ \cdots \circ f_1(x), f_l \circ \cdots \circ f_1(x'))$
    compute the kernel matrix K associated to $k(x,x')$ and $(x_1,\ldots,x_n)$
    do the eigendecomposition $K = (u_1|\ldots|u_n) \cdot \mathrm{diag}(\lambda_1,\ldots,\lambda_n) \cdot (u_1|\ldots|u_n)^\top$
    for $d \in \{0,1,2,\ldots\}$ **do**
      build a low rank approximation of the input $\hat{U} \leftarrow (u_1|\ldots|u_d)$
      fit the model $\beta^\star$ that predicts $(y_1,\ldots,y_n)$ from $\hat{U}$ (cf. Equation 1 and 2)
      compute the error $e(d,\sigma)$ of the model $\beta^\star$ (cf. Equation 3 and 4)
  $e(d) = \min_\sigma e(d,\sigma)$
  plot the curve $e(d)$



$d=1$ $e(d)=0.5$ | $d=2$ $e(d)=0.25$ | $d=3$ $e(d)=0.25$ | $d=4$ $e(d)=0$ | $d=5$ $e(d)=0$ | $d=6$ $e(d)=0$

Figure 3: Interpretation of a prediction model based on the leading components of the Gaussian kernel on a toy data set. As we add more and more leading components of the kernel, the model becomes more flexible, creating a better decision boundary. Note that with four leading components, all the samples are already perfectly classified.

learning problem can be encoded and the prediction improves. Figure 4 shows that by making the problem increasingly complex—for example, by distorting it—the number of dimensions required to approach the error of the optimal classifier becomes larger and larger.

The notion of simplicity encoded by the Gaussian kernel is meaningful for a wide range of learning problems, however, it does not explain how simple more intelligent systems perceive problems such as vision and speech. Indeed, domain-specific regularities such as invariance to translation, scale or occlusion can not be modeled efficiently by a Gaussian kernel. Consequently, observing the learning problem become simpler as we build deeper and deeper kernels highlights the capacity of deep networks to model the regularities of the input distribution.

A last aspect that has not been discussed yet is how to choose the scale parameter σ of the Gaussian kernel. We decide to choose the parameter σ that minimizes the error $e(d)$, leading to a different scale for each dimensionality. The rationale for taking a different scale for each d is that the

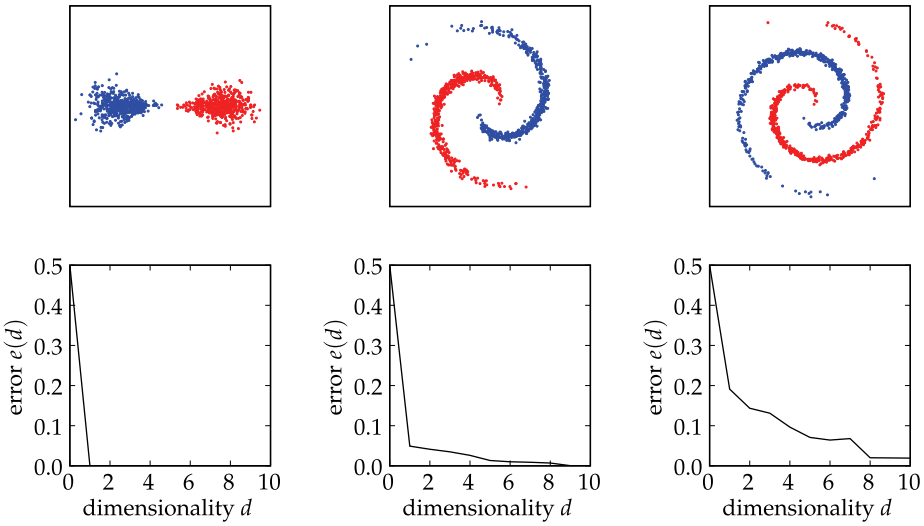


Figure 4: On the top, three increasingly complex representations of a binary classification problem. On the bottom, the curves $e(d)$ quantifying how good the representation of the learning problem is from the perspective of the Gaussian kernel. The original non-distorted learning problem can be solved perfectly with only one kernel principal component. As the input distribution gets distorted more and more, the number of leading components required to solve the learning problem increases, hinting that Gaussian kernels become progressively less suited.

optimal scale parameter typically shrinks as more leading components of the input distribution are observed. This parameter selection method also makes our analysis scale invariant. Scale invariance is desirable since the representation at a given layer of the deep network can take different scales due to the number of nodes contained in each layer or to the multiple types of nonlinearities that can be implemented in deep networks.

3. Methodology

In Section 2, we presented the theory and algorithms required to test our two hypotheses on the evolution of representations in deep networks. To summarize, the main idea of the analysis is to build a set of kernels that subsume the mapping performed at each layer of the deep network and, for each kernel, compute how good the representation is by measuring how many leading components of the kernel feature space are necessary in order to model the learning problem well. It remains to select a set of deep networks and data sets in order to test the hypotheses formulated in Section 1.

We consider the MNIST-10K and CIFAR-bw-10K data sets. These two data sets of 10000 samples each are a trimmed version of the larger MNIST handwritten digits and CIFAR image classification data sets (LeCun et al., 1998; Krizhevsky, 2009). The MNIST-10K data set is a 10-class classification data set that consists of 10000 grayscale images of 28×28 pixels representing handwritten digits and their associated label (a number between 0 and 9). The CIFAR-bw-10K data set is a 10-class classification data set that consists of 10000 grayscale images of 32×32 pixels representing different objects and their associated label (airplane, automobile, bird, cat, deer, dog, frog, horse, ship and truck).

State-of-the-art performance on these data sets is achieved with architectures made of several layers, suggesting that these data sets are well suited to test the first hypothesis stated earlier in the paper, that is, the progressive simplification of the learning problem performed by deep networks. The second hypothesis on the effect of the structure of deep networks can be tested by taking a set of structured and unstructured deep networks and observing how the layer-wise evolution of the representation differs between these deep networks. We consider a multilayer perceptron (MLP), a pretrained multilayer perceptron (PMLP) and a convolutional neural network (CNN).

The multilayer perceptron (MLP, Rumelhart et al., 1986) is built by alternating linear transformations and nonlinearities applied element-wise to the output of the linear transformations. On the MNIST data set, we apply successively the functions

$$\begin{aligned} f_1(x) &= \text{sigm}(w_1 \cdot x + b_1), \\ f_2(x) &= \text{sigm}(w_2 \cdot x + b_2), \\ f_3(x) &= \text{softmax}(v \cdot x) \end{aligned}$$

to the input where weight matrices w_1, w_2, v and biases b_1, b_2 are learned from data and where the size of hidden layers is set to 1600. The sigmoid and softmax functions are defined as $\text{sigm}(x) = e^x/(1+e^x)$ and $\text{softmax}(x) = e^x/\sum_j e^{x_j}$. On the CIFAR data set, the sigmoid nonlinearity is replaced by the rectifying function defined as $\text{rectify}(x) = \max(0,x)$ and the size of hidden layers is set to 3600. Since it has been observed that overparameterizing deep networks generally improves the generalization error, the size of layers is chosen large with the only constraint of computational cost. The MLP is mostly unstructured as any type of solution can emerge from the random weights initialization.

The pretrained multilayer perceptron (PMLP, Hinton et al., 2006; Bengio et al., 2006) referred in this paper as PMLP is a multilayer perceptron that has been pretrained using a deep belief network (DBN, Hinton et al., 2006) and then fine-tuned on the discriminative task. The pretraining procedure aims to build a deep generative model of the input that can be used as a starting point to learn the supervised task. In order to use the same architecture as for the MLP during the fine-tuning procedure, we set the visible and hidden units of the DBN to be binary on the MNIST data set and respectively Gaussian and rectified linear (Nair and Hinton, 2010) on the CIFAR data set. Here, the structure of the deep network is implicitly given by the weights initialization subsequent to the unsupervised pretraining.

The convolutional neural network (CNN, LeCun et al., 1998) is a deep network inspired by the structure of the primary visual cortex (Hubel and Wiesel, 1962). Its particular convolutional structure exploits the spatial invariance of images in order to learn well-generalizing solutions from few labeled samples. It is built by alternating (1) convolutional layers $y = w \circledast x + b$ transforming a set of input features maps $\{x_1, x_2, \dots\}$ into a set of output features maps $\{y_1, y_2, \dots\}$ such that $y_i = \sum_j w_{ij} * x_j + b_i$ and where w_{ij} are convolution kernels, (2) detection layers where a nonlinearity is applied element-wise to the output of the convolutions in order to extract important features and (3) pooling layers subsampling each feature map by a given factor. On the MNIST data set, we apply successively the functions

$$\begin{aligned} f_1(x) &= \text{pooling}(\text{sigm}(w_1 \circledast x + b_1)), \\ f_2(x) &= \text{pooling}(\text{sigm}(w_2 \circledast x + b_2)), \\ f_3(x) &= \text{softmax}(v \cdot x) \end{aligned}$$

to the input where weight tensors w_1, w_2, weight matrix v and biases b_1, b_2 are learned from data, convolution kernels have size 5×5, pooling layers downsample the input by a factor two and the number of feature maps in each layer is set to 100. On the CIFAR data set, the sigmoid nonlinearity is replaced by the rectifying function described above.

The deep networks described above are trained on a supervised task with backpropagation (Rumelhart et al., 1986) and stochastic gradient descent (Bottou, 1991) with minibatches of size 20. The last layer has a L2 weight penalty. The softmax module (Bishop, 1996) optimizes the deep network for maximum likelihood. Weights of each layer l are initialized so that the output is of constant magnitude, thus falling into the correct regime of the subsequent nonlinearity. These deep networks are analyzed in two different settings:

- *Supervised learning:* the deep network is trained in a supervised fashion on the target task (digit classification for the MNIST data set and image classification for the CIFAR data set).
- *Transfer learning:* the deep network is trained in a supervised fashion on a binary classification task that consists of determining whether the sample has been flipped vertically or not.

These settings allow us to measure how the structure contained in deep networks affects different aspects of learning such as the layer-wise organization of the learned solution or the transferability of features from one task to another.

3.1 Experimental Setup

We train the deep networks on the 10000 samples of the data set until a training error of 2.5% is reached. Such stopping criterion ensures that the subsequent solutions have a constant complexity and that the limited capacity of the deep network has no side effect on the structure of the solution. As a sanity check, each architecture has been trained with the regular early stopping criterion on the full MNIST and CIFAR-bw data sets, leading to test errors that are on par with results published in the literature for similar architectures (MNIST-MLP: 1.6%, MNIST-PMLP: 1.3%, MNIST-CNN: 0.9%, CIFAR-bw-MLP: 48.1%, CIFAR-bw-PMLP: 46.8%, CIFAR-bw-CNN: 32.4%).

In our analysis, we estimate the kernel principal components with the 10000 samples used for training the deep network. Therefore, the empirical estimate of the d leading kernel principal components takes the form of d 10000-dimensional vectors, or similarly, of a data set of 10000 d-dimensional mapped samples. A lower bound of $e(d)$ is obtained by fitting and evaluating the linear model with the 10000 mapped samples. An upper bound of $e(d)$ is obtained by two-fold cross-validation (5000 samples to fit the model and the 5000 remaining samples to evaluate it). The set of candidate kernel widths is composed of the $0.1, 0.5$ and 0.9 quantiles of the distribution of distances between pairs of points. It turns out that the effect of the kernel scale is rather small and that no further scale parameters are required. The layers of interest are the input data ($l = 0$) and the output of each layer ($l = 1, 2, \ldots$).

4. Results

In this section, we present the results of our analysis on the evolution of the representation in deep networks. Section 4.1 discusses the empirical observation that deep networks trained on the supervised task produce gradually simpler and more accurate representations of the learning problem.

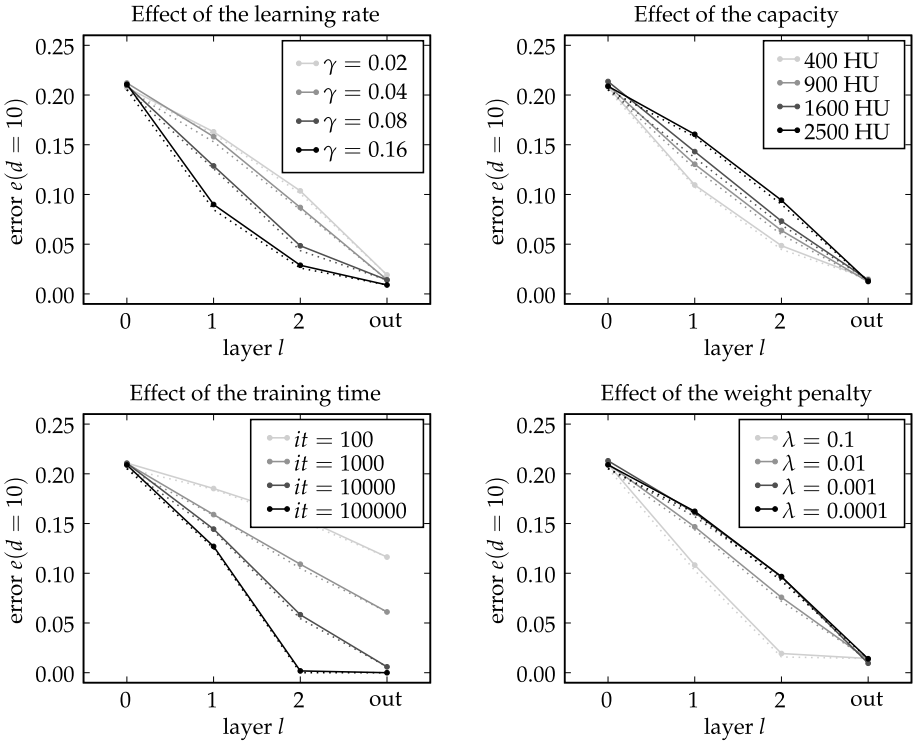


Figure 5: Effect of the learning rate, of the capacity, of the training time and of the weight penalty on the layer-wise evolution of the representation built by an MLP on the MNIST-10K data set. Solid and dotted lines respectively represent the upper and lower approximation bounds of the analysis. As the learning rate increases, the solution tends to make use primarily of the first layers of the deep network. The same effect is observed when we reduce the capacity, increase the weight penalty or increase the training time.

Then, Section 4.2 compares side-by-side the evolution of the representation in different deep networks and discusses the empirical observation that the structure of the deep network controls the layer-wise evolution of the representation in the deep network.

4.1 Better Representations are Built Layer After Layer

It is still an open question how the complex and multimodal form of intelligence observed in living organisms emerges from randomly disposed and locally scoped neurons. Machine learning researchers similarly pointed out that emergent properties also occur in artificial neural networks when trained with simple local algorithms such as Hebbian learning or backpropagation, without having to explicitly define the role of each individual neuron. Also, their ability to simultaneously specialize on specific tasks in output nodes and grow new functionalities from hidden nodes hints that information contained in the underlying distribution of sensed data should be ubiquitous, yet parsimonious where discrimination takes place.

It can be hypothesized that the organization of mapped data distributions within the neural network forms a continuum between general purpose distributions in the middle of the network and

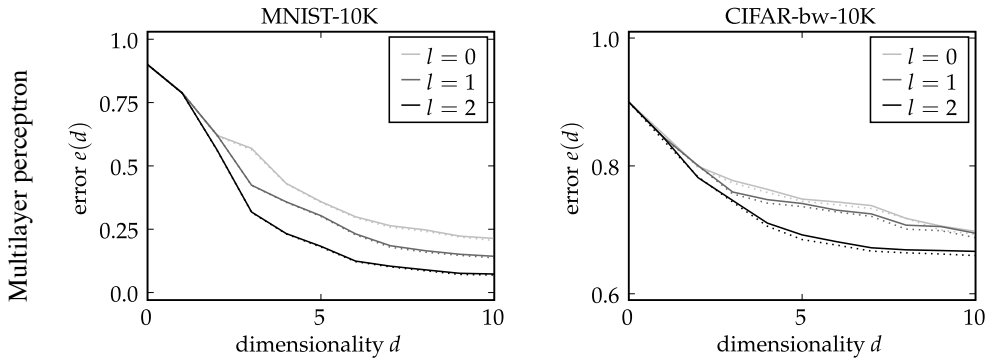


Figure 6: Layer-wise evolution of the error as a function of the number of dimensions when trained on the target task. Solid and dotted lines respectively represent the upper and lower approximation bounds of the analysis. As we move from the first to the last layers, the class information concentrates in the leading components of the mapped data distribution. This observation confirms the first hypothesis depicted in Figure 1.

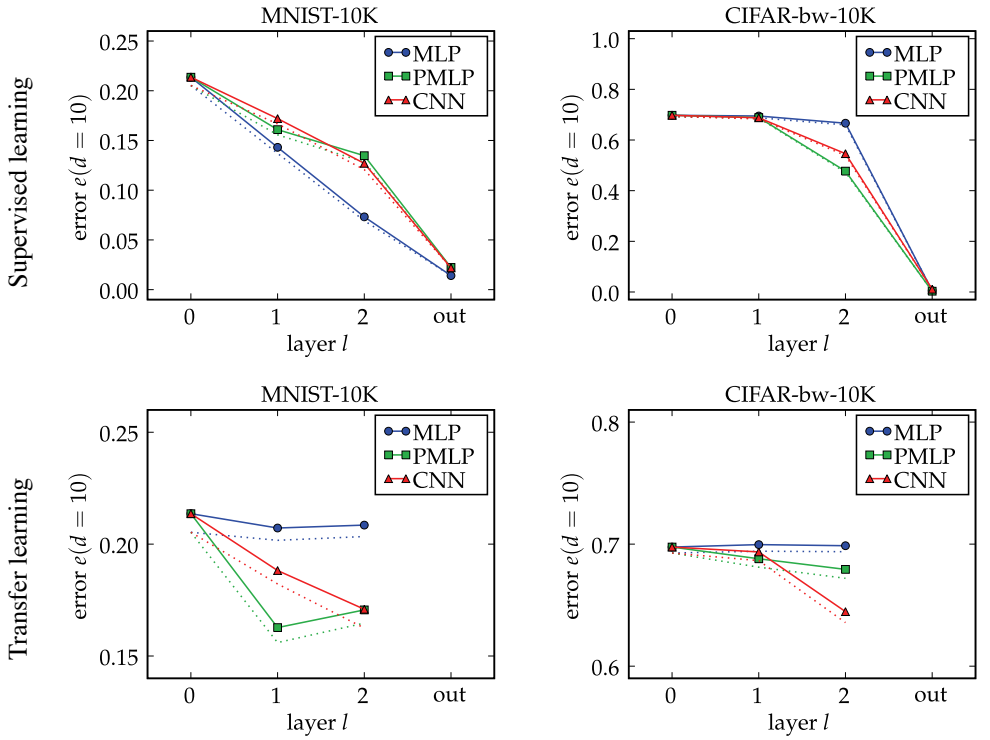


Figure 7: Layer-wise evolution of the error obtained for each training procedure for $d = 10$. Solid and dotted lines respectively represent the upper and lower approximation bounds of the analysis. We observe that the particular structure of the CNN and of the PMLP controls the layer-wise evolution of the representation. This confirms the second hypothesis depicted in Figure 1.

task specific distributions at its discriminative edges. Reformulating this hypothesis in the case of a simple multilayer feedforward network trained on image classification, the sensed distribution would evolve progressively from a distribution representing pixels well to a distribution representing classes well as the distribution is mapped to more and more layers.

We can observe in Figure 6 that this hypothesis holds within the span of our experimental setup and that simultaneously lower-dimensional and more accurate models of the task can be obtained layer after layer. This means that the task-relevant information, initially spread over a large number of principal components, converges progressively towards the leading components of the mapped data distribution.

This layer-wise preservation of the statistical tractability of the learning problem and its progressive simplification is a theoretical motivation for using these deep networks in a modular way (Caruana, 1997; Weston et al., 2008; Collobert and Weston, 2008): additional modules can be plugged on top of intermediate representations and still make sense of it.

4.2 Role of the Structure of Deep Networks

Training deep networks is a complex nonconvex learning problem with many reasonable solutions. As it can be seen in Figure 5, even simple hyperparameters such as the learning rate or the L2 weight penalty can greatly influence the layer-wise structure of the solution. Adding to the fact that those are only a fraction of the hyperparameters that needs to be tuned in order to achieve high generalization (e.g., importance of reconstruction error, orthogonality of hidden representations), it can therefore be tricky—if not, impossible—to find an appropriate combination of hyperparameters that leads to a well-structured solution for the learning problem.

On the other hand, the unsupervised pretraining proposed by Hinton et al. (2006) finds a network of latent variables that better represents the underlying distribution. As a consequence, the structure of the pretrained deep network already contains a certain part of the solution (Larochelle et al., 2009) and possibly makes better use of each layer. Similarly, in the context of sequential data, we can postulate that dedicating the early layers of the architecture to a convolutional preprocessing is also a more effective (LeCun, 1989; Serre et al., 2005) and biologically plausible (Ringach, 2002) way of solving the learning problem. Both approaches have shown empirically to produce better generalization (LeCun, 1989; Salakhutdinov and Hinton, 2007).

We corroborate this argument by comparing in Figure 7 the layer-wise evolution of the representation for different deep networks: a multilayer perceptron (MLP), a pretrained MLP (PMLP) and a convolutional neural network (CNN). On one side, the MLP does not embed any preconditioning on the learning problem. On the other side, the PMLP and the CNN embed respectively a generative model of the input and a spatial invariance prior on the problem. We can think of the mechanisms implemented by the PMLP and the CNN as complex regularizers on the solution of the learning problem.

Figure 7 (top) shows the evolution of the representation with respect to the learning problem when the deep network has been trained on the target task. We observe that the evolution of the representation of the MLP follows a different trend than the representation built by the PMLP and the CNN. The MLP tends to solve the MNIST problem greedily, discriminating from the first layers while the PMLP and the CNN postpone the discrimination to the last layers. On the other hand, on the CIFAR data set, the MLP doesn't discriminate until the last layer while the PMLP and the CNN spread the discrimination to more layers. Figure 7 (bottom) shows the evolution of the

representation with respect to the learning problem when the deep network has been trained on the transfer task. On both data sets, the representation built by the MLP does not improve as the deep network specializes on the transfer task while the PMLP and the CNN still build in the first layers a better representation of the learning problem, corroborating the effect of the PMLP and the CNN on structure of the solution.

These observations suggest that the complex regularizers implemented in the PMLP and the CNN have the effect of facilitating the construction of a structured solution, controlling the rate of discrimination at every layer. Erhan et al. (2010) already described the PMLP as a regularized version of the MLP and showed how it improves the generalization ability of deep networks. Our analysis completes the study, providing a layer-wise perspective on the effect and the role of regularization in deep networks and a unified view on the very different regularizers implemented by the PMLP and the CNN.

5. Conclusion and Discussion

We introduce a method for analyzing deep networks that combines kernel methods and descriptive statistics in order to quantify the layer-wise evolution of the representation in deep networks. Our method abstracts deep networks as a sequence of deeper and deeper kernels subsuming the mapping performed by more and more layers. The kernel framework expresses the relation between the representation built in the deep network and the learning problem.

Our analysis is able to detect and quantify the progressive and layer-wise transformation of the input performed by the deep network. In particular, we find that properly trained deep networks progressively simplify the statistics of complex data distributions, building in their last layers representations that are both simple and accurate.

The analysis also corroborates the hypothesis that a suitable structure for the deep network allows to make efficient use of its representational power by controlling the rate of discrimination at each layer of the deep network. This observation provides a new unified view on the role and effect of regularizers in deep networks.

Conceptually, our analysis is not only restricted to artificial neural networks. We believe that performing a similar analysis on different levels of processing in a biological neural architecture may reveal interesting parallels between artificial and biological neural systems.

Appendix A. More Background Information on kPCA Convergence

In this section, we briefly give some additional results on the convergence properties of kernel PCA. For the full account, please refer to Braun (2006) and Braun et al. (2008).

The rationale behind using the number of kPCA components as an estimate of the dimensionality rather than simpler metrics such as counting the number of support vectors of a trained SVM is that the first method provides an estimate of the dimensionality that is provably robust to the number of samples used in the analysis. This interesting fact was derived from a fundamental result on the approximation error of scalar products with eigenvectors of the kernel matrix with respect to their asymptotic counterparts.

More concretely, if $x_1, \ldots, x_n \in \mathcal{X}$ are points drawn i.i.d. from some probability distribution $P_{\mathcal{X}}$, we define the kernel matrix K of a Mercer kernel k by

$$K_{ij} = k(x_i, x_j) \qquad \text{for } 1 \leq i, j \leq n.$$

As $n \to \infty$, the eigenvalues and eigenvectors of K converge to those of the integral operator

$$T_k(f) = \int_{x \in \mathcal{X}} k(\,.\,,x) f(x) dP_{\mathcal{X}}$$

in an appropriate measure. In particular, it has been shown by Braun (2006) that the approximation error between the ith eigenvalue λ_i of K (in descending order) and corresponding eigenvalue l_i of T_k scales essentially multiplicatively, that is,

$$|\lambda_i - l_i| \leq C(n) l_i + \varepsilon(n),$$

where $C(n) \to 0$, $\varepsilon(n) \to 0$ as $n \to \infty$, and even for small n, $\varepsilon(n)$ is small.

So, even for a small number of points, the structure of the principal components in feature space are very similar compared to the asymptotic case. Moreover, as the next result shows, the same also holds for the location of the sample vector of a function with respect to the eigenspaces.

As shown by Braun et al. (2008), for a bounded function g which lies in the range of T_k (that is, there exists a h such that $T_k(h) = g$), one can bound the scalar products between the sample vector $G = (g(x_1), \ldots, g(x_n))$ and the eigenvectors u_i (normalized to unit length) of K by

$$\frac{1}{\sqrt{n}} |u_i^\top G| \leq \lambda_i C(n) + \varepsilon(n),$$

where $\varepsilon(n) \to 0$ as $n \to \infty$. Note that the scalar products with the eigenfunctions ψ_i of T_k also decay as $O(l_i)$, which are again linked to λ_i by the results discussed above.

In essence, this result shows that the scalar products between a subsampled smooth function decays as quickly as the eigenvalues of the kernel matrix, such that the information about g is contained in the leading kPCA components only. Here, smoothness means that g lies in the range of T_k such that it is a smoothed version of some function h after convolution with the kernel function k. On the other hand, any noise which is independent of the x_i is uniformly distributed over all kPCA components. In summary, if one plots the products $u_i^\top Y$ with the label vector $Y = (y_1, \ldots, y_n)$, one obtains a decomposition of the label information Y with respect to the kPCA components. From the above considerations, it follows that the spectrum will typically consist of a flat "noise bed" from which the relevant information in the leading components can clearly be distinguished. This result is illustrated in Figure 8 for a small toy example.

References

Gaston Baudat and Fatiha Anouar. Generalized discriminant analysis using a kernel approach. *Neural Computation*, 12(10):2385–2404, 2000.

Yoshua Bengio. Learning deep architectures for AI. *Foundations and Trends in Machine Learning*, 2(1):1–127, 2009.

Yoshua Bengio, Pascal Lamblin, Dan Popovici, and Hugo Larochelle. Greedy layer-wise training of deep networks. In *Advances in Neural Information Processing Systems 19*, pages 153–160, 2006.

Christopher M. Bishop. *Neural Networks for Pattern Recognition*. Oxford University Press, 1996.

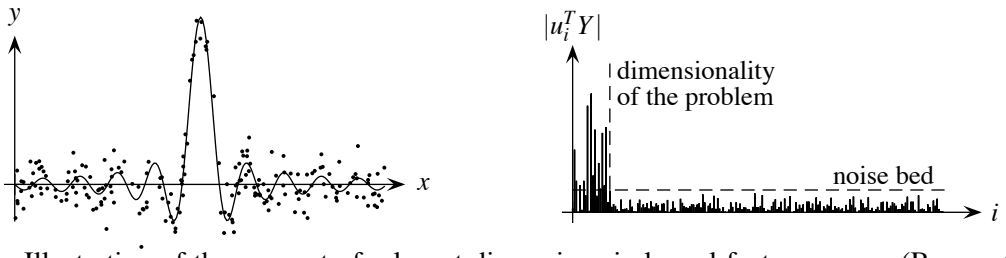


Figure 8: Illustration of the concept of relevant dimensions in kernel feature spaces (Braun et al., 2008). On the left, samples drawn from a toy distribution $p(x,y)$. On the right, label contributions of each kPCA component $u_1,\ldots,u_n$. It can be observed that a small number of leading principal components containing relevant label information emerge from a flat noise bed.

Léon Bottou. Stochastic gradient learning in neural networks. In *Proceedings of Neuro-Nîmes*, 1991.

Olivier Bousquet, Stéphane Boucheron, and Gábor Lugosi. Introduction to statistical learning theory. In Olivier Bousquet, Ulrike von Luxburg, and Gunnar Rätsch, editors, *Advanced Lectures on Machine Learning*, volume 3176, pages 169–207. Springer, 2004.

Mikio L. Braun. Accurate bounds for the eigenvalues of the kernel matrix. *Journal of Machine Learning Research*, 7:2303–2328, 2006.

Mikio L. Braun, Joachim Buhmann, and Klaus-Robert Müller. On relevant dimensions in kernel feature spaces. *Journal of Machine Learning Research*, 9:1875–1908, 2008.

Rich Caruana. Multitask learning. *Machine Learning*, 28(1):41–75, 1997.

Youngmin Cho and Lawrence Saul. Kernel methods for deep learning. In *Advances in Neural Information Processing Systems 22*, pages 342–350, 2009.

Ronan Collobert and Jason Weston. A unified architecture for natural language processing: deep neural networks with multitask learning. In *Proceedings of the 25th International Conference on Machine Learning*, pages 160–167, 2008.

Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine Learning*, 20(3):273–297, 1995.

Dumitru Erhan, Yoshua Bengio, Aaron Courville, Pierre-Antoine Manzagol, Pascal Vincent, and Samy Bengio. Why does unsupervised pre-training help deep learning? *Journal of Machine Learning Research*, 11:625–660, 2010.

Ian Goodfellow, Quoc Le, Andrew Saxe, and Andrew Y. Ng. Measuring invariances in deep networks. In *Advances in Neural Information Processing Systems 22*, pages 646–654, 2009.

Geoffrey E. Hinton, Simon Osindero, and Yee-Whye Teh. A fast learning algorithm for deep belief nets. *Neural Computation*, 18(7):1527–1554, 2006.

David H. Hubel and Torsten N. Wiesel. Receptive fields, binocular interaction and functional architecture in the cat's visual cortex. *The Journal of Physiology*, 160:106–154, January 1962.

Alex Krizhevsky. Learning multiple layers of features from tiny images. Technical report, University of Toronto, 2009.

Kevin J. Lang, Alex H. Waibel, and Geoffrey E. Hinton. A time-delay neural network architecture for isolated word recognition. *Neural Networks*, 3(1):23–43, 1990.

Hugo Larochelle, Yoshua Bengio, Jérôme Louradour, and Pascal Lamblin. Exploring strategies for training deep neural networks. *Journal of Machine Learning Research*, 10:1–40, 2009.

Yann LeCun. Generalization and network design strategies. In *Connectionism in Perspective*. Elsevier, 1989. An extended version was published as a technical report of the University of Toronto.

Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(1):2278–2324, 1998.

Sebastian Mika, Gunnar Rätsch, Jason Weston, Bernhard Schölkopf, and Klaus-Robert Müller. Fisher discriminant analysis with kernels. In *Neural Networks for Signal Processing IX, 1999. Proceedings of the 1999 IEEE Signal Processing Society Workshop*, pages 41–48, 1999.

Sebastian Mika, Gunnar Rätsch, Jason Weston, Bernhard Schölkopf, Alex J. Smola, and Klaus-Robert Müller. Learning discriminative and invariant nonlinear features. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(5):623–628, 2003.

Grégoire Montavon, Mikio Braun, and Klaus-Robert Müller. Layer-wise analysis of deep networks with Gaussian kernels. In *Advances in Neural Information Processing Systems 23*, pages 1678–1686, 2010.

Klaus-Robert Müller, Sebastian Mika, Gunnar Rätsch, Koji Tsuda, and Bernhard Schölkopf. An introduction to kernel-based learning algorithms. *IEEE Transactions on Neural Networks*, 12(2): 181–202, 2001.

Vinod Nair and Geoffrey E. Hinton. Rectified linear units improve restricted Boltzmann machines. In *Proceedings of the 27th International Conference on Machine Learning*, pages 807–814, 2010.

Genevieve B. Orr and Klaus-Robert Müller, editors. *Neural Networks: Tricks of the Trade*, volume 1524 of *Lecture Notes in Computer Science*, 1998. Springer. This book is an outgrowth of a 1996 NIPS workshop.

Dario L. Ringach. Spatial structure and symmetry of simple-cell receptive fields in macaque primary visual cortex. *The Journal of Neurophysiology*, 88(1):455–463, 2002.

David E. Rumelhart, Geoffrey E. Hinton, and Ronald J. Williams. Learning representations by back-propagating errors. *Nature*, 323(6088):533–536, 1986.

Ruslan Salakhutdinov and Geoffrey Hinton. Learning a nonlinear embedding by preserving class neighbourhood structure. In *Proceedings of the International Conference on Artificial Intelligence and Statistics*, volume 11, 2007.

Bernhard Schölkopf and Alexander J. Smola. *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. MIT Press, 2002.

Bernhard Schölkopf, Alexander Smola, and Klaus-Robert Müller. Nonlinear component analysis as a kernel eigenvalue problem. *Neural Computation*, 10(5):1299–1319, 1998.

Bernhard Schölkopf, Sebastian Mika, Chris J. C. Burges, Philipp Knirsch, Klaus-Robert Müller, Gunnar Rätsch, and Alex J. Smola. Input space versus feature space in kernel-based methods. *IEEE Transactions on Neural Networks*, 10(5):1000–1017, 1999.

Thomas Serre, Lior Wolf, and Tomaso Poggio. Object recognition with features inspired by visual cortex. In *Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 2, pages 994–1000, 2005.

Steve Smale, Lorenzo Rosasco, Jack Bouvrie, Andrea Caponnetto, and Tomaso Poggio. Mathematics of the neural response. *Foundations of Computational Mathematics*, 10(1):67–91, 2010.

Alex J. Smola, Bernhard Schölkopf, and Klaus-Robert Müller. The connection between regularization operators and support vector kernels. *Neural Networks*, 11(4):637–649, 1998.

Jason Weston, Frédéric Ratle, and Ronan Collobert. Deep learning via semi-supervised embedding. In *Proceedings of the 25th International Conference on Machine Learning*, pages 1168–1175, 2008.

Andre Wibisono, Jake Bouvrie, Lorenzo Rosasco, and Tomaso Poggio. Learning and invariance in a family of hierarchical kernels. Technical report, Massachusetts Institute of Technology, 2010.

Journal of Machine Learning Research 12 (2011) 2583-2648 Submitted 5/10; Revised 2/11; Published 9/11

Theoretical Analysis of Bayesian Matrix Factorization*

Shinichi Nakajima NAKAJIMA.S@NIKON.CO.JP
Optical Research Laboratory
Nikon Corporation
Tokyo 140-8601, Japan

Masashi Sugiyama SUGI@CS.TITECH.AC.JP
Department of Computer Science
Tokyo Institute of Technology
Tokyo 152-8552, Japan

Editor: Inderjit Dhillon

Abstract

Recently, *variational Bayesian* (VB) techniques have been applied to probabilistic matrix factorization and shown to perform very well in experiments. In this paper, we theoretically elucidate properties of the VB matrix factorization (VBMF) method. Through finite-sample analysis of the VBMF estimator, we show that two types of shrinkage factors exist in the VBMF estimator: the *positive-part James-Stein (PJS)* shrinkage and the *trace-norm* shrinkage, both acting on each singular component separately for producing low-rank solutions. The trace-norm shrinkage is simply induced by non-flat prior information, similarly to the maximum a posteriori (MAP) approach. Thus, no trace-norm shrinkage remains when priors are non-informative. On the other hand, we show a counter-intuitive fact that the PJS shrinkage factor is kept activated even with flat priors. This is shown to be induced by the *non-identifiability* of the matrix factorization model, that is, the mapping between the target matrix and factorized matrices is not one-to-one. We call this *model-induced regularization*. We further extend our analysis to empirical Bayes scenarios where hyperparameters are also learned based on the VB free energy. Throughout the paper, we assume no missing entry in the observed matrix, and therefore collaborative filtering is out of scope.

Keywords: matrix factorization, variational Bayes, empirical Bayes, positive-part James-Stein shrinkage, non-identifiable model, model-induced regularization

1. Introduction

The goal of *matrix factorization* (MF) is to find a low-rank expression of a target matrix. MF can be used for learning linear relation between vectors such as *reduced rank regression* (Baldi and Hornik, 1995; Reinsel and Velu, 1998), *canonical correlation analysis* (Hotelling, 1936; Anderson, 1984), *partial least-squares* (Wold, 1966; Worsley et al., 1997; Rosipal and Krämer, 2006), and *multi-task learning* (Chapelle and Harchaoui, 2005; Yu et al., 2005). More recently, MF is applied to *collaborative filtering* for imputing missing entries of a target matrix, for example, in the context of *recommender systems* (Konstan et al., 1997; Funk, 2006) and *microarray data analysis* (Baldi and Brunak, 1998). For these reasons, MF has attracted considerable attention these days.

*. This paper is an extended version of our earlier conference paper (Nakajima and Sugiyama, 2010).

1.1 MF Methods

Srebro and Jaakkola (2003) proposed the *weighted low-rank approximation* method, which is based on the *expectation-maximization* (EM) algorithm: a matrix is fitted to the data without a rank constraint in the E-step and it is projected back to the set of low-rank matrices by *singular value decomposition* (SVD) in the M-step. Since the optimization problem of the weighted low-rank approximation method involves a low-rank constraint, it is non-convex and thus only a local optimal solution may be obtained. Furthermore, SVD of the target matrix needs to be carried out in each iteration, which may be computationally intractable for large-scale data.

Funk (2006) proposed the *regularized SVD* method that minimizes a goodness-of-fit term combined with the *Frobenius-norm* penalty under a low-rank constraint by gradient descent (see also Paterek, 2007). The regularized SVD method could be computationally more efficient than the weighted low-rank approximation method in the context of collaborative filtering since only observed entries are referred to in each gradient iteration.

Srebro et al. (2005) proposed to use the *trace-norm* penalty instead of the Frobenius-norm penalty, so that a low-rank solution can be obtained without having an explicit low-rank constraint. Thanks to the convexity of the *trace-norm*, a semi-definite programming formulation can be obtained when the *hinge-loss* (Schölkopf and Smola, 2002) is used. See also Rennie and Srebro (2005) for a computationally efficient variant using a gradient-based optimization method with smooth approximation.

Salakhutdinov and Mnih (2008) proposed a Bayesian *maximum a posteriori* (MAP) method based on the Gaussian noise model and Gaussian priors on the decomposed matrices. This method actually corresponds to minimizing the squared-loss with the trace-norm penalty (Srebro et al., 2005).

Recently, the *variational Bayesian* (VB) approach (Attias, 1999) has been applied to MF (Lim and Teh, 2007; Raiko et al., 2007), which we refer to as *VBMF*. The VBMF method was shown to perform very well in experiments. However, its good performance was not completely understood beyond its experimental success. The purpose of this paper is to provide new insight into Bayesian MF.

1.2 MF Models and Non-identifiability

The MF models can be regarded as re-parameterization of the target matrix using low-rank matrices. This kind of re-parameterization often significantly changes the statistical behavior of the estimator (Gelman, 2004). Indeed, MF models possess a special structure called *non-identifiability* (Watanabe, 2009), meaning that the mapping between the target matrix and the factorized matrices is not one-to-one .

Previous theoretical studies on non-identifiable models investigated the behavior of *multi-layer pereptrons*, *Gaussian mixture models*, and *hidden Markov models*. It was shown that when such non-identifiable models are trained using *full-Baysian* (FB) estimation, the regularization effect is significantly stronger than the MAP method (Watanabe, 2001; Yamazaki and Watanabe, 2003). Since a single point in the function space corresponds to a set of points in the (redundant) parameter space in non-identifiable models, simple distributions such as the Gaussian distribution in the function space produce highly complicated *multimodal* distributions in the parameter space. This causes the MAP and FB solutions to be significantly different. Thus the behavior of non-identifiable models is substantially different from that of identifiable models. For Gaussian mixture models and

reduced rank regression models, theoretical properties of VB have also been investigated (Watanabe and Watanabe, 2006; Nakajima and Watanabe, 2007).

1.3 Our Contribution

In this paper, following the line of Nakajima and Watanabe (2007) which investigated asymptotic behavior of VBMF estimators and the generalization error, we provide a more precise analysis of VB estimators. More specifically, we derive *non-asymptotic* bounds of the VBMF estimator. The obtained solution can be seen as a re-weighted singular value decomposition, and the weights include a factor induced by the *Bayesian* inference procedure, in the same way as *automatic relevance determination* (Neal, 1996; Wipf and Nagarajan, 2008).

We show that VBMF consists of two shrinkage factors, the *positive-part James-Stein* (PJS) shrinkage (James and Stein, 1961; Efron and Morris, 1973) and the *trace-norm* shrinkage (Srebro et al., 2005), operating on each singular component separately for producing low-rank solutions.

The trace-norm shrinkage is simply induced by non-flat prior information, as in the MAP approach (Salakhutdinov and Mnih, 2008). Thus, no trace-norm shrinkage remains when priors are non-informative. On the other hand, we show a counter-intuitive fact that the PJS shrinkage factor is still kept activated even with uniform priors. This allows the VBMF method to avoid overfitting (or in some cases, this may cause underfitting) even when non-informative priors are provided. We call this regularization effect *model-induced regularization* since it is caused by the structure of the model likelihood function.

We further extend the above analysis to *empirical VBMF* (EVBMF) scenarios, where hyperparameters in prior distributions are also learned based on the *VB free energy*. We derive bounds of the EVBMF estimator, and show that the effect of PJS shrinkage is at least doubled compared with the uniform prior cases.

Finally, we note that our analysis relies on the following three assumptions: First, we assume that the given matrix is *fully* observed, and no missing entry exists. This means that missing entry prediction is out of scope of our theory. Second, we require the noise to be independent Gaussian noise and the priors to be isotropic Gaussian. Third, we assume the column-wise independence on the VB posterior, which is different from the standard VB assumption that only the matrix-wise independence is required.

1.4 Organization

The rest of this paper is organized as follows. In Section 2, we formulate the MF problem and review its Bayesian approaches including FB, MAP, VB methods, and their empirical variants. In Section 3, we analyze the behavior of MAPMF, VBMF, and their empirical variants, and elucidate the regularization mechanism. In Section 4, we illustrate the characteristic behavior of MF solutions through simple numerical experiments, highlighting the influence of non-identifiability of the MF models. Finally, we conclude in Section 5. A brief review of the James-Stein shrinkage estimator and all the technical details are provided in Appendix.

2. Bayesian Approaches to Matrix Factorization

In this section, we give a probabilistic formulation of the *matrix factorization* (MF) problem and review its Bayesian methods.

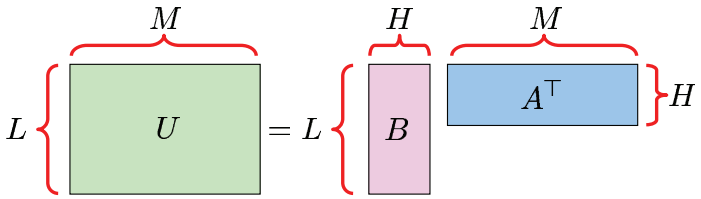


Figure 1: Matrix factorization model.

2.1 Formulation

The goal of the MF problem is to estimate a target matrix U ($\in \mathbb{R}^{L\times M}$) from its observation

$$V \in \mathbb{R}^{L\times M}.$$

Throughout the paper, we assume that

$$L \leq M.$$

If $L > M$, we may simply re-define the transpose $U^\top$ as U so that $L \leq M$ holds. Thus this does not impose any restriction.

A key assumption of MF is that U is a low-rank matrix. Let H ($\leq L$) be the rank of U. Then the matrix U can be decomposed into the product of $A \in \mathbb{R}^{M\times H}$ and $B \in \mathbb{R}^{L\times H}$ as follows (see Figure 1):

$$U = BA^\top.$$

With appropriate *pre-whitening* (Hyvärinen et al., 2001), *reduced rank regression* (Baldi and Hornik, 1995; Reinsel and Velu, 1998), *canonical correlation analysis* (Hotelling, 1936; Anderson, 1984), *partial least-squares* (Wold, 1966; Worsley et al., 1997; Rosipal and Krämer, 2006), and *multi-task learning* (Chapelle and Harchaoui, 2005; Yu et al., 2005) can be seen as special cases of the MF problem. *Collaborative filtering* (Konstan et al., 1997; Baldi and Brunak, 1998; Funk, 2006) and *image processing* (Lee and Seung, 1999) would be popular applications of MF. Note that, some of these applications such as *collaborative filtering* and *multi-task learning* with unshared input sets are out of scope of our theory, since they require missing entry prediction.

Assume that the observed matrix V is subject to the following additive-noise model:

$$V = U + \mathcal{E},$$

where $\mathcal{E}$ ($\in \mathbb{R}^{L\times M}$) is a noise matrix. Each entry of $\mathcal{E}$ is assumed to independently follow the Gaussian distribution with mean zero and variance σ^2. Then, the likelihood $p(V|A,B)$ is given by

$$p(V|A,B) \propto \exp\left(-\frac{1}{2\sigma^2}\|V - BA^\top\|_{\mathrm{Fro}}^2\right), \tag{1}$$

where $\|\cdot\|_{\mathrm{Fro}}$ denotes the *Frobenius norm* of a matrix.

2.2 Full-Bayesian Matrix Factorization (FBMF) and Its Empirical Variant (EFBMF)

We use the Gaussian priors on the parameters A and B:

$$\phi(U) = \phi_{\mathrm{A}}(A)\phi_{\mathrm{B}}(B),$$

where

$$\phi_{\mathrm{A}}(A) \propto \exp\left(-\sum_{h=1}^{H} \frac{\|\boldsymbol{a}_h\|^2}{2c_{a_h}^2}\right) = \exp\left(-\frac{\mathrm{tr}(AC_A^{-1}A^\top)}{2}\right), \tag{2}$$

$$\phi_{\mathrm{B}}(B) \propto \exp\left(-\sum_{h=1}^{H} \frac{\|\boldsymbol{b}_h\|^2}{2c_{b_h}^2}\right) = \exp\left(-\frac{\mathrm{tr}(BC_B^{-1}B^\top)}{2}\right). \tag{3}$$

Here, $\boldsymbol{a}_h$ and $\boldsymbol{b}_h$ are the h-th column vectors of A and B, respectively, that is,

$$A = (\boldsymbol{a}_1, \ldots, \boldsymbol{a}_H),$$
$$B = (\boldsymbol{b}_1, \ldots, \boldsymbol{b}_H).$$

$c_{a_h}^2$ and $c_{b_h}^2$ are hyperparameters corresponding to the prior variances of those vectors. Without loss of generality, we assume that the product $c_{a_h}c_{b_h}$ is non-increasing with respect to h. We also denote them as covariance matrices:

$$C_A = \mathrm{diag}(c_{a_1}^2, \ldots, c_{a_H}^2),$$
$$C_B = \mathrm{diag}(c_{b_1}^2, \ldots, c_{b_H}^2),$$

where $\mathrm{diag}(\boldsymbol{c})$ denotes the diagonal matrix with its entries specified by vector $\boldsymbol{c}$. $\mathrm{tr}(\cdot)$ denotes the trace of a matrix.

With the Bayes theorem and the definition of marginal distributions, the *Bayes posterior* $p(A,B|V)$ can be written as

$$p(A,B|V) = \frac{p(A,B,V)}{p(V)} = \frac{p(V|A,B)\phi_{\mathrm{A}}(A)\phi_{\mathrm{B}}(B)}{\langle p(V|A,B)\rangle_{\phi_{\mathrm{A}}(A)\phi_{\mathrm{B}}(B)}}, \tag{4}$$

where $\langle\cdot\rangle_p$ denotes the expectation over p. The *full-Bayesian* (FB) solution is given by the *Bayes posterior mean*:

$$\widehat{U}^{\mathrm{FB}} = \langle BA^\top\rangle_{p(A,B|V)}. \tag{5}$$

We call this method *FBMF*.

The hyperparameters c_{a_h} and c_{b_h} may be determined so that the *Bayes free energy* $F(V)$ is minimized.

$$\begin{aligned} F(V) &= -\log p(V) \\ &= -\log\langle p(V|A,B)\rangle_{\phi_{\mathrm{A}}(A)\phi_{\mathrm{B}}(B)}. \end{aligned} \tag{6}$$

We call this method the *empirical full-Bayesian MF* (EFBMF). The Bayes free energy is also referred to as the *marginal log-likelihood* (MacKay, 2003), the *evidence* (MacKay, 1992) or the *stochastic complexity* (Rissanen, 1986).

2.3 Maximum A Posteriori Matrix Factorization (MAPMF) and Its Empirical Variant (EMAPMF)

When computing the Bayes posterior (4), the expectation in the denominator of Equation (4) is often intractable due to high dimensionality of the parameters A and B. More importantly, computing the posterior mean (5) is also intractable. A simple approach to mitigating this problem is to use the *maximum a posteriori* (MAP) approximation, which we refer to as MAPMF. The MAP solution $\widehat{U}^{\mathrm{MAP}}$ is given by

$$\widehat{U}^{\mathrm{MAP}} = \widehat{B}^{\mathrm{MAP}}(\widehat{A}^{\mathrm{MAP}})^{\top},$$

where

$$(\widehat{A}^{\mathrm{MAP}}, \widehat{B}^{\mathrm{MAP}}) = \operatorname*{argmax}_{A,B} p(A,B|V).$$

In the MAP framework, one may determine the hyperparameters c_{a_h} and c_{b_h} so that the Bayes posterior $p(A,B|V)$ is maximized (equivalently, the negative log posterior is minimized). We call this method *empirical MAPMF* (EMAPMF). Note that EMAPMF does not work properly, as explained in Section 3.3.

2.4 Variational Bayesian Matrix Factorization (VBMF) and Its Empirical Variant (EVBMF)

Another approach to avoiding computational intractability of the FB method is to use the *variational Bayes* (VB) approximation (Attias, 1999; Bishop, 2006). Here, we review the VB-based MF method (Lim and Teh, 2007; Raiko et al., 2007).

Let $r(A,B|V)$ be a *trial* distribution for A and B, and we define the following functional F_{VB} called the *VB free energy* with respect to $r(A,B|V)$:

$$F_{\mathrm{VB}}(r|V) = \left\langle \log \frac{r(A,B|V)}{p(V,A,B)} \right\rangle_{r(A,B|V)}. \tag{7}$$

Using $p(V,A,B) = p(A,B|V)p(V)$, we can decompose Equation (7) into two terms:

$$F_{\mathrm{VB}}(r|V) = \left\langle \log \frac{r(A,B|V)}{p(A,B|V)} \right\rangle_{r(A,B|V)} + F(V), \tag{8}$$

where $F(V)$ is the Bayes free energy defined by Equation (6). The first term in Equation (8) is the *Kullback-Leibler divergence* (Kullback and Leibler, 1951) from $r(A,B|V)$ to the Bayes posterior $p(A,B|V)$. This is non-negative and vanishes if and only if the two distributions agree with each other. Therefore, the VB free energy $F_{\mathrm{VB}}(r|V)$ is lower-bounded by the Bayes free energy $F(V)$:

$$F_{\mathrm{VB}}(r|V) \geq F(V),$$

where the equality is satisfied if and only if $r(A,B|V)$ agrees with $p(A,B|V)$.

The VB approach minimizes the VB free energy $F_{\mathrm{VB}}(r|V)$ with respect to the trial distribution $r(A,B|V)$, by restricting the search space of $r(A,B|V)$ so that the minimization is computationally tractable. Typically, dissolution of probabilistic dependency between entangled parameters (A and B in the case of MF) makes the calculation feasible:

$$r(A,B|V) = r_{\mathrm{A}}(A|V)r_{\mathrm{B}}(B|V). \tag{9}$$

Then, the VB free energy (7) is written as

$$F_{\mathrm{VB}}(r|V) = \left\langle \log \frac{r_{\mathrm{A}}(A|V) r_{\mathrm{B}}(B|V)}{p(V|A,B)\phi_{\mathrm{A}}(A)\phi_{\mathrm{B}}(B)} \right\rangle_{r_{\mathrm{A}}(A|V) r_{\mathrm{B}}(B|V)}. \tag{10}$$

The resulting distribution is called the *VB posterior*. The VB solution $\widehat{U}^{\mathrm{VB}}$ is given by the *VB posterior mean*:

$$\widehat{U}^{\mathrm{VB}} = \langle BA^{\top} \rangle_{r(A,B|V)}. \tag{11}$$

We call this method *VBMF*.

Applying the variational method to the VB free energy shows that the VB posterior satisfies the following conditions:

$$r_{\mathrm{A}}(A|V) \propto \phi_{\mathrm{A}}(A) \exp\left(\langle \log p(V|A,B) \rangle_{r_{\mathrm{B}}(B|V)} \right), \tag{12}$$

$$r_{\mathrm{B}}(B|V) \propto \phi_{\mathrm{B}}(B) \exp\left(\langle \log p(V|A,B) \rangle_{r_{\mathrm{A}}(A|V)} \right). \tag{13}$$

Recall that we are using the Gaussian priors (2) and (3). Also, Equation (1) implies that the log-likelihood $\log p(V|A,B)$ is a quadratic function of A when B is fixed, and vice versa. Then the conditions (12) and (13) imply that the VB posteriors $r_{\mathrm{A}}(A|V)$ and $r_{\mathrm{B}}(B|V)$ are also Gaussian. This enables one to derive a computationally efficient algorithm called the *iterated conditional modes* (Besag, 1986; Bishop, 2006), where the mean and the covariance of the parameters A and B are iteratively updated using Equations (12) and (13) (Lim and Teh, 2007; Raiko et al., 2007). This amounts to alternating between minimizing the free energy (10) with respect to $r_{\mathrm{A}}(A|V)$ and $r_{\mathrm{B}}(B|V)$.

As in Raiko et al. (2007), we assume in our theoretical analysis that the trial distribution $r(A,B|V)$ can be further factorized as

$$r(A,B|V) = \prod_{h=1}^{H} r_{\mathrm{a}_h}(\boldsymbol{a}_h|V) r_{\mathrm{b}_h}(\boldsymbol{b}_h|V). \tag{14}$$

Then the update rules (12) and (13) are simplified as

$$r_{\mathrm{a}_h}(\boldsymbol{a}_h|V) \propto \phi_{\mathrm{a}_h}(\boldsymbol{a}_h) \exp\left(\langle \log p(V|A,B) \rangle_{r_{\backslash \mathrm{a}_h}(A \backslash a_h, B|V)} \right), \tag{15}$$

$$r_{\mathrm{b}_h}(\boldsymbol{b}_h|V) \propto \phi_{\mathrm{b}_h}(\boldsymbol{b}_h) \exp\left(\langle \log p(V|A,B) \rangle_{r_{\backslash \mathrm{b}_h}(A, B \backslash b_h|V)} \right), \tag{16}$$

where $r_{\backslash \mathrm{a}_h}$ and $r_{\backslash \mathrm{b}_h}$ denote the VB posterior of the parameters A and B except $\boldsymbol{a}_h$ and $\boldsymbol{b}_h$, respectively.

The VB free energy also allows us to determine the hyperparameters $c_{a_h}^2$ and $c_{b_h}^2$ in a computationally tractable way. That is, instead of the Bayes free energy $F(V)$, the VB free energy $F_{\mathrm{VB}}(r|V)$ is minimized with respect to $c_{a_h}^2$ and $c_{b_h}^2$. We call this method *empirical VBMF* (EVBMF).

3. Analysis of Bayesian MF Methods

In this section, we theoretically analyze the behavior of MAPMF, VBMF, EMAPMF, and EVBMF solutions, and elucidate their regularization mechanism.

3.1 MAPMF

The MAP estimator $(\widehat{A}^{\mathrm{MAP}}, \widehat{B}^{\mathrm{MAP}})$ is the maximizer of the Bayes posterior. In our model (1), (2), and (3), the negative log of the Bayes posterior is expressed as

$$-\log p(A,B|V) = \frac{LM\log\sigma^2}{2} + \frac{1}{2}\sum_{h=1}^{H}\left(M\log c_{a_h}^2 + L\log c_{b_h}^2 + \frac{\|\boldsymbol{a}_h\|^2}{c_{a_h}^2} + \frac{\|\boldsymbol{b}_h\|^2}{c_{b_h}^2}\right) + \frac{1}{2\sigma^2}\left\|V - \sum_{h=1}^{H}\boldsymbol{b}_h\boldsymbol{a}_h^\top\right\|_{\mathrm{Fro}}^2 + \mathrm{Const.} \tag{17}$$

Differentiating Equation (17) with respect to A and B and setting the derivatives to zero, we have the following conditions:

$$\boldsymbol{a}_h = \left(\|\boldsymbol{b}_h\|^2 + \frac{\sigma^2}{c_{a_h}^2}\right)^{-1}\left(V - \sum_{h'\neq h}\boldsymbol{b}_{h'}\boldsymbol{a}_{h'}^\top\right)^\top\boldsymbol{b}_h, \tag{18}$$

$$\boldsymbol{b}_h = \left(\|\boldsymbol{a}_h\|^2 + \frac{\sigma^2}{c_{b_h}^2}\right)^{-1}\left(V - \sum_{h'\neq h}\boldsymbol{b}_{h'}\boldsymbol{a}_{h'}^\top\right)\boldsymbol{a}_h. \tag{19}$$

One may search a local solution (i.e., a local minimum of the negative log posterior (17)) by iterating Equations (18) and (19). However, as shown below, the optimal solution can be obtained analytically in the current setup.

When the hyperparameters are homogeneous, that is, $\{c_{a_h}c_{b_h} = c; {}^\forall h = 1,\ldots,H\}$, a closed-form expression of the MAP estimator can be immediately obtained by combining the results given in Srebro et al. (2005) and Cai et al. (2010). The following theorem is its slight extension that covers heterogeneous cases (its proof is given in Appendix B):

Theorem 1 *Let γ_h (≥ 0) be the h-th largest singular value of V. Let $\boldsymbol{\omega}_{a_h}$ and $\boldsymbol{\omega}_{b_h}$ be the associated right and left singular vectors:*

$$V = \sum_{h=1}^{L}\gamma_h\boldsymbol{\omega}_{b_h}\boldsymbol{\omega}_{a_h}^\top. \tag{20}$$

The MAP estimator $\widehat{U}^{\mathrm{MAP}}$ is given by

$$\widehat{U}^{\mathrm{MAP}} = \sum_{h=1}^{H}\widehat{\gamma}_h^{\mathrm{MAP}}\boldsymbol{\omega}_{b_h}\boldsymbol{\omega}_{a_h}^\top,$$

where

$$\widehat{\gamma}_h^{\mathrm{MAP}} = \max\left\{0, \gamma_h - \frac{\sigma^2}{c_{a_h}c_{b_h}}\right\}. \tag{21}$$

The theorem implies that the MAP solution cuts off the singular values less than $\sigma^2/(c_{a_h}c_{b_h})$; otherwise it reduces the singular values by $\sigma^2/(c_{a_h}c_{b_h})$ (see Figure 2). This shrinkage effect allows the MAPMF method to avoid overfitting.

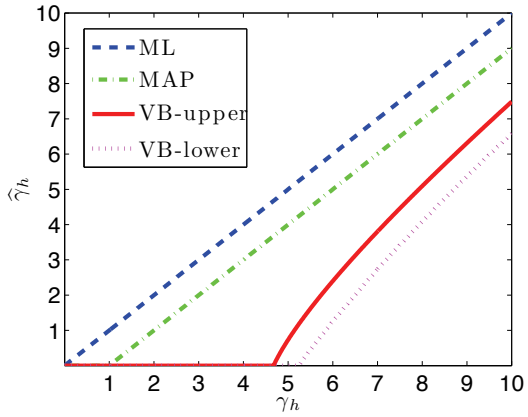


Figure 2: Shrinkage of the ML estimator (22), the MAP estimator (21), and the VB estimator (28) when $\sigma^2 = 0.1$, $c_{a_h} c_{b_h} = 0.1$, $L = 100$, and $M = 200$.

Similarly to Theorem 1, we can show that the *maximum likelihood* (ML) estimator is given by

$$\widehat{U}^{\mathrm{ML}} = \sum_{h=1}^{H} \widehat{\gamma}_h^{\mathrm{ML}} \boldsymbol{\omega}_{b_h} \boldsymbol{\omega}_{a_h}^{\top},$$

where

$$\widehat{\gamma}_h^{\mathrm{ML}} = \gamma_h \text{ for all } h. \tag{22}$$

Thus the ML solution is reduced to V when $H = L$ (see Figure 2):

$$\widehat{U}^{\mathrm{ML}} = \sum_{h=1}^{L} \widehat{\gamma}_h^{\mathrm{ML}} \boldsymbol{\omega}_{b_h} \boldsymbol{\omega}_{a_h}^{\top} = V.$$

A parametric model is said to be *identifiable* if the mapping between parameters and functions is one-to-one; otherwise the model is said to be *non-identifiable* (Watanabe, 2001). Since the decomposition $U = BA^{\top}$ is redundant, the MF model is non-identifiable (Nakajima and Watanabe, 2007). For identifiable models, the MAP estimator with the uniform prior is reduced to the ML estimator (Bishop, 2006). On the other hand, in the MF model, a single point in the space of U corresponds to a set of points in the joint space of A and B. For this reason, the uniform priors on A and B do not produce the uniform prior on U. Nevertheless, Equations (21) and (22) imply that MAP is reduced to ML when the priors on A and B are uniform (i.e., $c_{a_h}, c_{b_h} \to \infty$).

More precisely, Equations (21) and (22) show that the product $c_{a_h} c_{b_h} \to \infty$ is sufficient for MAP to be reduced to ML, which is weaker than both $c_{a_h}, c_{b_h} \to \infty$. This implies that both priors on A and B do not have to be uniform; only the condition that one of the priors is uniform is sufficient for MAP to be reduced to ML in the MF model. This phenomenon is distinctively different from the case of identifiable models.

If the prior is uniform and the likelihood is Gaussian, then the posterior is also Gaussian. Thus the mean and mode of the posterior agree with each other due to the symmetry of the Gaussian

density. For identifiable models, this fact implies that the FB and MAP solutions agree with each other. However, the FB and MAP solutions are generally different in non-identifiable models since the symmetry of the Gaussian density in the space of U is no longer kept in the joint space of A and B. In Section 4.1, we will further investigate these distinctive features of the MF model using illustrative examples.

3.2 VBMF

Substituting Equations (1), (2), and (3) into Equations (15) and (16), we find that the VB posteriors can be expressed as follows:

$$r_{\mathrm{A}}(A|V) = \prod_{h=1}^{H} \mathcal{N}_M(\boldsymbol{a}_h; \boldsymbol{\mu}_{a_h}, \Sigma_{a_h}),$$
$$r_{\mathrm{B}}(B|V) = \prod_{h=1}^{H} \mathcal{N}_L(\boldsymbol{b}_h; \boldsymbol{\mu}_{b_h}, \Sigma_{b_h}),$$

where $\mathcal{N}_d(\cdot; \boldsymbol{\mu}, \Sigma)$ denotes the d-dimensional Gaussian density with mean $\boldsymbol{\mu}$ and covariance matrix Σ. $\boldsymbol{\mu}_{a_h}$, $\boldsymbol{\mu}_{b_h}$, Σ_{a_h}, and Σ_{b_h} satisfy

$$\boldsymbol{\mu}_{a_h} = \frac{1}{\sigma^2}\Sigma_{a_h}\left(V - \sum_{h' \neq h} \boldsymbol{\mu}_{b_{h'}}\boldsymbol{\mu}_{a_{h'}}^{\top}\right)^{\top}\boldsymbol{\mu}_{b_h}, \tag{23}$$

$$\boldsymbol{\mu}_{b_h} = \frac{1}{\sigma^2}\Sigma_{b_h}\left(V - \sum_{h' \neq h} \boldsymbol{\mu}_{b_{h'}}\boldsymbol{\mu}_{a_{h'}}^{\top}\right)\boldsymbol{\mu}_{a_h}, \tag{24}$$

$$\Sigma_{a_h} = \left(\frac{1}{\sigma^2}\left(\|\boldsymbol{\mu}_{b_h}\|^2 + \mathrm{tr}(\Sigma_{b_h})\right) + c_{a_h}^{-2}\right)^{-1} I_M, \tag{25}$$

$$\Sigma_{b_h} = \left(\frac{1}{\sigma^2}\left(\|\boldsymbol{\mu}_{a_h}\|^2 + \mathrm{tr}(\Sigma_{a_h})\right) + c_{b_h}^{-2}\right)^{-1} I_L. \tag{26}$$

I_d denotes the d-dimensional identity matrix. One may search a local solution (i.e., a local minimum of the free energy (10)) by iterating Equations (23)–(26).

It is straightforward to see that the VB solution $\widehat{U}^{\mathrm{VB}}$ (see Equation (11)) can be expressed as

$$\widehat{U}^{\mathrm{VB}} = \sum_{h=1}^{H} \boldsymbol{\mu}_{b_h}\boldsymbol{\mu}_{a_h}^{\top}. \tag{27}$$

Then we have the following theorem (its proof is given in Appendix C):[1]

Theorem 2 *$\widehat{U}^{\mathrm{VB}}$ is expressed as*

$$\widehat{U}^{\mathrm{VB}} = \sum_{h=1}^{H} \widehat{\gamma}_h^{\mathrm{VB}}\boldsymbol{\omega}_{b_h}\boldsymbol{\omega}_{a_h}^{\top},$$

1. This theorem could be regarded as a more precise version of Theorem 1 given in Nakajima and Watanabe (2007).

where $\boldsymbol{\omega}_{a_h}$ *and* $\boldsymbol{\omega}_{b_h}$ *are the right and the left singular vectors of* V *(see Equation* (20)*). When* $\gamma_h > \sqrt{M\sigma^2}$, $\widehat{\gamma}_h^{\mathrm{VB}} (= \|\boldsymbol{\mu}_{a_h}\| \|\boldsymbol{\mu}_{b_h}\|)$ *is bounded as*

$$\max\left\{0, \left(1 - \frac{M\sigma^2}{\gamma_h^2}\right)\gamma_h - \frac{\sigma^2\sqrt{M/L}}{c_{a_h}c_{b_h}}\right\} \leq \widehat{\gamma}_h^{\mathrm{VB}} < \left(1 - \frac{M\sigma^2}{\gamma_h^2}\right)\gamma_h. \tag{28}$$

Otherwise, $\widehat{\gamma}_h^{\mathrm{VB}} = 0$.

The upper and lower bounds given in Equation (28) are illustrated in Figure 2. Theorem 2 states that, in the limit of $c_{a_h}c_{b_h} \to \infty$, the lower bound agrees with the upper bound and we have

$$\lim_{c_{a_h}c_{b_h}\to\infty} \widehat{\gamma}_h^{\mathrm{VB}} = \begin{cases} \max\left\{0, \left(1 - \dfrac{M\sigma^2}{\gamma_h^2}\right)\gamma_h\right\} & \text{if } \gamma_h > 0, \\ 0 & \text{otherwise.} \end{cases} \tag{29}$$

This is the same form as the *positive-part James-Stein (PJS) shrinkage estimator* (James and Stein, 1961; Efron and Morris, 1973) (see Appendix A for the details of the PJS estimator). The factor $M\sigma^2$ is the expected contribution of the noise to γ_h^2—when the target matrix is $U = 0$, the expectation of γ_h^2 over all h is given by $M\sigma^2$. When $\gamma_h^2 < M\sigma^2$, Equation (29) implies that $\widehat{\gamma}_h^{\mathrm{VB}} = 0$. Thus, the PJS estimator cuts off the singular components dominated by noise. As γ_h^2 increases, the PJS shrinkage factor $M\sigma^2/\gamma_h^2$ tends to 0, and thus the estimated singular value $\widehat{\gamma}_h^{\mathrm{VB}}$ becomes close to the original singular value γ_h.

Let us compare the behavior of the VB solution (29) with that of the MAP solution (21) when $c_{a_h}c_{b_h} \to \infty$. In this case, the MAP solution merely results in the ML solution where no regularization is incorporated. In contrast, VB offers PJS-type regularization even when $c_{a_h}c_{b_h} \to \infty$. Thus VB can still mitigate overfitting (or it can possibly cause underfitting). This fact is in good agreement with the experimental results reported in Raiko et al. (2007), where no overfitting was observed when $c_{a_h}^2 = 1$ and $c_{b_h}^2$ is set to large values. This counter-intuitive fact stems again from the non-identifiability of the MF model—the Gaussian noise $\mathcal{E}$ imposed in the space of U possesses a very complex surface in the joint space of A and B, in particular, *multimodal* structure. This causes the MAP solution to be distinctively different from the VB solution. We call this regularization effect *model-induced regularization*. In Section 4.2, we investigate the effect of model-induced regularization in more detail using illustrative examples.

The following theorem more precisely specifies under which condition the VB estimator is strictly positive or zero (its proof is also included in Appendix C):

Theorem 3 *It holds that*

$$\widehat{\gamma}_h^{\mathrm{VB}} = 0 \text{ if } \gamma_h \leq \widetilde{\gamma}_h^{\mathrm{VB}},$$
$$\widehat{\gamma}_h^{\mathrm{VB}} > 0 \text{ if } \gamma_h > \widetilde{\gamma}_h^{\mathrm{VB}},$$

where

$$\widetilde{\gamma}_h^{\mathrm{VB}} = \sqrt{\frac{(L+M)\sigma^2}{2} + \frac{\sigma^4}{2c_{a_h}^2c_{b_h}^2} + \sqrt{\left(\frac{(L+M)\sigma^2}{2} + \frac{\sigma^4}{2c_{a_h}^2c_{b_h}^2}\right)^2 - LM\sigma^4}}. \tag{30}$$

$\widetilde{\gamma}_h^{\mathrm{VB}}$ is monotone decreasing with respect to $c_{a_h}c_{b_h}$, and is lower-bounded as

$$\widetilde{\gamma}_h^{\mathrm{VB}} > \lim_{c_{a_h}c_{b_h}\to\infty} \widetilde{\gamma}_h^{\mathrm{VB}} = \sqrt{M\sigma^2}.$$

As shown in Equation (21), $\widehat{\gamma}_h^{\mathrm{MAP}}$ satisfies

$$\widehat{\gamma}_h^{\mathrm{MAP}} = 0 \text{ if } \gamma_h \le \widetilde{\gamma}_h^{\mathrm{MAP}},$$
$$\widehat{\gamma}_h^{\mathrm{MAP}} > 0 \text{ if } \gamma_h > \widetilde{\gamma}_h^{\mathrm{MAP}},$$

where

$$\widetilde{\gamma}_h^{\mathrm{MAP}} = \frac{\sigma^2}{c_{a_h}c_{b_h}}.$$

Since

$$\widetilde{\gamma}_h^{\mathrm{VB}} > \sqrt{\frac{\sigma^4}{c_{a_h}^2 c_{b_h}^2}} = \widetilde{\gamma}_h^{\mathrm{MAP}},$$

VB has a stronger shrinkage effect than MAP in terms of the vanishing condition of singular values.

We can derive another upper bound of $\widehat{\gamma}_h^{\mathrm{VB}}$, which depends on hyperparameters c_{a_h} and c_{b_h} (its proof is also included in Appendix C):

Theorem 4 *When $\gamma_h > \sqrt{M\sigma^2}$, $\widehat{\gamma}_h^{\mathrm{VB}}$ is upper-bounded as*

$$\widehat{\gamma}_h^{\mathrm{VB}} \le \sqrt{\left(1-\frac{L\sigma^2}{\gamma_h^2}\right)\left(1-\frac{M\sigma^2}{\gamma_h^2}\right)}\cdot\gamma_h - \frac{\sigma^2}{c_{a_h}c_{b_h}}. \tag{31}$$

When $L=M$ and $\gamma_h > \sqrt{M\sigma^2}$, the lower bound in Equation (28) and the upper bound in Equation (31) agree with each other. Thus, we have an analytic-form expression of $\widehat{\gamma}_h^{\mathrm{VB}}$ as follows:

$$\widehat{\gamma}_h^{\mathrm{VB}} = \begin{cases} \max\left\{0, \left(1-\frac{M\sigma^2}{\gamma_h^2}\right)\gamma_h - \frac{\sigma^2}{c_{a_h}c_{b_h}}\right\} & \text{if } \gamma_h > 0, \\ 0 & \text{otherwise.} \end{cases} \tag{32}$$

Then, the complete VB posterior can also be obtained analytically (its proof is given in Appendix D):

Corollary 1 *When $L=M$, the VB posteriors are given by*

$$r_{\mathrm{A}}(A|V) = \prod_{h=1}^{H} \mathcal{N}_M(\boldsymbol{a}_h; \boldsymbol{\mu}_{a_h}, \Sigma_{a_h}),$$
$$r_{\mathrm{B}}(B|V) = \prod_{h=1}^{H} \mathcal{N}_M(\boldsymbol{b}_h; \boldsymbol{\mu}_{b_h}, \Sigma_{b_h}),$$

where, for $\widehat{\gamma}_h^{\mathrm{VB}}$ *given by Equation* (32),

$$\boldsymbol{\mu}_{a_h} = \pm\sqrt{\frac{c_{a_h}}{c_{b_h}}\widehat{\gamma}_h^{\mathrm{VB}}}\cdot\boldsymbol{\omega}_{a_h}, \tag{33}$$

$$\boldsymbol{\mu}_{b_h} = \pm\sqrt{\frac{c_{b_h}}{c_{a_h}}\widehat{\gamma}_h^{\mathrm{VB}}}\cdot\boldsymbol{\omega}_{b_h}, \tag{34}$$

$$\Sigma_{a_h} = \frac{c_{a_h}}{2c_{b_h}M}\left(\sqrt{\left(\widehat{\gamma}_h^{\mathrm{VB}}+\frac{\sigma^2}{c_{a_h}c_{b_h}}\right)^2+4\sigma^2M}-\left(\widehat{\gamma}_h^{\mathrm{VB}}+\frac{\sigma^2}{c_{a_h}c_{b_h}}\right)\right)I_M, \tag{35}$$

$$\Sigma_{b_h} = \frac{c_{b_h}}{2c_{a_h}M}\left(\sqrt{\left(\widehat{\gamma}_h^{\mathrm{VB}}+\frac{\sigma^2}{c_{a_h}c_{b_h}}\right)^2+4\sigma^2M}-\left(\widehat{\gamma}_h^{\mathrm{VB}}+\frac{\sigma^2}{c_{a_h}c_{b_h}}\right)\right)I_M. \tag{36}$$

3.3 EMAPMF

In the EMAPMF framework, the hyperparameters c_{a_h} and c_{b_h} are determined so that the Bayes posterior $p(A,B|V)$ is maximized (equivalently, the negative log posterior is minimized).

Differentiating the negative log posterior (17) with respect to $c_{a_h}^2$ and $c_{b_h}^2$ and setting the derivatives to zero lead to the following optimality conditions.

$$c_{a_h}^2 = \frac{\|\boldsymbol{a}_h\|^2}{M}, \tag{37}$$

$$c_{b_h}^2 = \frac{\|\boldsymbol{b}_h\|^2}{L}. \tag{38}$$

Alternating Equations (18), (19), (37), and (38), one may learn the parameters A,B and the hyperparameters c_{a_h},c_{b_h} at the same time.

However, as pointed out in Raiko et al. (2007), EMAPMF does not work properly since its objective (17) is unbounded from below at $\boldsymbol{a}_h,\boldsymbol{b}_h=\mathbf{0}$ and $c_{a_h},c_{b_h}\to 0$. Thus we end up in merely finding the trivial solution $(\boldsymbol{a}_h,\boldsymbol{b}_h=\mathbf{0})$ unless the iterative algorithm is stuck at some local optimum.

3.4 EVBMF

For the trial distribution (14), the VB free energy (10) can be written as follows:

$$\begin{aligned}F_{\mathrm{VB}}(r|V,\{c_{a_h}^2,c_{b_h}^2\}) = {}& \frac{LM}{2}\log\sigma^2+\sum_{h=1}^{H}\left(\frac{M}{2}\log c_{a_h}^2-\frac{1}{2}\log|\Sigma_{a_h}|+\frac{\|\boldsymbol{\mu}_{a_h}\|^2+\mathrm{tr}(\Sigma_{a_h})}{2c_{a_h}^2}\right.\\ &\left.+\frac{L}{2}\log c_{b_h}^2-\frac{1}{2}\log|\Sigma_{b_h}|+\frac{\|\boldsymbol{\mu}_{b_h}\|^2+\mathrm{tr}(\Sigma_{b_h})}{2c_{b_h}^2}\right)\\ &+\frac{1}{2\sigma^2}\left\|V-\sum_{h=1}^{H}\boldsymbol{\mu}_{b_h}\boldsymbol{\mu}_{a_h}^\top\right\|_{\mathrm{Fro}}^2\\ &+\frac{1}{2\sigma^2}\sum_{h=1}^{H}\left(\|\boldsymbol{\mu}_{a_h}\|^2\mathrm{tr}(\Sigma_{b_h})+\mathrm{tr}(\Sigma_{a_h})\|\boldsymbol{\mu}_{b_h}\|^2+\mathrm{tr}(\Sigma_{a_h})\mathrm{tr}(\Sigma_{b_h})\right),\end{aligned} \tag{39}$$

where $|\cdot|$ denotes the determinant of a matrix. Differentiating Equation (39) with respect to $c_{a_h}^2$ and $c_{b_h}^2$ and setting the derivatives to zero, we obtain the following optimality conditions:

$$c_{a_h}^2 = \frac{\|\boldsymbol{\mu}_{a_h}\|^2 + \mathrm{tr}(\Sigma_{a_h})}{M}, \tag{40}$$

$$c_{b_h}^2 = \frac{\|\boldsymbol{\mu}_{b_h}\|^2 + \mathrm{tr}(\Sigma_{b_h})}{L}. \tag{41}$$

Here, we observe the invariance of Equation (39) with respect to the transform

$$\left\{(\boldsymbol{\mu}_{a_h}, \boldsymbol{\mu}_{b_h}, \Sigma_{a_h}, \Sigma_{b_h}, c_{a_h}^2, c_{b_h}^2)\right\} \to \left\{(s_h^{1/2}\boldsymbol{\mu}_{a_h}, s_h^{-1/2}\boldsymbol{\mu}_{b_h}, s_h\Sigma_{a_h}, s_h^{-1}\Sigma_{b_h}, s_h c_{a_h}^2, s_h^{-1}c_{b_h}^2)\right\} \tag{42}$$

for any $\{s_h \in \mathbb{R}; s_h > 0, h = 1, \ldots, H\}$. This redundancy can be eliminated by fixing the ratio between the hyperparameters to some constant—we choose 1 without loss of generality:

$$\frac{c_{a_h}}{c_{b_h}} = 1. \tag{43}$$

Then, Equations (40) and (41) yield

$$c_{a_h}^2 = \sqrt{\frac{(\|\boldsymbol{\mu}_{a_h}\|^2 + \mathrm{tr}(\Sigma_{a_h}))\,(\|\boldsymbol{\mu}_{b_h}\|^2 + \mathrm{tr}(\Sigma_{b_h}))}{LM}}, \tag{44}$$

$$c_{b_h}^2 = \sqrt{\frac{(\|\boldsymbol{\mu}_{a_h}\|^2 + \mathrm{tr}(\Sigma_{a_h}))\,(\|\boldsymbol{\mu}_{b_h}\|^2 + \mathrm{tr}(\Sigma_{b_h}))}{LM}}. \tag{45}$$

One may learn the parameters A, B and the hyperparameters c_{a_h}, c_{b_h} by applying Equations (44) and (45) after every iteration of Equations (23)–(26) (this gives a local minimum of Equation (39) at convergence).

For the EVB solution $\widehat{U}^{\mathrm{EVB}}$, we have the following theorem (its proof is provided in Appendix E):

Theorem 5 *The EVB estimator is given by the following form:*

$$\widehat{U}^{\mathrm{EVB}} = \sum_{h=1}^{H} \widehat{\gamma}_h^{\mathrm{EVB}} \boldsymbol{\omega}_{b_h} \boldsymbol{\omega}_{a_h}^{\top}.$$

$\widehat{\gamma}_h^{\mathrm{EVB}} = 0$ *if* $\gamma_h < \underline{\gamma}_h^{\mathrm{EVB}}$, *where*

$$\underline{\gamma}_h^{\mathrm{EVB}} = \left(\sqrt{L} + \sqrt{M}\right)\sigma.$$

If $\gamma_h \geq \underline{\gamma}_h^{\mathrm{EVB}}$, $\widehat{\gamma}_h^{\mathrm{EVB}}$ *is upper-bounded as*

$$\widehat{\gamma}_h^{\mathrm{EVB}} < \left(1 - \frac{M\sigma^2}{\gamma_h^2}\right)\gamma_h. \tag{46}$$

If $\gamma_h \geq \overline{\gamma}_h^{\mathrm{EVB}}$, *where*

$$\overline{\gamma}_h^{\mathrm{EVB}} = \sqrt{7M} \cdot \sigma > \underline{\gamma}_h^{\mathrm{EVB}},$$

$\widehat{\gamma}_h^{\mathrm{EVB}}$ *is lower-bounded as*

$$\widehat{\gamma}_h^{\mathrm{EVB}} > \max\left\{0, \left(1 - \frac{2M\sigma^2}{\gamma_h^2 - \sqrt{\gamma_h^2(L+M+\sqrt{LM})\sigma^2}}\right)\gamma_h\right\}. \tag{47}$$

Theorem 5 implies that

$$\widehat{\gamma}_h^{\mathrm{EVB}} = 0 \text{ if } \gamma_h < \underline{\gamma}_h^{\mathrm{EVB}},$$
$$\widehat{\gamma}_h^{\mathrm{EVB}} > 0 \text{ if } \gamma_h \geq \overline{\gamma}_h^{\mathrm{EVB}}.$$

When

$$\underline{\gamma}_h^{\mathrm{EVB}} \leq \gamma_h < \overline{\gamma}_h^{\mathrm{EVB}},$$

our theoretical analysis is not precise enough to conclude whether $\widehat{\gamma}_h^{\mathrm{EVB}}$ is zero or not. As explained in Section 3.3, EMAP always results in the trivial solution (i.e., $\widehat{\gamma}_h^{\mathrm{EMAP}} = 0$). In contrast, Theorem 5 states that EVB gives a non-trivial solution (i.e., $\widehat{\gamma}_h^{\mathrm{EVB}} > 0$) when $\gamma_h \geq \overline{\gamma}_h^{\mathrm{EVB}}$. Since $\lim_{c_{a_h}c_{b_h}\to\infty}\widetilde{\gamma}_h^{\mathrm{VB}} = \sqrt{M\sigma^2} < \underline{\gamma}_h^{\mathrm{EVB}}$ (see Theorem 3), EVB has stronger shrinkage effect than VB with flat priors in terms of the vanishing condition of singular values.

It is also note worthy that the upper bound in Equation (46) is the same as that in Theorem 2. Thus, even when the hyperparameters c_{a_h} and c_{b_h} are learned from data by EVB, the same upper bound as the fixed-hyperparameter case in VB holds.

Another upper bound of $\widehat{\gamma}_h^{\mathrm{EVB}}$ is given as follows (its proof is also included in Appendix E):

Theorem 6 *When* $\gamma_h \geq \underline{\gamma}_h^{\mathrm{EVB}}$ *(*$= (\sqrt{L}+\sqrt{M})\sigma$*),* $\widehat{\gamma}_h^{\mathrm{EVB}}$ *is upper-bounded as*

$$\widehat{\gamma}_h^{\mathrm{EVB}} < \sqrt{\left(1 - \frac{L\sigma^2}{\gamma_h^2}\right)\left(1 - \frac{M\sigma^2}{\gamma_h^2}\right)}\gamma_h - \frac{\sqrt{LM}\sigma^2}{\gamma_h}. \tag{48}$$

Note that the right-hand side of (48) is strictly positive under $\gamma_h \geq \underline{\gamma}_h^{\mathrm{EVB}}$.

When $L = M$, the upper bound in Equation (48) is sharper than that in Equation (46), resulting in

$$\widehat{\gamma}_h^{\mathrm{EVB}} < \left(1 - \frac{2M\sigma^2}{\gamma_h^2}\right)\gamma_h. \tag{49}$$

The PJS shrinkage factor of the upper bound (49) is $2M\sigma^2/\gamma_h^2$. On the other hand, as shown in Equation (29), the PJS shrinkage factor of the plain VB with uniform priors on A and B (i.e., $c_a, c_b \to \infty$) is $M\sigma^2/\gamma_h^2$, which is *less than a half* of EVB. Thus, EVB provides substantially stronger regularization effect than the plain VB with uniform priors. Furthermore, from Equation (32), we can confirm that the upper bound (49) is equivalent to the VB solution when $c_{a_h}c_{b_h} = \gamma_h/M$.

When $L = M$, the complete EVB posterior is obtained analytically by using the following corollary (the proof is given in Appendix F):

Corollary 2 *For* $\gamma_h \geq 2\sqrt{M}\sigma$*, we define*

$$\varphi(\gamma_h) = \log\left(\frac{\gamma_h^2}{M\sigma^2}(1-\rho_-)\right) - \frac{\gamma_h^2}{M\sigma^2}(1-\rho_-) + \left(1 + \frac{\gamma_h^2}{2M\sigma^2}\rho_+^2\right), \tag{50}$$

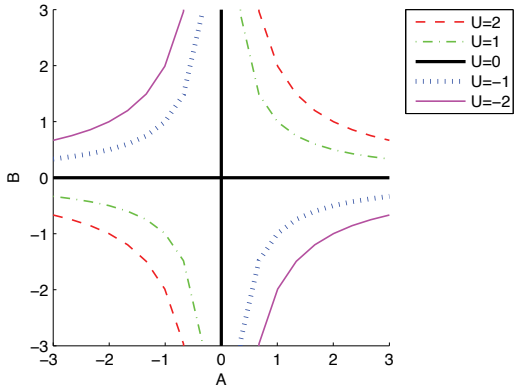


Figure 3: Equivalence class. Any A and B such that their product is unchanged give the same U.

where

$$\rho_{\pm} = \sqrt{\frac{1}{2}\left(1 - \frac{2M\sigma^2}{\gamma_h^2} \pm \sqrt{1 - \frac{4M\sigma^2}{\gamma_h^2}}\right)}.$$

Suppose $L = M$. *If* $\gamma_h \geq 2\sqrt{M}\sigma$ *and* $\varphi(\gamma_h) \leq 0$, *then the EVB estimator of* $c_{a_h}c_{b_h}$ *is given by*

$$\widehat{c}_{a_h}^{EVB}\widehat{c}_{b_h}^{EVB} = \frac{\gamma_h}{M}\rho_{+}. \tag{51}$$

Otherwise, $\widehat{c}_{a_h}^{EVB}\widehat{c}_{b_h}^{EVB} \to 0$. *The EVB posterior is obtained by Corollary 1 with*

$$(c_{a_h}^2, c_{b_h}^2) = \left(\widehat{c}_{a_h}^{EVB}\widehat{c}_{b_h}^{EVB}, \widehat{c}_{a_h}^{EVB}\widehat{c}_{b_h}^{EVB}\right).$$

Furthermore, when $\gamma_h \geq \sqrt{7M}\sigma$, *it holds that*

$$\varphi(\gamma_h) < 0. \tag{52}$$

Given γ_h, Equation (50) and then Equation (51) are computed analytically. By substituting Equations (51) and (43) into Equations (33)–(36), the complete EVB posterior is obtained. In Section 4.3, properties of EVBMF along with the behavior of the function (50) are further investigated through numerical examples.

4. Illustration of Influence of Non-identifiability

In order to understand the regularization mechanism of the Bayesian MF methods more intuitively, we illustrate the influence of non-identifiability when $L = M = H = 1$ (i.e., U, V, A, and B are merely scalars). In this case, any A and B such that their product is unchanged form an *equivalence class* and give the same U (see Figure 3). When $U = 0$, the equivalence class has a 'cross-shape' profile on the A- and B-axes; otherwise, it forms a pair of hyperbolic curves.

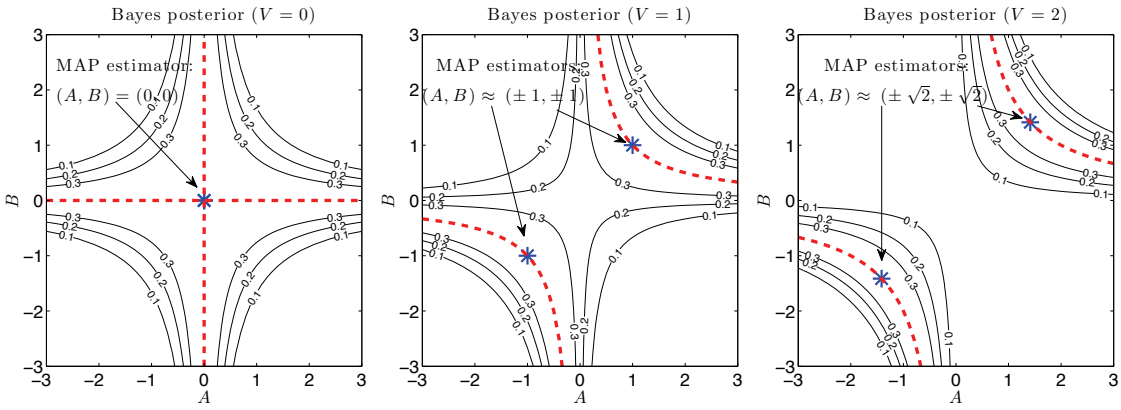


Figure 4: Bayes posteriors with $c_a = c_b = 100$ (i.e., almost flat priors). The asterisks are the MAP solutions, and the dashed lines indicate the ML solutions (the modes of the contour when $c_a = c_b = c \to \infty$).

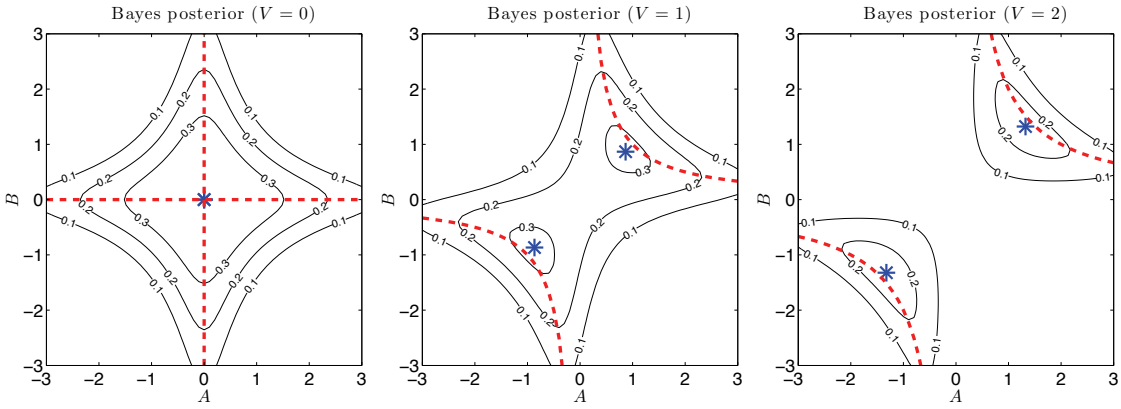


Figure 5: Bayes posteriors with $c_a = c_b = 2$. The dashed lines indicating the ML solutions are identical to those in Figure 4.

4.1 MAPMF

First, we illustrate the behavior of the MAP estimator.

When $L = M = H = 1$, Equation (17) yields that the Bayes posterior $p(A,B|V)$ is given as

$$p(A,B|V) \propto \exp\left(-\frac{1}{2\sigma^2}(V - BA)^2 - \frac{A^2}{2c_a^2} - \frac{B^2}{2c_b^2}\right). \tag{53}$$

Figure 4 shows the contour of the above Bayes posterior when $V = 0, 1, 2$ are observed, where the noise variance is $\sigma^2 = 1$ and the hyperparameters are $c_a = c_b = 100$ (i.e., almost flat priors). When $V = 0$, the surface of the Bayes posterior has a cross-shape profile and its maximum is at the origin. When $V > 0$, the surface is divided into the positive orthant (i.e., $A, B > 0$) and the negative orthant (i.e., $A, B < 0$), and the two 'modes' get farther as V increases.

For finite c_a and c_b, Theorem 1 and Equation (66) (in Appendix B) imply that the MAP solution can be expressed as

$$\widehat{A}^{\mathrm{MAP}} = \pm\sqrt{\frac{c_a}{c_b}\max\left\{0, |V| - \frac{\sigma^2}{c_a c_b}\right\}},$$
$$\widehat{B}^{\mathrm{MAP}} = \pm\mathrm{sign}(V)\sqrt{\frac{c_b}{c_a}\max\left\{0, |V| - \frac{\sigma^2}{c_a c_b}\right\}},$$

where $\mathrm{sign}(\cdot)$ denotes the sign of a scalar. In Figure 4, the asterisks indicate the MAP estimators, and the dashed lines indicate the ML estimators (the modes of the contour of Equation (53) when $c_a = c_b = c \to \infty$). When $V = 0$, the Bayes posterior takes the maximum value on the A- and B-axes, which results in $\widehat{U}^{\mathrm{MAP}} = 0$. When $V = 1$, the profile of the Bayes posterior is hyperbolic and the maximum value is achieved on the hyperbolic curves in the positive orthant (i.e., $A, B > 0$) and the negative orthant (i.e., $A, B < 0$); in either case, $\widehat{U}^{\mathrm{MAP}} \approx 1$ (and $\widehat{U}^{\mathrm{MAP}} \to 1$ as $c_a, c_b \to \infty$). When $V = 2$, a similar multimodal structure is observed and the solution is $\widehat{U}^{\mathrm{MAP}} \approx 2$ (and $\widehat{U}^{\mathrm{MAP}} \to 2$ as $c_a, c_b \to \infty$). From these plots, we can visually confirm that the MAP solution with almost flat priors ($c_a = c_b = 100$) approximately agrees with the ML solution: $\widehat{U}^{\mathrm{MAP}} \approx \widehat{U}^{\mathrm{ML}} = V$ (and $\widehat{U}^{\mathrm{MAP}} \to \widehat{U}^{\mathrm{ML}}$ as $c_a, c_b \to \infty$).

Furthermore, these graphs illustrate the reason why the product $c_a c_b \to \infty$ is sufficient for MAP to agree with ML in the MF setup (see Section 3.1). Suppose c_a is kept small, say $c_a = 1$, in Figure 4. Then the Gaussian 'decay' remains along the horizontal axis in the profile of the Bayes posterior. However, the MAP solution $\widehat{U}^{\mathrm{MAP}}$ does not change since the mode of the Bayes posterior is kept lying on the dashed line (equivalence class). Thus, MAP agrees with ML if either c_a or c_b tends to infinity.

Figure 5 shows the contour of the Bayes posterior when $c_a = c_b = 2$. The MAP estimators are shifted from the ML estimators (dashed lines) toward the origin, and they are more clearly contoured as peaks.

4.2 VBMF

Here, we illustrate the behavior of the VB estimator, where the Bayes posterior is approximated by a spherical Gaussian.

In the current one-dimensional setup, Corollary 1 implies that the VB posteriors $r_{\mathrm{A}}(A|V)$ and $r_{\mathrm{B}}(B|V)$ can be expressed as

$$r_{\mathrm{A}}(A|V) = \mathcal{N}(A; \pm\sqrt{\widehat{\gamma}^{\mathrm{VB}} c_a/c_b}, \zeta c_a/c_b),$$
$$r_{\mathrm{B}}(B|V) = \mathcal{N}(B; \pm\mathrm{sign}(V)\sqrt{\widehat{\gamma}^{\mathrm{VB}} c_b/c_a}, \zeta c_b/c_a),$$

where $\mathcal{N}(\cdot; \mu, \sigma^2)$ denotes the Gaussian density with mean μ and variance σ^2, and

$$\zeta = \sqrt{\left(\frac{\widehat{\gamma}^{\mathrm{VB}}}{2} + \frac{\sigma^2}{2c_a c_b}\right)^2 + \sigma^2} - \left(\frac{\widehat{\gamma}^{\mathrm{VB}}}{2} + \frac{\sigma^2}{2c_a c_b}\right),$$
$$\widehat{\gamma}^{\mathrm{VB}} = \begin{cases} \max\left\{0, \left(1 - \frac{\sigma^2}{V^2}\right)|V| - \frac{\sigma^2}{c_a c_b}\right\} & \text{if } V \neq 0, \\ 0 & \text{otherwise.} \end{cases}$$

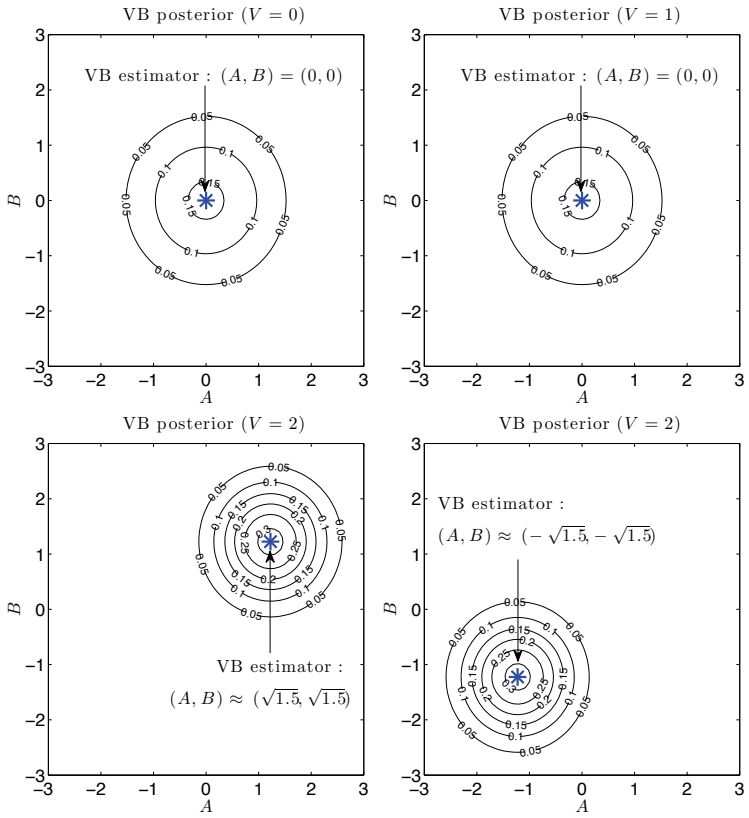


Figure 6: VB posteriors and VB solutions when $L = M = 1$ (i.e., the matrices V, U, A, and B are scalars). When $V = 2$, VB gives either one of the two solutions shown in the bottom row.

Figure 6 shows the contour of the VB posterior $r(A,B|V) = r_{\mathrm{A}}(A|V)r_{\mathrm{B}}(B|V)$ when $V = 0,1,2$ are observed, where the noise variance is $\sigma^2 = 1$ and the hyperparameters are $c_a = c_b = 100$ (i.e., almost flat priors). When $V = 0$, the cross-shaped contour of the Bayes posterior (see Figure 4) is approximated by a spherical Gaussian function located at the origin. Thus, the VB estimator is $\widehat{U}^{\mathrm{VB}} = 0$, which is equivalent to the MAP solution. When $V = 1$, two hyperbolic 'modes' of the Bayes posterior are approximated again by a spherical Gaussian function located at the origin. Thus, the VB estimator is still $\widehat{U}^{\mathrm{VB}} = 0$, which is different from the MAP solution.

$V = \widetilde{\gamma}_h^{\mathrm{VB}} \approx \sqrt{M\sigma^2} = 1$ ($\widetilde{\gamma}_h^{\mathrm{VB}} \to \sqrt{M\sigma^2}$ as $c_a, c_b \to \infty$) is actually a transition point of the behavior of the VB estimator. When V is not larger than the threshold $\sqrt{M\sigma^2}$, the VB method tries to approximate the two 'modes' of the Bayes posterior by the origin-centered Gaussian function. When V goes beyond the threshold $\sqrt{M\sigma^2}$, the 'distance' between two hyperbolic modes of the Bayes posterior becomes so large that the VB method chooses to approximate one of the two modes in the positive and negative orthants. As such, the symmetry is broken spontaneously and the VB solution is detached from the origin. Note that, as discussed in Section 3, $M\sigma^2$ amounts to the expected contribution of noise $\mathcal{E}$ to the squared singular value γ^2 ($= V^2$ in the current setup).

The bottom row of Figure 6 shows the contour of two possible VB posteriors when $V = 2$. Note that, in either case, the VB solution is the same: $\widehat{U}^{\mathrm{VB}} \approx 3/2$. The VB solution is closer to the origin

than the MAP solution $\widehat{U}^{\mathrm{MAP}} = 2$, and the difference between the VB and MAP solutions tends to shrink as V increases.

4.3 EVBMF

Next, we illustrate the behavior of the EVB estimator.

In the current one-dimensional setup, the free energy (39) is expressed as

$$F_{\mathrm{VB}}(r|V,c_a^2,c_b^2) = \log\frac{c_a^2c_b^2}{\Sigma_a\Sigma_b} + \frac{\mu_a^2+\Sigma_a}{2c_a^2} + \frac{\mu_b^2+\Sigma_b}{2c_b^2} - \frac{1}{\sigma^2}V\mu_a\mu_b + \frac{1}{2\sigma^2}\left(\mu_a^2+\Sigma_a\right)\left(\mu_b^2+\Sigma_b\right) + \mathrm{Const.}$$

According to Corollary 2, if $|V| \geq 2\sigma$ and $\varphi(|V|) \leq 0$, the EVB estimator of the hyperparameters is given by

$$(\widehat{c}_a^{\mathrm{EVB}})^2 = (\widehat{c}_b^{\mathrm{EVB}})^2 = |V|\rho_+, \tag{54}$$

where

$$\varphi(|V|) = \log\left(\frac{|V|^2}{\sigma^2}(1-\rho_-)\right) - \frac{|V|^2}{\sigma^2}(1-\rho_-) + \left(1+\frac{|V|^2}{2\sigma^2}\rho_+^2\right),$$

$$\rho_\pm = \sqrt{\frac{1}{2}\left(1-\frac{\sigma^2}{|V|^2} \pm \sqrt{1-\frac{4\sigma^2}{|V|^2}}\right)}.$$

Based on a simple numerical evaluation (Figure 7) of $\varphi(|V|)$, we can confirm that Equation (54) holds if $|V| \geq \widetilde{\gamma}^{\mathrm{EVB}}$, where

$$\widetilde{\gamma}^{\mathrm{EVB}} \approx 2.22.$$

Otherwise $\widehat{c}_{a_h}^{\mathrm{EVB}}, \widehat{c}_{b_h}^{\mathrm{EVB}} \to 0$. Note that $\widetilde{\gamma}^{\mathrm{EVB}}$ is theoretically bounded as

$$\left(2 = 2\sigma^2 =\right) \underline{\gamma}^{\mathrm{EVB}} \leq \widetilde{\gamma}^{\mathrm{EVB}} \leq \overline{\gamma}^{\mathrm{EVB}} \left(= \sqrt{7}\sigma^2 \approx 2.64\right),$$

as shown in Equation (52).

Using Corollary 1 with Equation (54), we can plot the EVB posterior. When

$$|V| < \widetilde{\gamma}^{\mathrm{EVB}} \approx 2.22,$$

the infimum of the free energy with respect to $(\mu_a,\mu_b,\Sigma_a,\Sigma_b,c_a^2,c_b^2)$ is attained by $c_a^2 = c_b^2 = \varepsilon$, $\mu_a = \mu_b = 0$, and

$$\Sigma_a = \Sigma_b = \frac{\sigma^2}{2\varepsilon}\left(\sqrt{1+\frac{4n\varepsilon^2}{\sigma^2}} - 1\right),$$

where $\varepsilon \to 0$ (i.e., $c_a^2 = c_b^2 \to 0$, $\mu_a = \mu_b = 0$, and $\Sigma_a = \Sigma_b \to 0$). Therefore, the Gaussian width of the EVB posterior approaches zero (i.e., *Dirac's delta function* located at the origin). The left graph of Figure 8 illustrates the contour of the EVB posterior $r(A,B|V) = r_{\mathrm{A}}(A|V)r_{\mathrm{B}}(B|V)$ when $V = 2$

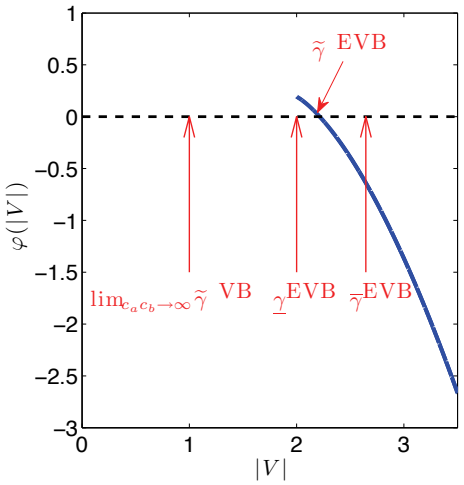


Figure 7: Numerical evaluation of $\varphi(|V|)$ when $L = M = 1$ and $\sigma^2 = 1$ (the blue solid curve). The blue solid curve crosses the black dashed line ($\varphi(|V|) = 0$) at $|V| = \widetilde{\gamma}^{\mathrm{EVB}} \approx 2.22$.

is observed, where the noise variance is $\sigma^2 = 1$. Since $\widehat{U}^{\mathrm{MAP}} \approx 2$ and $\widehat{U}^{\mathrm{VB}} \approx 1.5$ under almost flat priors (see Figure 4 and Figure 6), $\widehat{U}^{\mathrm{EVB}} = 0$ is more strongly regularized than VB and MAP.

On the other hand, when

$$|V| \geq \widetilde{\gamma}^{\mathrm{EVB}} \approx 2.22,$$

the EVB posteriors $r_{\mathrm{A}}(A|V)$ and $r_{\mathrm{B}}(B|V)$ can be expressed as

$$r_{\mathrm{A}}(A|V) = \mathcal{N}(A; \pm\sqrt{\widehat{\gamma}^{\mathrm{EVB}}}, \zeta),$$
$$r_{\mathrm{B}}(B|V) = \mathcal{N}(B; \pm\mathrm{sign}(V)\sqrt{\widehat{\gamma}^{\mathrm{EVB}}}, \zeta),$$

where

$$\zeta = \sqrt{\left(\frac{\widehat{\gamma}^{\mathrm{EVB}}}{2} + \frac{|V|\rho_-}{2}\right)^2 + \sigma^2} - \left(\frac{\widehat{\gamma}^{\mathrm{EVB}}}{2} + \frac{|V|\rho_-}{2}\right),$$
$$\rho_- = \sqrt{\frac{1}{2}\left(1 - \frac{2\sigma^2}{\gamma_h^2} - \sqrt{1 - \frac{4\sigma^2}{\gamma_h^2}}\right)},$$
$$\widehat{\gamma}^{\mathrm{EVB}} = \left(1 - \frac{\sigma^2}{V^2} - \rho_-\right)|V|.$$

When $V = 3$ is observed, we have $\widehat{U}^{\mathrm{EVB}} \approx 2.28$ ($c_a^2 = c_b^2 \approx 2.62$, $\mu_a = \mu_b \approx \sqrt{2.28}$, and $\Sigma_a = \Sigma_b \approx 0.33$). The possible posteriors are plotted in the middle and the right graphs of Figure 8. Since $\widehat{U}^{\mathrm{MAP}} \approx 3$ and $\widehat{U}^{\mathrm{VB}} = 3/8 \approx 2.67$ under almost flat priors, EVB has stronger regularization effect than VB and MAP.

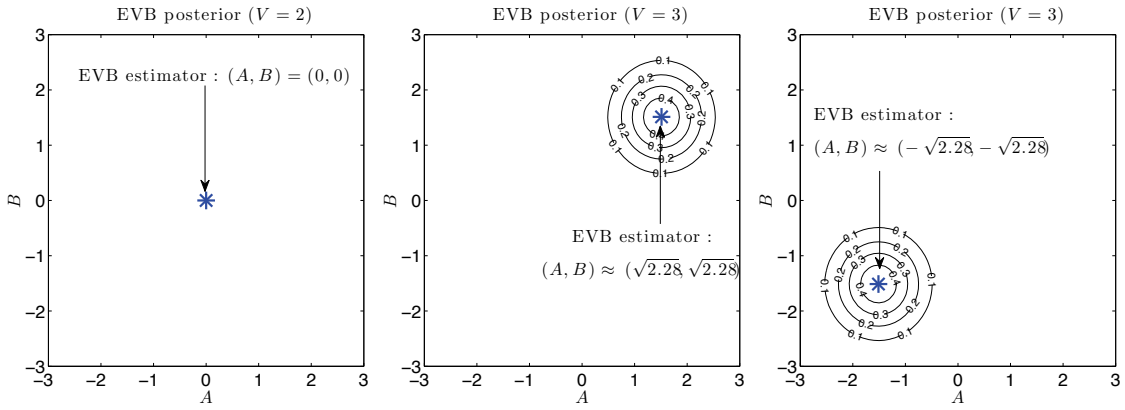


Figure 8: EVB posteriors and EVB solutions when $L = M = 1$. Left: When $V = 2$, the EVB posterior is reduced to Dirac's delta function located at the origin. Right: When $V = 3$, the solution is detached from the origin and given by $(A,B) \approx (\sqrt{2.28}, \sqrt{2.28})$ or $(A,B) \approx (-\sqrt{2.28}, -\sqrt{2.28})$, which both yields the same solution $\widehat{U}^{\mathrm{EVB}} \approx 2.28$.

4.4 FBMF

Here, we illustrate the behavior of the FB estimator.

When $L = M = H = 1$, the FB solution (5) is expressed as

$$\widehat{U}^{\mathrm{FB}} = \langle AB \rangle_{p(V|A,B)\phi_{\mathrm{A}}(A)\phi_{\mathrm{B}}(B)}. \tag{55}$$

If $V = 0, 1, 2, 3$ are observed, the FB solutions with almost flat priors are $0, 0.92, 1.93, 2.95$, respectively, which were numerically computed.[2] Since the corresponding MAP solutions (with the almost flat priors) are $0, 1, 2, 3$, FB and MAP were shown to produce different solutions.

The theory by Jeffreys (1946) explains the origin of *model-induced regularization* in FB. Let us consider the *non-factorizing* model

$$p(V|A,B) \propto \exp\left(-\frac{1}{2\sigma^2}\|V - U\|_{\mathrm{Fro}}^2\right), \tag{56}$$

where U itself is the parameter to be estimated. The Jeffreys (non-informative) prior for this model is uniform

$$\phi_{\mathrm{U}}^{\mathrm{Jef}}(U) \propto 1. \tag{57}$$

On the other hand, the Jeffreys prior for the MF model (1) is given by

$$\phi_{\mathrm{A,B}}^{\mathrm{Jef}}(A,B) \propto \sqrt{A^2 + B^2}, \tag{58}$$

which is illustrated in Figure 9 (see Appendix I for the derivation of Equations (57) and (58)). Note that $\phi_{\mathrm{U}}^{\mathrm{Jef}}(U)$ and $\phi_{\mathrm{A,B}}^{\mathrm{Jef}}(A,B)$ are both *improper*.

2. More precisely, we numerically calculated the FB solution (55) by sampling A and B from the almost flat prior distributions $\phi_{\mathrm{A}}(A)\phi_{\mathrm{B}}(B)$ with $c_a = c_b = 100$ and taking the sample average of $AB \cdot p(V|A,B)$.

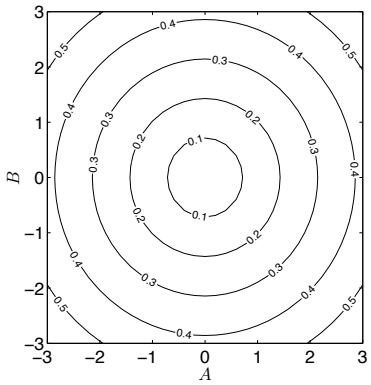


Figure 9: The Jeffreys non-informative prior of the MF model in the joint space of A and B: $\phi_{\mathrm{Jef}}(A,B) \propto \sqrt{A^2+B^2}$. The scaling of the density value in the graph is arbitrary due to impropriety.

Jeffreys (1946) states that the both combinations, the *non-factorizing* model (56) with its Jeffreys prior (57) and the MF model (1) with its Jeffreys prior (58), give the equivalent FB solution. We can easily show that the former combination, Equations (56) and (57), gives an unregularized solution. Thus, the FB solution in the MF model (1) with its Jeffreys prior (58) is also unregularized. Since the flat prior on (A,B) has more probability mass around the origin than the Jeffreys prior (58) (see Figure 9), it favors smaller $|U|$ and regularizes the FB solution.

4.5 EMAPMF

As explained in Section 3.3, EMAPMF always results in the trivial solution, $A,B=0$ and $c_{a_h}, c_{b_h} \to 0$.

4.6 EFBMF

The EFBMF solution is written as follows:

$$\widehat{U}^{\mathrm{EFB}} = \langle AB \rangle_{p(V|A,B)\phi_{\mathrm{A}}(A;\widehat{c}_a)\phi_{\mathrm{B}}(B;\widehat{c}_b)},$$

where

$$(\widehat{c}_a, \widehat{c}_b) = \operatorname*{argmin}_{(c_a,c_b)} F(V;c_a,c_b).$$

Here $F(V;c_a,c_b)$ is the Bayes free energy (6).

When $V=0,1,2,3$ are observed, the EFB solutions are $0, 0.00, 1.25, 2.58$ ($\widehat{c}_a=\widehat{c}_b \approx 0, 0.0, 1.4$, 2.1), respectively, which were numerically computed.[3] Since $F(V;c_a,c_b) \to \infty$ when $c_a c_b \to \infty$, the

3. The model (1) and the priors (2) and (3) are invariant under the following parameter transformation

$$(\boldsymbol{a}_h, \boldsymbol{b}_h, c_{a_h}, c_{b_h}) \to (s_h^{1/2}\boldsymbol{a}_h, s_h^{-1/2}\boldsymbol{b}_h, s_h^{1/2}c_{a_h}, s_h^{-1/2}c_{b_h})$$

for any $\{s_h \in \mathbb{R}; s_h > 0, h=1,\ldots,H\}$. Here, we fixed the ratio to $c_a/c_b=1$. For $c_a c_b = 10^{-2.00}, 10^{-1.99}, \ldots, 10^{1.00}$, we numerically computed the free energy (6), and chose the minimizer $\widehat{c}_a\widehat{c}_b$, with which the FB solution is computed.

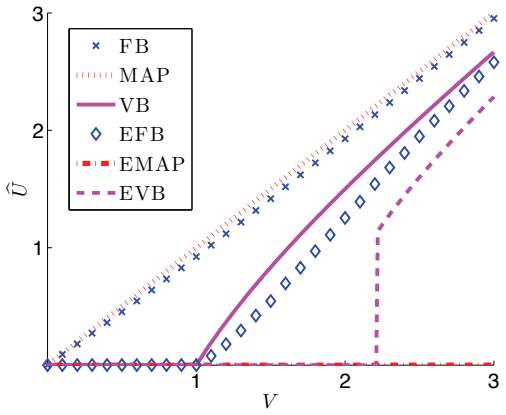


Figure 10: Numerical results of the FBMF solution $\widehat{U}^{\mathrm{FB}}$, the MAPMF solution $\widehat{U}^{\mathrm{MAP}}$, the VBMF solution $\widehat{U}^{\mathrm{VB}}$, the EFBMF solution $\widehat{U}^{\mathrm{EFB}}$, the EMAPMF solution $\widehat{U}^{\mathrm{EMAP}}$, and the EVBMF solution $\widehat{U}^{\mathrm{EVB}}$ when the noise variance is $\sigma^2 = 1$. For MAPMF, VBMF, and FBMF, the hyperparameters are set to $c_a = c_b = 100$ (i.e., almost flat priors).

minimizer of $F(V; c_a, c_b)$ with respect to $\widehat{c}_a$ and $\widehat{c}_b$ are always finite. This implies that EFBMF is more strongly regularized than FBMF with almost flat priors ($c_a c_b \to \infty$).

4.7 Summary

Finally, we summarize the numerical results of all Bayes estimators in Figure 10, including the FBMF solution $\widehat{U}^{\mathrm{FB}}$, the MAPMF solution $\widehat{U}^{\mathrm{MAP}}$, the VBMF solution $\widehat{U}^{\mathrm{VB}}$, the EFBMF solution $\widehat{U}^{\mathrm{EFB}}$, the EMAPMF solution $\widehat{U}^{\mathrm{EMAP}}$, and the EVBMF solution $\widehat{U}^{\mathrm{EVB}}$ when the noise variance is $\sigma^2 = 1$. For MAPMF, VBMF, and FBMF, the hyperparameters are set to $c_a = c_b = 100$ (i.e., almost flat priors). Overall, the solutions satisfy

$$\widehat{U}^{\mathrm{EMAP}} \leq \widehat{U}^{\mathrm{EVB}} \leq \widehat{U}^{\mathrm{EFB}} \leq \widehat{U}^{\mathrm{VB}} \leq \widehat{U}^{\mathrm{FB}} \leq \widehat{U}^{\mathrm{MAP}},$$

which shows the strength of regularization effect of each method.

5. Conclusion

In this paper, we theoretically analyzed the behavior of Bayesian matrix factorization methods. More specifically, in Section 3, we derived *non-asymptotic* bounds of the *maximum a posteriori matrix factorization* (MAPMF) estimator and the *variational Bayesian matrix factorization* (VBMF) estimator. Then we showed that MAPMF consists of the *trace-norm* shrinkage alone, while VBMF consists of the *positive-part James-Stein* (PJS) shrinkage and the trace-norm shrinkage.

An interesting finding was that, while the trace-norm shrinkage does not take effect when the priors are flat, the PJS shrinkage remains activated even with flat priors. The fact that the PJS shrinkage remains activated even with flat priors is induced by the non-identifiability of the MF models, where parameters form equivalent classes. Thus, flat priors in the space of factorized matrices are no longer flat in the space of the target (composite) matrix. Furthermore, simple distributions such

as the Gaussian distribution in the space of the target matrix produce highly complicated *multimodal* distributions in the space of factorized matrices.

We further extended the above analysis to *empirical VBMF* scenarios where hyperparameters included in priors are optimized based on the VB free energy. We showed that the 'strength' of the PJS shrinkage is more than doubled compared with the flat prior cases. We also illustrated the behavior of Bayesian matrix factorization methods using one-dimensional examples in Section 4.

Our theoretical analysis relies on the assumption that a fully observed matrix is provided as a training sample. Thus, our results are not directly applicable to the collaborative filtering scenarios where an observed matrix with missing entries is given. Our important future work is to extend the current analysis so that the behavior of the collaborative filtering algorithms can also be explained. The correspondence between MAPMF and the trace-norm regularization still holds even if missing entries exist. Likewise, we hope to find a relation between VBMF and a regularization term acting on a matrix, which results in the PJS shrinkage if a fully observed matrix is given.

Our analysis also relies on the column-wise independence constraint (14), which was also used in Raiko et al. (2007), on the VB posterior. In principle, the weaker matrix-wise constraint (9) which was used in Lim and Teh (2007) allows non-zero covariances between column vectors, and can achieve a better approximation to the true Bayes posterior. How this affects the performance and when the difference is substantial are to be investigated.

As explained in Appendix A, the PJS estimator dominates (i.e., uniformly better than) the maximum likelihood (ML) estimator in vector estimation. This means that, when $L = 1$, VBMF with (almost) flat priors dominates MLMF. Another interesting future direction is to investigate whether this nice property is inherited to matrix estimation. For matrix estimation ($L > 1$), a variety of estimators which shrink singular values have been proposed (Stein, 1975; Ledoit and Wolf, 2004; Daniels and Kass, 2001), and were shown to possess nice properties under different criteria. Discussing the superiority of such shrinkage estimators including VBMF is interesting future work.

Our investigation revealed a gap between the *fully-Bayesian* (FB) estimator and the VB estimator (see Section 4.7). Figure 10 showed that the VB estimator tends to be strongly regularized. This could cause underfitting and degrade the performance. On the other hand, it is also possible that, in some cases, this stronger regularization could work favorably to suppress overfitting, if we take into account the fact that practitioners do not always choose their prior distributions based on explicit prior information (it is often the case that conjugate priors are chosen only for computational convenience). Further theoretical analysis and empirical investigation are needed to clarify when the stronger regularization of the VB estimator is harmful or helpful.

Tensor factorization is a high-dimensional extension of matrix factorization, which gathers considerable attention recently as a novel data analysis tool (Cichocki et al., 2009). Among various methods, Bayesian methods of tensor factorization have been shown to be promising (Tao et al., 2008; Yu et al., 2008; Hayashi et al., 2009; Chu and Ghahramani, 2009). In our future work, we will elucidate the behavior of tensor factorization methods based on a similar line of discussion to the current work.

Acknowledgments

We would like to thank anonymous reviewers for helpful comments and suggestions for future work. Masashi Sugiyama thanks the support from the FIRST program.

Appendix A. James-Stein Shrinkage Estimator

Here, we briefly introduce the *James-Stein* (JS) shrinkage estimator and its variants (James and Stein, 1961; Efron and Morris, 1973).

Let us consider the problem of estimating the mean $\boldsymbol{\mu}$ ($\in \mathbb{R}^d$) of the d-dimensional Gaussian distribution $\mathcal{N}(\boldsymbol{\mu}, \sigma^2 I_d)$ from its independent and identically distributed samples

$$\mathcal{X}^n = \{\boldsymbol{x}_i \in \mathbb{R}^d \mid i = 1, \ldots, n\}.$$

We measure the generalization error (or the risk) of an estimator $\widehat{\boldsymbol{\mu}}$ by the expected squared error:

$$\mathbb{E}\|\widehat{\boldsymbol{\mu}} - \boldsymbol{\mu}\|^2,$$

where $\mathbb{E}$ denotes the expectation over the samples $\mathcal{X}^n$.

An estimator $\widehat{\boldsymbol{\mu}}$ is said to *dominate* another estimator $\widehat{\boldsymbol{\mu}}'$ if

$$\mathbb{E}\|\widehat{\boldsymbol{\mu}} - \boldsymbol{\mu}\|^2 \leq \mathbb{E}\|\widehat{\boldsymbol{\mu}}' - \boldsymbol{\mu}\|^2 \text{ for all } \boldsymbol{\mu},$$

and

$$\mathbb{E}\|\widehat{\boldsymbol{\mu}} - \boldsymbol{\mu}\|^2 < \mathbb{E}\|\widehat{\boldsymbol{\mu}}' - \boldsymbol{\mu}\|^2 \text{ for some } \boldsymbol{\mu}.$$

An estimator is said to be *admissible* if no estimator dominates it.

Stein (1956) proved the inadmissibility of the maximum likelihood (ML) estimator (or equivalently the least-squares estimator),

$$\widehat{\boldsymbol{\mu}}^{\mathrm{ML}} = \frac{1}{n}\sum_{i=1}^{n} \boldsymbol{x}_i,$$

when $d \geq 3$. This discovery was surprising because the ML estimator had been believed to be a *good* estimator. James and Stein (1961) subsequently proposed the JS shrinkage estimator $\widehat{\boldsymbol{\mu}}^{\mathrm{JS}}$, which was proved to dominate the ML estimator:

$$\widehat{\boldsymbol{\mu}}^{\mathrm{JS}} = \left(1 - \frac{\chi\sigma^2}{n\|\widehat{\boldsymbol{\mu}}^{\mathrm{ML}}\|^2}\right)\widehat{\boldsymbol{\mu}}^{\mathrm{ML}}, \tag{59}$$

where $\chi = d - 2$. Efron and Morris (1973) showed that the JS shrinkage estimator can be derived as an empirical Bayes estimator. In the current paper, we refer to all estimators of the form (59) with arbitrary $\chi > 0$ as the JS shrinkage estimators.

The *positive-part James-Stein* (PJS) shrinkage estimator, which was shown to dominate the JS estimator, is given as follows (Baranchik, 1964):

$$\widehat{\boldsymbol{\mu}}^{\mathrm{PJS}} = \max\left\{0, \left(1 - \frac{\chi\sigma^2}{n\|\widehat{\boldsymbol{\mu}}^{\mathrm{ML}}\|^2}\right)\widehat{\boldsymbol{\mu}}^{\mathrm{ML}}\right\}.$$

Note that the PJS estimator itself is also inadmissible, following the fact that admissible estimators are necessarily smooth (Lehmann, 1983). Indeed, there exist several estimators that dominate the PJS estimator (Strawderman, 1971; Guo and Pal, 1992; Shao and Strawderman, 1994). However, their improvement is rather minor, and they are not as simple as the PJS estimator. Moreover, none of these estimators is admissible.

Appendix B. Proof of Theorem 1

The MAP estimator is defined as the minimizer of the negative log (17) of the Bayes posterior. Let us double Equation (17) and neglect some constant terms which are irrelevant to its minimization with respect to $\{\boldsymbol{a}_h, \boldsymbol{b}_h\}_{h=1}^H$:

$$\mathcal{L}^{\mathrm{MAP}}(\{\boldsymbol{a}_h, \boldsymbol{b}_h\}_{h=1}^H) = \sum_{h=1}^H \left(\frac{\|\boldsymbol{a}_h\|^2}{c_{a_h}^2} + \frac{\|\boldsymbol{b}_h\|^2}{c_{b_h}^2} \right) + \frac{1}{\sigma^2} \left\| V - \sum_{h=1}^H \boldsymbol{b}_h \boldsymbol{a}_h^\top \right\|_{\mathrm{Fro}}^2. \tag{60}$$

We use the following lemma (its proof is given in Appendix G.1):

Lemma 7 *For arbitrary matrices $A \in \mathbb{R}^{M\times H}$ and $B \in \mathbb{R}^{L\times H}$, let*

$$BA^\top = \Omega_L \Gamma \Omega_R^\top$$

be the singular value decomposition of the product $BA^\top$, where $\Gamma = diag(\widehat{\gamma}_1, \ldots, \widehat{\gamma}_H)$ ($\{\widehat{\gamma}_h\}$ are in non-increasing order). Remember that $\{c_{a_h} c_{b_h}\}$, where $C_A = diag(c_{a_1}^2, \ldots, c_{a_H}^2)$ and $C_B = diag(c_{b_1}^2, \ldots, c_{b_H}^2)$ are positive-definite, are also arranged in non-increasing order. Then, it holds that

$$\mathrm{tr}(AC_A^{-1}A^\top) + \mathrm{tr}(BC_B^{-1}B^\top) \geq \sum_{h=1}^H \frac{2\widehat{\gamma}_h}{c_{a_h} c_{b_h}}. \tag{61}$$

Using Lemma 7, we obtain the following lemma (its proof is given in Appendix G.2):

Lemma 8 *The MAP solution $\widehat{U}^{\mathrm{MAP}}$ is written in the following form:*

$$\widehat{U}^{\mathrm{MAP}} = \widehat{B}\widehat{A}^\top = \sum_{h=1}^H \widehat{\gamma}_h \boldsymbol{\omega}_{b_h} \boldsymbol{\omega}_{a_h}^\top. \tag{62}$$

There exists at least one minimizer that can be written as

$$\boldsymbol{a}_h = a_h \boldsymbol{\omega}_{a_h}, \tag{63}$$
$$\boldsymbol{b}_h = b_h \boldsymbol{\omega}_{b_h}, \tag{64}$$

where $\{a_h, b_h\}$ are scalars such that

$$\widehat{\gamma}_h = a_h b_h \geq 0.$$

Lemma 8 implies that the minimization of Equation (60) amounts to a re-weighted singular value decomposition.

We can also prove the following lemma (its proof is given in Appendix G.3):

Lemma 9 *Let $\{\mathcal{H}_k; k = 1, \ldots, K(\leq H)\}$ be the partition of $\{1, \ldots, H\}$ such that $c_{a_h} c_{b_h} = c_{a_{h'}} c_{b_{h'}}$ if and only if h and h' belong to the same group (i.e., $^\exists k$ such that $h, h' \in \mathcal{H}_k$). Suppose that $(\widehat{A}, \widehat{B})$ is a MAP solution. Then,*

$$\widehat{A}' = \widehat{A}\Theta^\top,$$
$$\widehat{B}' = \widehat{B}\Theta^{-1},$$

is also a MAP solution, for any Θ *defined by*

$$\begin{aligned}\Theta &= C_A^{1/2}\Xi C_A^{-1/2} \\ &= C_B^{-1/2}\Xi C_B^{1/2}.\end{aligned}$$

Here, Ξ *is a block diagonal matrix such that the blocks are organized based on the partition* $\{\mathcal{H}_k\}$*, and each block consists of an arbitrary orthogonal matrix.*

Lemma 9 states that non-orthogonal solutions (i.e., $\{\boldsymbol{a}_h\}$, as well as $\{\boldsymbol{b}_h\}$, are not orthogonal with each other) can exist. However, Lemma 8 guarantees that any non-orthogonal solution has its *equivalent* orthogonal solution, which is written in the form of Equations (63) and (64). Here, by *equivalent* solution, we denote a solution resulting in the identical $\widehat{U}^{\mathrm{MAP}}$ in Equation (62). Since we are interested in finding $\widehat{U}^{\mathrm{MAP}}$, we regard the orthogonal solution as the representative of the *equivalent* solutions, and focus on it.

The expression (63) and (64) allows us to decompose the minimization of Equation (60) into the minimization of the following H separate objective functions: for $h = 1, \ldots, H$,

$$\mathcal{L}_h^{\mathrm{MAP}}(a_h, b_h) = \left(\frac{a_h^2}{c_{a_h}^2} + \frac{b_h^2}{c_{b_h}^2}\right) + \frac{1}{\sigma^2}\left(\gamma_h - a_h b_h\right)^2.$$

This can be written as

$$\mathcal{L}_h^{\mathrm{MAP}}(a_h, b_h) = \frac{b_h^2}{c_{a_h}^2}\left(\frac{a_h}{b_h} - \frac{c_{a_h}}{c_{b_h}}\right)^2 + \frac{1}{\sigma^2}\left(a_h b_h - \left(\gamma_h - \frac{\sigma^2}{c_{a_h}c_{b_h}}\right)\right)^2 + \left(\frac{2\gamma_h}{c_{a_h}c_{b_h}} - \frac{\sigma^2}{c_{a_h}^2 c_{b_h}^2}\right). \quad (65)$$

The third term is constant with respect to a_h and b_h. The first nonnegative term vanishes by setting the ratio a_h/b_h to

$$\frac{a_h}{b_h} = \frac{c_{a_h}}{c_{b_h}} \quad (\text{or } b_h = 0). \quad (66)$$

Minimizing the second term in Equation (65), which is quadratic with respect to the product $a_h b_h$ (≥ 0), we can easily obtain Equation (21), which completes the proof. ■

Appendix C. Proof of Theorem 2, Theorem 3, and Theorem 4

We denote by $\mathbb{R}_+^d$ the set of the d-dimensional vectors with non-negative elements, by $\mathbb{R}_{++}^d$ the set of the d-dimensional vectors with positive elements, by $\mathbb{S}_+^d$ the set of $d \times d$ positive semi-definite symmetric matrices, and by $\mathbb{S}_{++}^d$ the set of $d \times d$ positive definite symmetric matrices. The VB free energy to be minimized can be expressed as Equation (39). Neglecting constant terms, we define

the objective function as follows:

$$\begin{aligned}\mathcal{L}^{\mathrm{VB}}(\{\boldsymbol{a}_h,\boldsymbol{b}_h,\Sigma_{a_h},\Sigma_{b_h}\}) &= 2F_{\mathrm{VB}}(r|V,\{c_{a_h}^2,c_{b_h}^2\})+\mathrm{Const.}\\ &= \sum_{h=1}^{H}\left(-\log|\Sigma_{a_h}|+\frac{\|\boldsymbol{\mu}_{a_h}\|^2+\mathrm{tr}(\Sigma_{a_h})}{c_{a_h}^2}-\log|\Sigma_{b_h}|+\frac{\|\boldsymbol{\mu}_{b_h}\|^2+\mathrm{tr}(\Sigma_{b_h})}{c_{b_h}^2}\right)\\ &\quad+\frac{1}{\sigma^2}\left\|V-\sum_{h=1}^{H}\boldsymbol{\mu}_{b_h}\boldsymbol{\mu}_{a_h}^{\top}\right\|_{\mathrm{Fro}}^2\\ &\quad+\frac{1}{\sigma^2}\sum_{h=1}^{H}\left(\|\boldsymbol{\mu}_{a_h}\|^2\mathrm{tr}(\Sigma_{b_h})+\mathrm{tr}(\Sigma_{a_h})\|\boldsymbol{\mu}_{b_h}\|^2+\mathrm{tr}(\Sigma_{a_h})\mathrm{tr}(\Sigma_{b_h})\right). \qquad (67)\end{aligned}$$

We solve the following problem:

$$\begin{aligned}&\text{Given } (c_{a_h}^2,c_{b_h}^2)\in\mathbb{R}_{++}^2(^\forall h=1,\dots,H),\sigma^2\in\mathbb{R}_{++},\\ &\quad\min\quad \mathcal{L}^{\mathrm{VB}}(\{\boldsymbol{\mu}_{a_h},\boldsymbol{\mu}_{b_h},\Sigma_{a_h},\Sigma_{b_h};h=1,\dots,H\}) \qquad (68)\\ &\quad\text{s.t. } \boldsymbol{\mu}_{a_h}\in\mathbb{R}^M,\boldsymbol{\mu}_{b_h}\in\mathbb{R}^L,\Sigma_{a_h}\in\mathbb{S}_{++}^M,\Sigma_{b_h}\in\mathbb{S}_{++}^L(^\forall h=1,\dots,H). \qquad (69)\end{aligned}$$

First, we have the following lemma (its proof is given in Appendix G.4):

Lemma 10 *At least one minimizer always exists, and any minimizer is a stationary point.*

Given fixed $\{(\Sigma_{a_h},\Sigma_{b_h})\}$, the objective function (67) is of the same form as Equation (60) if we replace $\{(c_{a_h}^2,c_{b_h}^2)\}$ in Equation (60) with $\{(c_{a_h}'^2,c_{b_h}'^2)\}$ defined by

$$c_{a_h}'^2=\left(\frac{1}{c_{a_h}^2}+\frac{\mathrm{tr}(\Sigma_{b_h})}{\sigma^2}\right)^{-1}, \qquad (70)$$

$$c_{b_h}'^2=\left(\frac{1}{c_{b_h}^2}+\frac{\mathrm{tr}(\Sigma_{a_h})}{\sigma^2}\right)^{-1}. \qquad (71)$$

Therefore, Lemma 8 implies that the minimizers of $\boldsymbol{\mu}_{a_h}$ and $\boldsymbol{\mu}_{b_h}$ are parallel (or zero) to the singular vectors of V associated with the H largest singular values.[4] On the other hand, Lemma 10 guarantees that Equations (23)–(26), which together form a necessary and sufficient condition to be a stationary point, hold at any minimizer. Equations (25) and (26) suggest that Σ_{a_h} and Σ_{b_h} are proportional to I_M and I_L, respectively. Accordingly, any minimizer can be written as $\boldsymbol{\mu}_{a_h}=\mu_{a_h}\boldsymbol{\omega}_{a_h}$, $\boldsymbol{\mu}_{b_h}=\mu_{b_h}\boldsymbol{\omega}_{b_h}$, $\Sigma_{a_h}=\sigma_{a_h}^2I_M$, and $\Sigma_{b_h}=\sigma_{b_h}^2I_L$, where $\mu_{a_h},\mu_{b_h},\sigma_{a_h}^2$, and $\sigma_{b_h}^2$ are scalars. This allows us to decompose the problem (68) into H separate problems: for $h=1,\dots,H$,

$$\begin{aligned}&\text{Given } (c_{a_h}^2,c_{b_h}^2)\in\mathbb{R}_{++}^2,\sigma^2\in\mathbb{R}_{++},\\ &\quad\min\quad \mathcal{L}_h^{\mathrm{VB}}(\mu_{a_h},\mu_{b_h},\sigma_{a_h}^2,\sigma_{b_h}^2)\\ &\quad\text{s.t. } (\mu_{a_h},\mu_{b_h})\in\mathbb{R}^2,(\sigma_{a_h}^2,\sigma_{b_h}^2)\in\mathbb{R}_{++}^2, \qquad (72)\end{aligned}$$

4. As in Appendix B, we regard the orthogonal solution of the form (63) and (64) as the representative of the *equivalent* solutions, and focus on it. See Lemma 9 and its subsequent paragraph.

where

$$\mathcal{L}_h^{\mathrm{VB}}(\mu_{a_h},\mu_{b_h},\sigma_{a_h}^2,\sigma_{b_h}^2) = -M\log\sigma_{a_h}^2 + \frac{\mu_{a_h}^2+M\sigma_{a_h}^2}{c_{a_h}^2} - L\log\sigma_{b_h}^2 + \frac{\mu_{b_h}^2+L\sigma_{b_h}^2}{c_{b_h}^2} - \frac{2}{\sigma^2}\gamma_h\mu_{a_h}\mu_{b_h} + \frac{1}{\sigma^2}\left(\mu_{a_h}^2+M\sigma_{a_h}^2\right)\left(\mu_{b_h}^2+L\sigma_{b_h}^2\right). \tag{73}$$

Moreover, the necessary and sufficient condition (23)–(26) is reduced to

$$\mu_{a_h} = \frac{1}{\sigma^2}\sigma_{a_h}^2\gamma_h\mu_{b_h}, \tag{74}$$

$$\mu_{b_h} = \frac{1}{\sigma^2}\sigma_{b_h}^2\gamma_h\mu_{a_h}, \tag{75}$$

$$\sigma_{a_h}^2 = \sigma^2\left(\mu_{b_h}^2+L\sigma_{b_h}^2+\frac{\sigma^2}{c_{a_h}^2}\right)^{-1}, \tag{76}$$

$$\sigma_{b_h}^2 = \sigma^2\left(\mu_{a_h}^2+M\sigma_{a_h}^2+\frac{\sigma^2}{c_{b_h}^2}\right)^{-1}. \tag{77}$$

We use the following definition:

$$\widehat{\gamma}_h = \mu_{a_h}\mu_{b_h}, \tag{78}$$

Note that Equations (27) and (78) imply that the VB solution $\widehat{U}^{\mathrm{VB}}$ can be expressed as

$$\widehat{U}^{\mathrm{VB}} = \sum_{h=1}^{H}\widehat{\gamma}_h\boldsymbol{\omega}_{b_h}\boldsymbol{\omega}_{a_h}^{\top}.$$

Equations (74) and (75) imply that μ_{a_h} and μ_{b_h} have the same sign (or both are zero), since $\gamma_h \geq 0$ by definition. Therefore, Equation (78) yields

$$\widehat{\gamma}_h \geq 0.$$

In the following, we investigate two types of stationary points. We say that $(\mu_{a_h},\mu_{b_h},\sigma_{a_h}^2,\sigma_{b_h}^2) = (\mathring{\mu}_{a_h},\mathring{\mu}_{b_h},\mathring{\sigma}_{a_h}^2,\mathring{\sigma}_{b_h}^2)$ is a *null* stationary point if it is a stationary point resulting in the null output ($\widehat{\gamma}_h = \mathring{\mu}_{a_h}\mathring{\mu}_{b_h} = 0$). On the other hand, we say that $(\mu_{a_h},\mu_{b_h},\sigma_{a_h}^2,\sigma_{b_h}^2) = (\breve{\mu}_{a_h},\breve{\mu}_{b_h},\breve{\sigma}_{a_h}^2,\breve{\sigma}_{b_h}^2)$ is a *positive* stationary point if it is a stationary point resulting in a positive output ($\widehat{\gamma}_h = \breve{\mu}_{a_h}\breve{\mu}_{b_h} > 0$).

Let

$$\widehat{\eta}_h = \sqrt{\left(\mu_{a_h}^2+\frac{\sigma^2}{c_{b_h}^2}\right)\left(\mu_{b_h}^2+\frac{\sigma^2}{c_{a_h}^2}\right)}. \tag{79}$$

The explicit form of the *null* stationary point is derived as follows (its proof is given in Appendix G.5):

Lemma 11 *The unique* null *stationary point always exists, and it is given by*

$$\mathring{\mu}_{a_h} = 0, \tag{80}$$

$$\mathring{\mu}_{b_h} = 0, \tag{81}$$

$$\mathring{\sigma}^2_{a_h} = \frac{c_{a_h}}{2Mc_{b_h}} \left\{ -\left(\frac{\sigma^2}{c_{a_h}c_{b_h}} - c_{a_h}c_{b_h}(M-L) \right) + \sqrt{\left(\frac{\sigma^2}{c_{a_h}c_{b_h}} - c_{a_h}c_{b_h}(M-L) \right)^2 + 4M\sigma^2} \right\}, \tag{82}$$

$$\mathring{\sigma}^2_{b_h} = \frac{c_{b_h}}{2Lc_{a_h}} \left\{ -\left(\frac{\sigma^2}{c_{a_h}c_{b_h}} + c_{a_h}c_{b_h}(M-L) \right) + \sqrt{\left(\frac{\sigma^2}{c_{a_h}c_{b_h}} + c_{a_h}c_{b_h}(M-L) \right)^2 + 4L\sigma^2} \right\}. \tag{83}$$

Next, we investigate the *positive* stationary points, assuming that $\mu_{a_h} \neq 0, \mu_{b_h} \neq 0$. Equations (74) and (75) suggest that no *positive* stationary point exists when $\gamma_h = 0$. Below, we focus on the case when $\gamma_h > 0$. Let

$$\widehat{\delta}_h = \frac{\mu_{a_h}}{\mu_{b_h}}. \tag{84}$$

We can transform the necessary and sufficient condition (74)–(77) as follows (its proof is given in Appendix G.6):

Lemma 12 *No* positive *stationary point exists if*

$$\gamma_h^2 \leq \sigma^2 M.$$

When

$$\gamma_h^2 > \sigma^2 M, \tag{85}$$

at least one positive *stationary point exists if and only if the following five equations*

$$\widehat{\eta}_h = \sqrt{\left(\widehat{\gamma}_h \widehat{\delta}_h + \frac{\sigma^2}{c_{b_h}^2} \right) \left(\widehat{\gamma}_h \widehat{\delta}_h^{-1} + \frac{\sigma^2}{c_{a_h}^2} \right)}, \tag{86}$$

$$\widehat{\eta}_h^2 = \left(1 - \frac{\sigma^2 L}{\gamma_h^2} \right) \left(1 - \frac{\sigma^2 M}{\gamma_h^2} \right) \gamma_h^2, \tag{87}$$

$$\sigma^2 \left(\frac{M\widehat{\delta}_h}{c_{a_h}^2} - \frac{L}{c_{b_h}^2 \widehat{\delta}_h} \right) = (M-L)(\gamma_h - \widehat{\gamma}_h), \tag{88}$$

$$\sigma^2_{a_h} = \frac{-\left(\widehat{\eta}_h^2 - \sigma^2(M-L)\right) + \sqrt{(\widehat{\eta}_h^2 - \sigma^2(M-L))^2 + 4M\sigma^2\widehat{\eta}_h^2}}{2M(\widehat{\gamma}_h \widehat{\delta}_h^{-1} + \sigma^2 c_{a_h}^{-2})}, \tag{89}$$

$$\sigma^2_{b_h} = \frac{-\left(\widehat{\eta}_h^2 + \sigma^2(M-L)\right) + \sqrt{(\widehat{\eta}_h^2 + \sigma^2(M-L))^2 + 4L\sigma^2\widehat{\eta}_h^2}}{2L(\widehat{\gamma}_h \widehat{\delta}_h + \sigma^2 c_{b_h}^{-2})} \tag{90}$$

have a solution with respect to $(\widehat{\gamma}_h, \widehat{\delta}_h, \sigma^2_{a_h}, \sigma^2_{b_h}, \widehat{\eta}_h)$ *such that*

$$(\widehat{\gamma}_h, \widehat{\delta}_h, \sigma^2_{a_h}, \sigma^2_{b_h}, \widehat{\eta}_h) \in \mathbb{R}^5_{++}. \tag{91}$$

When a solution exists, the corresponding pair of positive *stationary points*

$$(\mu_{a_h}, \mu_{b_h}, \sigma^2_{a_h}, \sigma^2_{b_h}) = (\pm\sqrt{\widehat{\gamma}_h \widehat{\delta}_h}, \pm\sqrt{\widehat{\gamma}_h \widehat{\delta}_h^{-1}}, \sigma^2_{a_h}, \sigma^2_{b_h}) \tag{92}$$

exist.

Then we obtain a simpler necessary and sufficient condition for existence of *positive* stationary points (its proof is given in Appendix G.7):

Lemma 13 *At least one* positive *stationary point exists if and only if Equation* (85) *holds and*

$$\widehat{\gamma}_h^2 + q_1(\widehat{\gamma}_h) \cdot \widehat{\gamma}_h + q_0 = 0 \tag{93}$$

has any positive real solution with respect to $\widehat{\gamma}_h$, *where*

$$q_1(\widehat{\gamma}_h) = \frac{-(M-L)^2(\gamma_h - \widehat{\gamma}_h) + (L+M)\sqrt{(M-L)^2(\gamma_h - \widehat{\gamma}_h)^2 + \frac{4\sigma^4 LM}{c^2_{a_h} c^2_{b_h}}}}{2LM}, \tag{94}$$

$$q_0 = \frac{\sigma^4}{c^2_{a_h} c^2_{b_h}} - \left(1 - \frac{\sigma^2 L}{\gamma_h^2}\right)\left(1 - \frac{\sigma^2 M}{\gamma_h^2}\right)\gamma_h^2. \tag{95}$$

Any positive solution $\widehat{\gamma}_h$ *satisfies*

$$0 < \widehat{\gamma}_h < \gamma_h. \tag{96}$$

Equation (96) guarantees that

$$q_1(\widehat{\gamma}_h) > 0.$$

Recall that a quadratic equation

$$\widehat{\gamma}^2 + q_1 \widehat{\gamma} + q_0 = 0 \text{ for } q_1 > 0 \tag{97}$$

has only one positive solution when $q_0 < 0$ (otherwise no positive solution exists) (see Figure 11). The condition for the negativity of Equation (95) leads to the following lemma:

Lemma 14 *At least one* positive *stationary point exists if and only if*

$$\gamma_h^2 > \sigma^2 M \qquad \textit{and} \qquad \sqrt{\left(1 - \frac{\sigma^2 L}{\gamma_h^2}\right)\left(1 - \frac{\sigma^2 M}{\gamma_h^2}\right)}\gamma_h - \frac{\sigma^2}{c_{a_h} c_{b_h}} > 0. \tag{98}$$

The following lemma also holds (its proof is given in Appendix G.8):

Lemma 15 *Equation* (98) *holds if and only if*

$$\gamma_h > \widetilde{\gamma}_h^{\mathrm{VB}},$$

where $\widetilde{\gamma}_h^{VB}$ *is defined by Equation* (30).

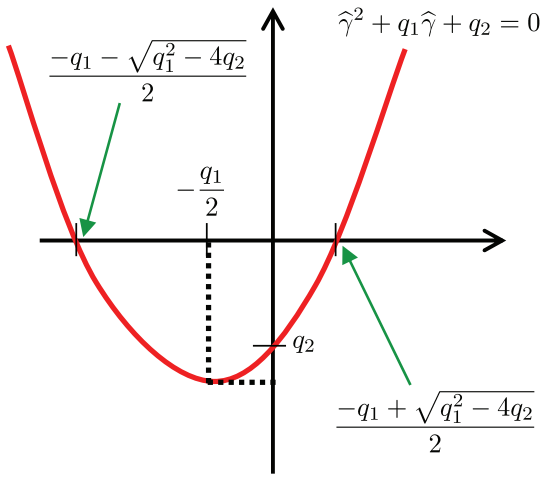


Figure 11: Quadratic function $f(\widehat{\gamma}) = \widehat{\gamma}^2 + q_1\widehat{\gamma} + q_0$, where $q_1 > 0$ and $q_0 < 0$.

Combining Lemma 10 and Lemma 14 together, we conclude that the *null* stationary point (which always exists) is the minimizer when Equation (98) does not hold. On the other hand, when a *positive* stationary point exists, we have to clarify which stationary point is the minimum. The following lemma holds (its proof is given in Appendix G.9).

Lemma 16 *The* null *stationary point is a saddle point when any* positive *stationary point exists*.

Combining Lemma 10, Lemma 14, and Lemma 16 together, we obtain the following lemma:

Lemma 17 *When Equation* (98) *holds, the minimizers consist of* positive *stationary points*. *Otherwise, the minimizer is the* null *stationary point*.

Combining Lemma 15 and Lemma 17 completes the proof of Theorem 3.

Finally, we derive bounds of the *positive* stationary points (its proof is given in Appendix G.10):

Lemma 18 *Equations* (28) *and* (31) *hold for any* positive *stationary point*.

Combining Lemma 17 and Lemma 18 completes the proof of Theorem 2 and Theorem 4. ■

Appendix D. Proof of Corollary 1

From Equations (78) and (84), we have $\mu_{a_h}^2 = \widehat{\gamma}_h\widehat{\delta}_h$ and $\mu_{b_h}^2 = \widehat{\gamma}_h/\widehat{\delta}_h$. When $L = M$, $\widehat{\gamma}_h$ is expressed analytically by Equation (32) and $\widehat{\delta}_h = c_a/c_b$ follows from Equation (88). From these, we have Equations (33) and (34).

When $L = M$, Equations (137) and (138) are reduced to

$$\sigma_{a_h}^2 = \frac{\widehat{\eta}_h \sqrt{\widehat{\eta}_h^2 + 4\sigma^2 M} - \widehat{\eta}_h^2}{2M\left(\mu_{b_h}^2 + \sigma^2 / c_{a_h}^2\right)}, \tag{99}$$

$$\sigma_{b_h}^2 = \frac{\widehat{\eta}_h \sqrt{\widehat{\eta}_h^2 + 4\sigma^2 M} - \widehat{\eta}_h^2}{2M\left(\mu_{a_h}^2 + \sigma^2 / c_{b_h}^2\right)}. \tag{100}$$

Substituting Equation (79) into Equations (99) and (100) and using Equations (33) and (34) give Equations (35) and (36). Because of the symmetry of the objective function (73), the two *positive* stationary points (33)–(36) give the same objective value, which completes the proof. ■

Note that *equivalent* nonorthogonal (with respect to $\{\boldsymbol{\mu}_{a_h}\}$, as well as $\{\boldsymbol{\mu}_{b_h}\}$) solutions may exist in principle. We neglect such solutions, because they almost surely do not exist; Equations (70), (71), (35), and (36) together imply that any pair $\{(h,h'); h \neq h'\}$ such that $\max(\widehat{\gamma}_h^{\mathrm{VB}}, \widehat{\gamma}_{h'}^{\mathrm{VB}}) > 0$ and $c'_{a_h} c'_{b_h} = c'_{a_{h'}} c'_{b_{h'}}$ can exist only when $c_{a_h} c_{b_h} = c_{a_{h'}} c_{b_{h'}}$ and $\gamma_h = \gamma_{h'}$ (i.e., two singular values of a random matrix coincide with each other).

Appendix E. Proof of Theorem 5 and Theorem 6

The EVB estimator is the minimizer of the VB free energy (39). Neglecting constant terms, we define the objective function as follows:

$$\begin{aligned}
\mathcal{L}^{\mathrm{EVB}}(\{\boldsymbol{a}_h, \boldsymbol{b}_h, \Sigma_{a_h}, \Sigma_{b_h}, c_{a_h}^2, c_{b_h}^2\}) &= 2F_{\mathrm{VB}}(r|V, \{c_{a_h}^2, c_{b_h}^2\}) + \mathrm{Const.} \\
&= \sum_{h=1}^{H} \left(\log \frac{c_{a_h}^{2M}}{|\Sigma_{a_h}|} + \frac{\|\boldsymbol{\mu}_{a_h}\|^2 + \mathrm{tr}(\Sigma_{a_h})}{c_{a_h}^2} + \log \frac{c_{b_h}^2}{|\Sigma_{b_h}|} + \frac{\|\boldsymbol{\mu}_{b_h}\|^2 + \mathrm{tr}(\Sigma_{b_h})}{c_{b_h}^2} \right) \\
&\quad + \frac{1}{\sigma^2} \left\| V - \sum_{h=1}^{H} \boldsymbol{\mu}_{b_h} \boldsymbol{\mu}_{a_h}^{\top} \right\|_{\mathrm{Fro}}^2 \\
&\quad + \frac{1}{\sigma^2} \sum_{h=1}^{H} \left(\|\boldsymbol{\mu}_{a_h}\|^2 \mathrm{tr}(\Sigma_{b_h}) + \mathrm{tr}(\Sigma_{a_h}) \|\boldsymbol{\mu}_{b_h}\|^2 + \mathrm{tr}(\Sigma_{a_h}) \mathrm{tr}(\Sigma_{b_h}) \right).
\end{aligned}$$

We solve the following problem:

$$\begin{aligned}
&\text{Given } \sigma^2 \in \mathbb{R}_{++}, \\
&\quad \min \quad \mathcal{L}^{\mathrm{EVB}}(\{\boldsymbol{\mu}_{a_h}, \boldsymbol{\mu}_{b_h}, \Sigma_{a_h}, \Sigma_{b_h}, c_{a_h}^2, c_{b_h}^2; h = 1, \ldots, H\}) && (101) \\
&\quad \text{s.t. } \boldsymbol{\mu}_{a_h} \in \mathbb{R}^M, \boldsymbol{\mu}_{b_h} \in \mathbb{R}^L, \Sigma_{a_h} \in \mathbb{S}_{++}^M, \Sigma_{b_h} \in \mathbb{S}_{++}^L, (c_{a_h}^2, c_{b_h}^2) \in \mathbb{R}_{++}^2 (^\forall h = 1, \ldots, H). && (102)
\end{aligned}$$

Define a partial minimization problem of (101) with fixed $\{c_{a_h}^2, c_{b_h}^2\}$:

$$\begin{aligned}
\tilde{\mathcal{L}}^{\mathrm{EVB}}(\{c_{a_h}^2, c_{b_h}^2\}) = \min_{(\boldsymbol{\mu}_{a_h}, \boldsymbol{\mu}_{b_h}, \Sigma_{a_h}, \Sigma_{b_h})} & \mathcal{L}_h^{\mathrm{EVB}}(\{\boldsymbol{\mu}_{a_h}, \boldsymbol{\mu}_{b_h}, \Sigma_{a_h}, \Sigma_{b_h}\}; \{c_{a_h}^2, c_{b_h}^2\}) && (103) \\
\text{s.t. } \boldsymbol{\mu}_{a_h} \in \mathbb{R}^M, \boldsymbol{\mu}_{b_h} \in \mathbb{R}^L, \Sigma_{a_h} \in \mathbb{S}_{++}^M, \Sigma_{b_h} \in \mathbb{S}_{++}^L & (^\forall h = 1, \ldots, H).
\end{aligned}$$

This is identical to the VB estimation problem (68), and therefore, we can use the results proved in Appendix C. According to Lemma 10, at least one solution of the problem (103) exists. Therefore, the following problem is equivalent to the original problem (101):

$$\min_{\{c_{a_h}^2, c_{b_h}^2\}} \tilde{\mathcal{L}}^{\mathrm{EVB}}(\{c_{a_h}^2, c_{b_h}^2\}) \tag{104}$$
$$\text{s.t. } (c_{a_h}^2, c_{b_h}^2) \in \mathbb{R}_{++}^2 (^\forall h = 1, \ldots, H).$$

We have proved in Appendix C that any solution of the problem (103) can be written as $\boldsymbol{\mu}_{a_h} = \mu_{a_h}\boldsymbol{\omega}_{a_h}$, $\boldsymbol{\mu}_{b_h} = \mu_{b_h}\boldsymbol{\omega}_{b_h}$, $\Sigma_{a_h} = \sigma_{a_h}^2 I_M$, and $\Sigma_{b_h} = \sigma_{b_h}^2 I_L$, where μ_{a_h}, μ_{b_h}, $\sigma_{a_h}^2$, and $\sigma_{b_h}^2$ are scalars. This allows us to decompose the problem (101) into H separate problems: for $h = 1, \ldots, H$,

$$\begin{aligned} &\text{Given } \sigma^2 \in \mathbb{R}_{++}, \\ &\quad \min \quad \mathcal{L}_h^{\mathrm{EVB}}(\mu_{a_h}, \mu_{b_h}, \sigma_{a_h}^2, \sigma_{b_h}^2, c_{a_h}^2, c_{b_h}^2) \\ &\quad \text{s.t. } (\mu_{a_h}, \mu_{b_h}) \in \mathbb{R}^2, (\sigma_{a_h}^2, \sigma_{b_h}^2) \in \mathbb{R}_{++}^2, (c_{a_h}^2, c_{b_h}^2) \in \mathbb{R}_{++}^2, \end{aligned} \tag{105}$$

where

$$\begin{aligned} \mathcal{L}_h^{\mathrm{EVB}}(\mu_{a_h}, \mu_{b_h}, \sigma_{a_h}^2, \sigma_{b_h}^2, c_{a_h}^2, c_{b_h}^2) = M \log \frac{c_{a_h}^2}{\sigma_{a_h}^2} + \frac{\mu_{a_h}^2 + M\sigma_{a_h}^2}{c_{a_h}^2} + L \log \frac{c_{b_h}^2}{\sigma_{b_h}^2} + \frac{\mu_{b_h}^2 + L\sigma_{b_h}^2}{c_{b_h}^2} \\ - \frac{2}{\sigma^2}\gamma_h \mu_{a_h}\mu_{b_h} + \frac{1}{\sigma^2}\left(\mu_{a_h}^2 + M\sigma_{a_h}^2\right)\left(\mu_{b_h}^2 + L\sigma_{b_h}^2\right). \end{aligned} \tag{106}$$

Let

$$\kappa = \begin{cases} \sigma^2 \left(\sqrt{\left(1 - \frac{\sigma^2 L}{\gamma_h^2}\right)\left(1 - \frac{\sigma^2 M}{\gamma_h^2}\right)}\gamma_h \right)^{-1} & \text{if } \gamma_h > \sqrt{\sigma^2 M}, \\ \infty & \textit{otherwise}. \end{cases}$$

We divide the domain (105) into two regions (see Figure 12):

$$\mathring{\mathcal{R}} = \left\{ (\mu_{a_h}, \mu_{b_h}, \sigma_{a_h}^2, \sigma_{b_h}^2, c_{a_h}^2, c_{b_h}^2) \in \mathbb{R}^2 \times \mathbb{R}_{++}^2 \times \mathbb{R}_{++}^2 ; c_{a_h} c_{b_h} \le \kappa \right\}, \tag{107}$$
$$\breve{\mathcal{R}} = \left\{ (\mu_{a_h}, \mu_{b_h}, \sigma_{a_h}^2, \sigma_{b_h}^2, c_{a_h}^2, c_{b_h}^2) \in \mathbb{R}^2 \times \mathbb{R}_{++}^2 \times \mathbb{R}_{++}^2 ; c_{a_h} c_{b_h} > \kappa \right\}. \tag{108}$$

Below, we will separately investigate the infimum of $\mathcal{L}_h^{\mathrm{EVB}}$ over $\mathring{\mathcal{R}}$,

$$\underline{\mathring{\mathcal{L}}}_h^{\mathrm{EVB}} = \inf_{(\mu_{a_h}, \mu_{b_h}, \sigma_{a_h}^2, \sigma_{b_h}^2, c_{a_h}^2, c_{b_h}^2) \in \mathring{\mathcal{R}}} \mathcal{L}_h^{\mathrm{EVB}}(\mu_{a_h}, \mu_{b_h}, \sigma_{a_h}^2, \sigma_{b_h}^2, c_{a_h}^2, c_{b_h}^2), \tag{109}$$

and the infimum over $\breve{\mathcal{R}}$,

$$\underline{\breve{\mathcal{L}}}_h^{\mathrm{EVB}} = \inf_{(\mu_{a_h}, \mu_{b_h}, \sigma_{a_h}^2, \sigma_{b_h}^2, c_{a_h}^2, c_{b_h}^2) \in \breve{\mathcal{R}}} \mathcal{L}_h^{\mathrm{EVB}}(\mu_{a_h}, \mu_{b_h}, \sigma_{a_h}^2, \sigma_{b_h}^2, c_{a_h}^2, c_{b_h}^2).$$

Rigorously speaking, no minimizer over $\mathring{\mathcal{R}}$ exists. To make discussion simple, we approximate $\mathring{\mathcal{R}}$ by its subregion with an arbitrary accuracy; for any ε ($0 < \varepsilon < \kappa$), we define an ε-margin subregion of $\mathring{\mathcal{R}}$:

$$\mathring{\mathcal{R}}_\varepsilon = \left\{ (\mu_{a_h}, \mu_{b_h}, \sigma_{a_h}^2, \sigma_{b_h}^2, c_{a_h}^2, c_{b_h}^2) \in \mathring{\mathcal{R}} ; c_{a_h} c_{b_h} \ge \varepsilon \right\}.$$

Then the following lemma holds (its proof is given in Appendix G.11):

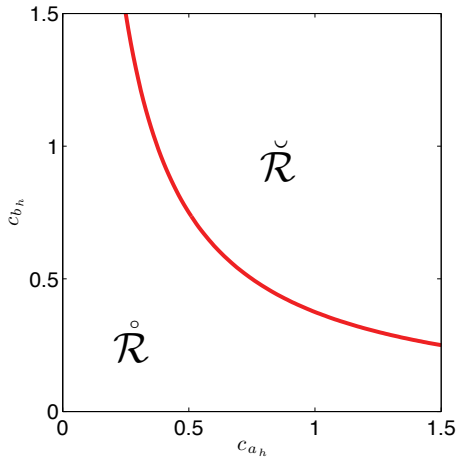


Figure 12: Division of the domain, defined by Equations (107) and (108), when $\gamma = 3, M = L = \sigma^2 = 1$. The hyperbolic boundary belongs to $\mathring{\mathcal{R}}$.

Lemma 19 *The minimizer over* $\mathring{\mathcal{R}}_\varepsilon$ *is given by*

$$\mathring{\mu}_{a_h} = 0, \tag{110}$$

$$\mathring{\mu}_{b_h} = 0, \tag{111}$$

$$\mathring{\sigma}^2_{a_h} = \frac{1}{2M}\left\{-\left(\frac{\sigma^2}{\varepsilon} - \varepsilon(M-L)\right) + \sqrt{\left(\frac{\sigma^2}{\varepsilon} - \varepsilon(M-L)\right)^2 + 4M\sigma^2}\right\}, \tag{112}$$

$$\mathring{\sigma}^2_{b_h} = \frac{1}{2L}\left\{-\left(\frac{\sigma^2}{\varepsilon} + \varepsilon(M-L)\right) + \sqrt{\left(\frac{\sigma^2}{\varepsilon} + \varepsilon(M-L)\right)^2 + 4L\sigma^2}\right\}, \tag{113}$$

$$\mathring{c}^2_{a_h} = \varepsilon, \tag{114}$$

$$\mathring{c}^2_{b_h} = \varepsilon, \tag{115}$$

and the infimum (109) *over* $\mathring{\mathcal{R}}$ *is given by*

$$\underline{\mathring{L}}_h^{\mathrm{EVB}} = L + M. \tag{116}$$

Note that Equations (110) and (111) result in the null output ($\widehat{\gamma}_h = \mathring{\mu}_{a_h}\mathring{\mu}_{b_h} = 0$). Accordingly, we call the minimizer (110)–(115) over $\mathring{\mathcal{R}}_\varepsilon$ the *null* (approximated) local minimizer.

On the other hand, we call any stationary point resulting in a *positive* output ($\widehat{\gamma}_h = \breve{\mu}_{a_h}\breve{\mu}_{b_h} > 0$) a *positive* stationary point. The following lemma holds (its proof is given in Appendix G.12):

Lemma 20 *Any* positive *stationary point lies in* $\breve{\mathcal{R}}$.

If

$$\underline{\mathring{L}}_h^{\mathrm{EVB}} < \underline{\breve{L}}_h^{\mathrm{EVB}}, \tag{117}$$

the *null* local minimizer is global over the whole domain (105) (more accurately, over $\mathring{\mathcal{R}}_\varepsilon \cup \breve{\mathcal{R}}$ for any $0 < \varepsilon < \kappa$). If

$$\mathring{\underline{\mathcal{L}}}_h^{\mathrm{EVB}} \geq \breve{\underline{\mathcal{L}}}_h^{\mathrm{EVB}}, \tag{118}$$

the global minimizers consist of *positive* stationary points, as the following lemma states (its proof is given in Appendix G.13):

Lemma 21 *When Equation* (118) *holds, the global minimizers consist of* positive *stationary points*.

Now, we look for the *positive* stationary points. According to Lemma 20, we can assume that Equation (98) holds. Equations (40) and (41) are reduced to

$$c_{a_h}^2 = \frac{\mu_{a_h}^2 + M\sigma_{a_h}^2}{M}, \tag{119}$$

$$c_{b_h}^2 = \frac{\mu_{b_h}^2 + L\sigma_{b_h}^2}{L}. \tag{120}$$

Then, Equations (74)–(77), (119), and (120) form a necessary and sufficient condition to be a stationary point of the objective function (106). Solving these equations, we have the following lemma (its proof is given in Appendix G.14):

Lemma 22 *At least one* positive *stationary point exists if and only if*

$$\gamma_h^2 \geq (\sqrt{L} + \sqrt{M})^2\sigma^2. \tag{121}$$

At any positive *stationary point,* $c_{a_h}^2 c_{b_h}^2$ *is given either by*

$$c_{a_h}^2 c_{b_h}^2 = \breve{c}_{a_h}^2 \breve{c}_{b_h}^2 = \frac{\left(\gamma_h^2 - (L+M)\sigma^2\right) + \sqrt{\left(\gamma_h^2 - (L+M)\sigma^2\right)^2 - 4LM\sigma^4}}{2LM}, \tag{122}$$

or by

$$c_{a_h}^2 c_{b_h}^2 = \acute{c}_{a_h}^2 \acute{c}_{b_h}^2 = \frac{\left(\gamma_h^2 - (L+M)\sigma^2\right) - \sqrt{\left(\gamma_h^2 - (L+M)\sigma^2\right)^2 - 4LM\sigma^4}}{2LM}. \tag{123}$$

We categorize the *positive* stationary points into two groups, based on the above two solutions of $c_{a_h}^2 c_{b_h}^2$; we say that a stationary point satisfying Equation (122) is a *large positive* stationary point, and one satisfying Equation (123) is a *small positive* stationary point. Note that, when

$$\gamma_h^2 = (\sqrt{L} + \sqrt{M})^2\sigma^2, \tag{124}$$

it holds that $\breve{c}_{a_h}^2 \breve{c}_{b_h}^2 = \acute{c}_{a_h}^2 \acute{c}_{b_h}^2$, and therefore, the *large positive* stationary points and the *small positive* stationary points coincide with each other. The following lemma allows us to focus on the *large positive* stationary points (its proof is given in Appendix G.15.):

Lemma 23 *When*

$$\gamma_h^2 > (\sqrt{L} + \sqrt{M})^2\sigma^2, \tag{125}$$

any small positive *stationary point is a saddle point*.

Summarizing Lemmas 19–23, we have the following lemma:

Lemma 24 *When Equation* (121) *holds, there are two possibilities: that the global minimizers consist of* large positive *stationary points (in the case when Equation* (118) *holds); or that the global minimizer is the* null *local minimizer (in the case when Equation* (117) *holds). When Equation* (121) *does not hold, the global minimizer is the* null *local minimizer.*

Hereafter, we assume that Equation (121) holds. We like to clarify when Equation (118) holds, so that *large positive* stationary points become global minimizers. The EVB objective function (106) is substantially more complex (see Appendix H for illustration) than the VB objective function (73) where the *null* stationary point turns from the global minimum to a saddle point no sooner than any *positive* stationary point arises.

Below, we derive a sufficient condition for any *large positive* stationary point to give a lower objective value than $\mathring{\underline{L}}_h^{\mathrm{EVB}}$. We evaluate the difference between the objectives:

$$\Delta_h(\breve{\mu}_{a_h}, \breve{\mu}_{b_h}, \breve{\sigma}^2_{a_h}, \breve{\sigma}^2_{b_h}, \breve{c}^2_{a_h}, \breve{c}^2_{b_h}) = \mathcal{L}_h^{\mathrm{EVB}}(\breve{\mu}_{a_h}, \breve{\mu}_{b_h}, \breve{\sigma}^2_{a_h}, \breve{\sigma}^2_{b_h}, \breve{c}^2_{a_h}, \breve{c}^2_{b_h}) - \mathring{\underline{L}}_h^{\mathrm{EVB}}. \tag{126}$$

If $\Delta_h(\breve{\mu}_{a_h}, \breve{\mu}_{b_h}, \breve{\sigma}^2_{a_h}, \breve{\sigma}^2_{b_h}, \breve{c}^2_{a_h}, \breve{c}^2_{b_h}) \leq 0$, Equation (118) holds. We obtain the following lemma (its proof is given in Appendix G.16.):

Lemma 25 $\Delta_h(\breve{\mu}_{a_h}, \breve{\mu}_{b_h}, \breve{\sigma}^2_{a_h}, \breve{\sigma}^2_{b_h}, \breve{c}^2_{a_h}, \breve{c}^2_{b_h})$ *is upper-bounded as*

$$\Delta_h(\breve{\mu}_{a_h}, \breve{\mu}_{b_h}, \breve{\sigma}^2_{a_h}, \breve{\sigma}^2_{b_h}, \breve{c}^2_{a_h}, \breve{c}^2_{b_h}) < M\psi(\alpha, \beta), \tag{127}$$

where

$$\psi(\alpha, \beta) = \log\beta + \alpha\log\left(\frac{\beta - (1-\alpha)}{\alpha}\right) + (1-\alpha) + \frac{2}{\sqrt{1 - \frac{(\alpha+\sqrt{\alpha}+1)}{\beta}}} - \beta, \tag{128}$$

$$\alpha = \frac{L}{M}, \tag{129}$$

$$\beta = \frac{\gamma_h^2}{M\sigma^2}. \tag{130}$$

Furthermore, the following lemma states that $\psi(\alpha, \beta)$ is negative when β is large enough (its proof is given in Appendix G.17.):

Lemma 26 $\psi(\alpha, \beta) < 0$ *for any* $0 < \alpha \leq 1$ *and* $\beta \geq 7$.

Combining Lemma 24 and Lemma 25, we obtain the following lemma:

Lemma 27 *When the condition* (127) *holds, the global minimizers consist of* large positive *stationary points*.

Combining Lemma 26 and Lemma 27, we obtain the following lemma:

Lemma 28 *When* $\beta \geq 7$, *the global minimizers consist of* large positive *stationary points*.

Finally, we derive bounds of the *large positive* stationary points (its proof is given in Appendix G.18):

Lemma 29 *Equations* (46), (47), *and* (48) *hold for any* large positive *stationary point*.

Combining Lemma 24, Lemma 28, and Lemma 29 completes the proof of Theorem 5. Combining Lemma 24 and Lemma 29 completes the proof of Theorem 6.

■

Appendix F. Proof of Corollary 2

Assume that $L = M$. When $\gamma_h \geq 2\sqrt{M}$, Lemma 22 guarantees that at least one *large positive* stationary point exists. In this case, Equation (122) leads to

$$\breve{c}_{a_h}\breve{c}_{b_h} = \frac{\gamma_h}{M}\rho_+. \tag{131}$$

Its inverse can be written as

$$\frac{1}{\breve{c}_{a_h}\breve{c}_{b_h}} = \frac{\gamma_h}{\sigma^2}\rho_-.$$

Corollary 1 provides the exact values for the *positive* stationary points $(\breve{\mu}_{a_h}, \breve{\mu}_{b_h}, \breve{\sigma}^2_{a_h}, \breve{\sigma}^2_{b_h})$, given $(\breve{c}^2_{a_h}, \breve{c}^2_{b_h}) = (\breve{c}_{a_h}\breve{c}_{b_h}, \breve{c}_{a_h}\breve{c}_{b_h})$. Therefore, we can compute the exact value of the difference (126) of the objective values between the *large positive* stationary points and the *null* local minimizer:

$$\begin{aligned}
\Delta_h &= 2M\log\left(\frac{\gamma_h}{M\sigma^2}\breve{\mu}_{a_h}\breve{\mu}_{b_h} + 1\right) + \frac{1}{\sigma^2}\left(-2\gamma_h\breve{\mu}_{a_h}\breve{\mu}_{b_h} + M^2\breve{c}^2_{a_h}\breve{c}^2_{b_h}\right) \\
&= 2M\left\{\log\left(\frac{\gamma_h^2}{M\sigma^2} - \frac{\gamma_h}{M\breve{c}_{a_h}\breve{c}_{b_h}}\right) - \left(\frac{\gamma_h^2}{M\sigma^2} - \frac{\gamma_h}{M\breve{c}_{a_h}\breve{c}_{b_h}}\right) + \left(1 + \frac{nM}{2\sigma^2}\breve{c}^2_{a_h}\breve{c}^2_{b_h}\right)\right\} \\
&= 2M\varphi(\gamma_h).
\end{aligned}$$

Here, the first equation directly comes from Equation (172), and the last equation is obtained by substituting Equation (131) into the second equation.

According to Lemma 24, when $\gamma_h \geq 2\sqrt{M}$ and $\Delta_h \leq 0$, the EVB solutions consist of *large positive* stationary points; otherwise, the EVB solution is the *null* local minimizer. Using Equations (114), (115), and (131), we obtain Equation (51). Equation (52) follows Lemma 26, because $\varphi(\gamma_h) = \Delta_h/(2M) < \psi(\alpha,\beta)/2$ for $\alpha = 1, \beta = \gamma_h^2/(M\sigma^2)$. ■

Appendix G. Proof of Lemmas

In this appendix, the proofs of all the lemmas are given.

G.1 Proof of Lemma 7

We minimize the left-hand side of Equation (61) with respect to A and B:

$$\begin{aligned}
&\min_{A,B}\left\{\mathrm{tr}(AC_A^{-1}A^\top) + \mathrm{tr}(BC_B^{-1}B^\top)\right\} \\
&\qquad \text{s.t. } BA^\top = \Omega_L\Gamma\Omega_R^\top.
\end{aligned} \tag{132}$$

We can remove the constraint by changing the variables as follows:

$$A \to \Omega_R\Gamma T^\top C_A^{1/2}, \qquad\qquad B \to \Omega_L T^{-1}C_A^{-1/2},$$

where T is a $H \times H$ non-singular matrix. Then, the problem (132) is rewritten as

$$\min_T\left\{\mathrm{tr}\left(T^\top T\Gamma^2\right) + \mathrm{tr}\left((TT^\top)^{-1}(C_AC_B)^{-1}\right)\right\}. \tag{133}$$

Let

$$T^{-1} = U_T D_T V_T^\top$$

be the singular value decomposition of T^{-1}, where $D_T = \mathrm{diag}(d_1, \ldots, d_H)$ ($\{d_h\}$ are in non-increasing order). Then, the problem (133) is written as

$$\min_{U_T, D_T, V_T} \left\{ \mathrm{tr}\left(U_T D_T^{-2} U_T^\top \Gamma^2 \right) + \mathrm{tr}\left(V_T D_T^2 V_T^\top (C_A C_B)^{-1} \right) \right\}. \tag{134}$$

The objective function in Equation (134) can be written with the doubly stochastic matrices

$$Q_U = U_T \bullet U_T,$$
$$Q_V = V_T \bullet V_T,$$

where $\bullet$ denotes the Hadamard product, as follows (Marshall et al., 2009):

$$(d_1^{-2}, \ldots, d_H^{-2}) Q_U (\widehat{\gamma}_1^2, \ldots, \widehat{\gamma}_H^2)^\top + (d_1^2, \ldots, d_H^2) Q_V ((c_{a_1} c_{b_1})^{-1}, \ldots, (c_{a_H} c_{b_H})^{-1})^\top.$$

Since $\{\widehat{\gamma}_h^2\}$ and $\{d_h^2\}$ are in non-increasing order, and $\{d_h^{-2}\}$ and $(c_{a_h} c_{b_h})^{-1}$ are in non-decreasing order, this is minimized when $Q_U = Q_V = I_H$ (which is attained with $U_T = V_T = I_H$) for any D_T.

Thus, the problem (134) is reduced to

$$\min_{\{d_h\}} \sum_{h=1}^H \left(\frac{\widehat{\gamma}_h^2}{d_h^2} + \frac{d_h^2}{(c_{a_h} c_{b_h})^2} \right).$$

This is minimized when $d_h^2 = \widehat{\gamma}_h c_{a_h} c_{b_h}$,[5] and the minimum coincides to the right-hand side of Equation (61), which completes the proof.

■

G.2 Proof of Lemma 8

It is known that the second term of Equation (60) is minimized when

$$A = (\sqrt{\gamma_1} \boldsymbol{\omega}_{a_1}, \ldots, \sqrt{\gamma_H} \boldsymbol{\omega}_{a_H}) T^\top,$$
$$B = (\sqrt{\gamma_1} \boldsymbol{\omega}_{b_1}, \ldots, \sqrt{\gamma_H} \boldsymbol{\omega}_{b_H}) T^{-1},$$

where T is any $H \times H$ non-singular matrix. Since the first term of Equation (60) does not depend on the directions of $\{\boldsymbol{a}_h, \boldsymbol{b}_h\}$, any minimizer can be written in the form of Equation (62) with $\{\widehat{\gamma}_h \geq 0\}$.

The degeneracy with respect to T is partly resolved by the first term of Equation (60). Suppose that we have obtained the best set of $\{\widehat{\gamma}_h\}$. Then, minimizing Equation (60) is equivalent to the following problem:

$$\begin{aligned} &\text{Given } \{\widehat{\gamma}_h \geq 0\}, \\ &\min_{A,B} \quad \left\{ \mathrm{tr}(A C_A^{-1} A^\top) + \mathrm{tr}(B C_B^{-1} B^\top) \right\} \\ &\text{s.t. } B A^\top = \sum_{h=1}^H \widehat{\gamma}_h \boldsymbol{\omega}_{b_h} \boldsymbol{\omega}_{a_h}^\top. \end{aligned} \tag{135}$$

5. If $\widehat{\gamma}_h = 0$, the minimum is attained by simply setting the corresponding column vectors of A and B to $(\boldsymbol{a}_h, \boldsymbol{b}_h) = (\boldsymbol{0}, \boldsymbol{0})$.

Lemma 7 guarantees that

$$\boldsymbol{a}_h = \sqrt{\frac{c_{a_h}}{c_{b_h}}\widehat{\gamma}_h}\boldsymbol{\omega}_{a_h},$$
$$\boldsymbol{b}_h = \sqrt{\frac{c_{b_h}}{c_{a_h}}\widehat{\gamma}_h}\boldsymbol{\omega}_{b_h},$$

give a solution for the problem (135) for any (so far unknown) set of $\{\widehat{\gamma}_h\}$, which completes the proof. ■

G.3 Proof of Lemma 9

Equation (60) can be written as

$$\mathcal{L}^{\mathrm{MAP}}(A,B) = \mathrm{tr}(AC_A^{-1}A^\top) + \mathrm{tr}(BC_B^{-1}B^\top) + \frac{1}{\sigma^2}\left\|V - BA^\top\right\|_{\mathrm{Fro}}^2.$$

This is invariant with respect to the transform

$$A \to A\Theta^\top,$$
$$B \to B\Theta^{-1},$$

since

$$\mathrm{tr}(A\Theta^\top C_A^{-1}\Theta A^\top) = \mathrm{tr}(AC_A^{-1/2}\Xi^\top C_A^{1/2}C_A^{-1}C_A^{1/2}\Xi C_A^{-1/2}A^\top) = \mathrm{tr}(AC_A^{-1}A^\top),$$
$$\mathrm{tr}(B\Theta^{-1}C_B^{-1}(\Theta^{-1})^\top B^\top) = \mathrm{tr}(BC_B^{-1/2}\Xi^\top C_B^{1/2}C_B^{-1}C_B^{1/2}\Xi C_B^{-1/2}B^\top) = \mathrm{tr}(BC_B^{-1}B^\top),$$
$$B\Theta^{-1}\Theta A = BA.$$

This completes the proof. ■

G.4 Proof of Lemma 10

Let

$$\Sigma_{a_h} = \sum_{m=1}^{M} \tau_m^{(a_h)} \boldsymbol{t}_m^{(a_h)} \boldsymbol{t}_m^{(a_h)\top},$$
$$\Sigma_{b_h} = \sum_{l=1}^{L} \tau_l^{(b_h)} \boldsymbol{t}_l^{(b_h)} \boldsymbol{t}_l^{(b_h)\top},$$

be the eigenvalue decompositions of Σ_{a_h} and Σ_{b_h} , where

$$\left(\tau_1^{(a_h)},\ldots,\tau_M^{(a_h)}\right) \in \mathbb{R}_{++}^M, \qquad \left(\tau_1^{(b_h)},\ldots,\tau_L^{(b_h)}\right) \in \mathbb{R}_{++}^L.$$

are the eigenvalues. Then, the objective function (67) is written as

$$\begin{aligned}
&\mathcal{L}^{\mathrm{VB}}(\{\boldsymbol{a}_h, \boldsymbol{b}_h, \boldsymbol{\tau}_m^{(a_h)}, \boldsymbol{\tau}_l^{(b_h)}\}) \\
&= \sum_{h=1}^{H} \left(- \sum_{m=1}^{M} \log \boldsymbol{\tau}_m^{(a_h)} + \frac{\|\boldsymbol{\mu}_{a_h}\|^2 + \sum_{m=1}^{M} \boldsymbol{\tau}_m^{(a_h)}}{c_{a_h}^2} - \sum_{l=1}^{L} \log \boldsymbol{\tau}_l^{(b_h)} + \frac{\|\boldsymbol{\mu}_{b_h}\|^2 + \sum_{l=1}^{L} \boldsymbol{\tau}_l^{(b_h)}}{c_{b_h}^2} \right) \\
&\quad + \frac{1}{\sigma^2} \left\| V - \sum_{h=1}^{H} \boldsymbol{\mu}_{b_h} \boldsymbol{\mu}_{a_h}^{\top} \right\|_{\mathrm{Fro}}^2 \\
&\quad + \frac{1}{\sigma^2} \sum_{h=1}^{H} \left(\|\boldsymbol{\mu}_{a_h}\|^2 \sum_{l=1}^{L} \boldsymbol{\tau}_l^{(b_h)} + \sum_{m=1}^{M} \boldsymbol{\tau}_m^{(a_h)} \|\boldsymbol{\mu}_{b_h}\|^2 + \left(\sum_{m=1}^{M} \boldsymbol{\tau}_m^{(a_h)} \right) \left(\sum_{l=1}^{L} \boldsymbol{\tau}_l^{(b_h)} \right) \right).
\end{aligned}$$

Since the second and the third terms are positive, this is lower-bounded as

$$\begin{aligned}
\mathcal{L}^{\mathrm{VB}}(\{\boldsymbol{a}_h, \boldsymbol{b}_h, \boldsymbol{\tau}_m^{(a_h)}, \boldsymbol{\tau}_l^{(b_h)}\}) &> \sum_{h=1}^{H} \left(\frac{\|\boldsymbol{\mu}_{a_h}\|^2}{c_{a_h}^2} + \sum_{m=1}^{M} \left(\frac{\boldsymbol{\tau}_m^{(a_h)}}{c_{a_h}^2} - \log \frac{\boldsymbol{\tau}_m^{(a_h)}}{c_{a_h}^2} \right) \right) \\
&\quad + \sum_{h=1}^{H} \left(\frac{\|\boldsymbol{\mu}_{b_h}\|^2}{c_{b_h}^2} + \sum_{l=1}^{L} \left(\frac{\boldsymbol{\tau}_l^{(b_h)}}{c_{b_h}^2} - \log \frac{\boldsymbol{\tau}_l^{(b_h)}}{c_{b_h}^2} \right) \right) - \sum_{h=1}^{H} \left(M \log c_{a_h}^2 + L \log c_{b_h}^2 \right). \qquad (136)
\end{aligned}$$

Focusing on the first term in Equation (136), we find that

$$\lim_{\|\mu_{a_h}\| \to \infty} \mathcal{L}^{\mathrm{VB}}(\{\boldsymbol{a}_h, \boldsymbol{b}_h, \boldsymbol{\tau}_m^{(a_h)}, \boldsymbol{\tau}_l^{(b_h)}\}) = \infty$$

for any h. Further,

$$\begin{aligned}
\lim_{\boldsymbol{\tau}_m^{(a_h)} \to 0} \mathcal{L}^{\mathrm{VB}}(\{\boldsymbol{a}_h, \boldsymbol{b}_h, \boldsymbol{\tau}_m^{(a_h)}, \boldsymbol{\tau}_l^{(b_h)}\}) &= \infty, \\
\lim_{\boldsymbol{\tau}_m^{(a_h)} \to \infty} \mathcal{L}^{\mathrm{VB}}(\{\boldsymbol{a}_h, \boldsymbol{b}_h, \boldsymbol{\tau}_m^{(a_h)}, \boldsymbol{\tau}_l^{(b_h)}\}) &= \infty,
\end{aligned}$$

for any (h,m), because $(x - \log x) \geq 1$ for any $x > 0$, $\lim_{x \to +0}(x - \log x) = \infty$, and $\lim_{x \to \infty}(x - \log x) = \infty$. The same holds for $\{\boldsymbol{\mu}_{b_h}\}$ and $\{\boldsymbol{\tau}_l^{(b_h)}\}$ because of the second term in Equation (136). Consequently, the objective function (67) goes to infinity when approaching to any point on the boundary of the domain (69). Since the objective function (67) is differentiable in the domain, any minimizer is a stationary point. For any observation V, the objective function (67) can be finite, for example, when $\|\boldsymbol{\mu}_{a_h}\| = \|\boldsymbol{\mu}_{b_h}\| = 0, \Sigma_{a_h} = I_M, \Sigma_{b_h} = I_L$. Therefore, at least one minimizer always exists. ■

G.5 Proof of Lemma 11

Combining Equations (76) and (77) and eliminating $\sigma_{b_h}^2$, we obtain

$$M \left(\mu_{b_h}^2 + \frac{\sigma^2}{c_{a_h}^2} \right) \sigma_{a_h}^4 + \left(\widehat{\eta}_h^2 - \sigma^2 (M - L) \right) \sigma_{a_h}^2 - \sigma^2 \left(\mu_{a_h}^2 + \frac{\sigma^2}{c_{b_h}^2} \right) = 0.$$

This has one positive and one negative solutions. Neglecting the negative one, we obtain

$$\sigma_{a_h}^2 = \frac{-\left(\widehat{\eta}_h^2 - \sigma^2(M-L)\right) + \sqrt{(\widehat{\eta}_h^2 - \sigma^2(M-L))^2 + 4M\sigma^2\widehat{\eta}_h^2}}{2M(\mu_{b_h}^2 + \sigma^2 c_{a_h}^{-2})}. \tag{137}$$

Similarly, combining Equations (76) and (77) and eliminating $\sigma_{a_h}^2$, we obtain

$$\sigma_{b_h}^2 = \frac{-\left(\widehat{\eta}_h^2 + \sigma^2(M-L)\right) + \sqrt{(\widehat{\eta}_h^2 + \sigma^2(M-L))^2 + 4L\sigma^2\widehat{\eta}_h^2}}{2L(\mu_{a_h}^2 + \sigma^2 c_{b_h}^{-2})}. \tag{138}$$

Note that Equations (137) and (138) are real and positive for any $(\mu_{a_h}, \mu_{b_h}) \in \mathbb{R}^2$ and $\widehat{\eta}_h \in \mathbb{R}_{++}$.

Let us focus on the *null* stationary points. Apparently, Equations (80) and (81) are necessary to satisfy Equations (74) and (75) and result in the *null* output $\widehat{\gamma}_h = \mathring{\mu}_{a_h}\mathring{\mu}_{b_h} = 0$. Substituting Equations (80) and (81) into Equations (137) and (138) leads to Equations (82) and (83).

■

G.6 Proof of Lemma 12

To prove the lemma, we transform the set of variables $(\mu_{a_h}, \mu_{b_h}, \sigma_{a_h}^2, \sigma_{b_h}^2)$ to $(\widehat{\gamma}_h, \widehat{\delta}_h, \sigma_{a_h}^2, \sigma_{b_h}^2, \widehat{\eta}_h)$, and the necessary and sufficient condition (74)–(77) to (86)–(90). The transform (92) is obtained from the definitions (78) and (84), which we use in the following when necessary.

First we show that Equation (91) is necessary for any *positive* stationary point. $\widehat{\gamma}_h$ and $\widehat{\delta}_h$ must be positive because Equations (74) and (75) imply that μ_{a_h} and μ_{b_h} have the same sign. $\sigma_{a_h}^2$ and $\sigma_{b_h}^2$ must be positive because of their original domain (72). $\widehat{\eta}_h$ must be positive by its definition (79).

Next, we obtain Equations (86)–(90) from Equations (74)–(77). Equation (86) simply comes from the definition (79) of the additional variable $\widehat{\eta}_h$, which we have introduced for convenience. Equations (89) and (90) are equivalent to Equations (137) and (138), which were derived from Equations (76) and (77) in Appendix G.5. Equations (87) and (88) are derived from Equations (74) and (75), as shown below.

Equations (137) and (138) can be rewritten as

$$\sigma_{a_h}^2 = \frac{-\left(\widehat{\eta}_h^2 - \sigma^2(M-L)\right) + \sqrt{(\widehat{\eta}_h^2 + \sigma^2(L+M))^2 - 4\sigma^4 LM}}{2M(\mu_{b_h}^2 + \sigma^2 c_{a_h}^{-2})}, \tag{139}$$

$$\sigma_{b_h}^2 = \frac{-\left(\widehat{\eta}_h^2 + \sigma^2(M-L)\right) + \sqrt{(\widehat{\eta}_h^2 + \sigma^2(L+M))^2 - 4\sigma^4 LM}}{2L(\mu_{a_h}^2 + \sigma^2 c_{b_h}^{-2})}. \tag{140}$$

Substituting Equations (139) and (140) into Equations (74) and (75), respectively, we have

$$2\sigma^2 M\left(\mu_{b_h}^2+\frac{\sigma^2}{c_{a_h}^2}\right)\frac{\mu_{a_h}}{\mu_{b_h}} = \gamma_h\left\{-\left(\widehat{\eta}_h^2-\sigma^2(M-L)\right)+\sqrt{\left(\eta_h^2+\sigma^2(L+M)\right)^2-4\sigma^4 LM}\right\}, \tag{141}$$

$$2\sigma^2 L\left(\mu_{a_h}^2+\frac{\sigma^2}{c_{b_h}^2}\right)\frac{\mu_{b_h}}{\mu_{a_h}} = \gamma_h\left\{-\left(\widehat{\eta}_h^2+\sigma^2(M-L)\right)+\sqrt{\left(\eta_h^2+\sigma^2(L+M)\right)^2-4\sigma^4 LM}\right\}. \tag{142}$$

Subtraction of Equation (142) from Equation (141) gives

$$2\sigma^2(M-L)\mu_{a_h}\mu_{b_h}+2\sigma^4\left(\frac{M\mu_{a_h}}{c_{a_h}^2\mu_{b_h}}-\frac{L\mu_{b_h}}{c_{b_h}^2\mu_{a_h}}\right)=2\sigma^2(M-L)\gamma_h,$$

which is equivalent to Equation (88).

The last condition (87) is derived by multiplying Equations (141) and (142) (of which the both sides are positive):

$$4\sigma^4 LM\widehat{\eta}_h^2=\gamma_h^2\left(2\widehat{\eta}_h^4+2\widehat{\eta}_h^2\sigma^2(L+M)-2\widehat{\eta}_h^2\sqrt{(\widehat{\eta}_h^2+\sigma^2(L+M))^2-4\sigma^4 LM}\right).$$

Dividing both sides by $2\widehat{\eta}_h^2\gamma_h^2$ (>0), we have

$$\sqrt{(\widehat{\eta}_h^2+\sigma^2(L+M))^2-4\sigma^4 LM}=\widehat{\eta}_h^2+\sigma^2(L+M)-\frac{2\sigma^4 LM}{\gamma_h^2}. \tag{143}$$

Note that the left-hand side of Equation (143) is always real and positive since

$$\begin{aligned}(\widehat{\eta}_h^2+\sigma^2(L+M))^2-4\sigma^4 LM &= (\widehat{\eta}_h^2-\sigma^2(M-L))^2+4M\sigma^2\widehat{\eta}_h^2\\ &>0.\end{aligned}$$

Therefore, the right-hand side of Equation (143) is non-negative when Equation (143) holds:

$$\widehat{\eta}_h^2+\sigma^2(L+M)-\frac{2\sigma^4 LM}{\gamma_h^2}\geq 0. \tag{144}$$

To obtain Equation (87) from Equation (143), we square Equation (143):

$$(\widehat{\eta}_h^2+\sigma^2(L+M))^2-4\sigma^4 LM=\left(\widehat{\eta}_h^2+\sigma^2(L+M)-\frac{2\sigma^4 LM}{\gamma_h^2}\right)^2. \tag{145}$$

Note that this is equivalent to Equation (143) only when Equation (144) holds. Equation (145) leads to

$$\frac{\sigma^4 LM}{\gamma_h^2}-(\widehat{\eta}_h^2+\sigma^2(L+M))+\gamma_h^2=0.$$

Solving this with respect to $\widehat{\eta}_h^2$ results in Equation (87). Equation (87) cannot hold with any real and positive value of $\widehat{\eta}_h$ when $\sigma^2 L \leq \gamma_h^2 \leq \sigma^2 M$. Further, substituting Equation (87) into Equation (144) gives

$$\gamma_h^2 - \frac{\sigma^4 LM}{\gamma_h^2} \geq 0.$$

Therefore, Equation (87) satisfies Equation (144) only when $\gamma_h^2 \geq \sigma^2\sqrt{LM}$. Accordingly, when Equation (85) holds, Equation (87) is equivalent to Equation (143). Otherwise, Equation (143) cannot hold, and no *positive* stationary point exists. ■

G.7 Proof of Lemma 13

Squaring both sides of Equation (86) (which are positive) and substituting Equation (87) into it, we have

$$\begin{aligned}\widehat{\gamma}_h^2 + \frac{\sigma^2}{c_{a_h} c_{b_h}} \left(\frac{c_{b_h}\widehat{\delta}_h}{c_{a_h}} + \frac{c_{a_h}}{c_{b_h}\widehat{\delta}_h} \right) \widehat{\gamma}_h \\ + \left(\frac{\sigma^4}{c_{a_h}^2 c_{b_h}^2} - \left(1 - \frac{\sigma^2 L}{\gamma_h^2}\right)\left(1 - \frac{\sigma^2 M}{\gamma_h^2}\right)\gamma_h^2 \right) = 0. \end{aligned} \tag{146}$$

Multiplying both sides of Equation (88) by $\widehat{\delta}_h$ (> 0) and solving it with respect to $\widehat{\delta}_h$, we obtain

$$\widehat{\delta}_h = \frac{(M-L)(\gamma_h - \widehat{\gamma}_h) + \sqrt{(M-L)^2(\gamma_h - \widehat{\gamma}_h)^2 + \frac{4\sigma^4 LM}{c_{a_h}^2 c_{b_h}^2}}}{2\sigma^2 M c_{a_h}^{-2}} \tag{147}$$

as a positive solution. We neglect the other solution, since it is negative. Substituting Equation (147) into Equation (146) gives Equation (93). Thus, we have transformed the necessary and sufficient condition Equations (86)–(90) to (93), (87), (147), (89), and (90). This proves the necessity.

Assume that Equation (85) holds and a positive real solution $\widehat{\gamma}_h$ of Equation (93) exists. Then, a positive real $\widehat{\eta}_h$ satisfying Equation (87) exists. For any existing $(\widehat{\gamma}_h, \widehat{\eta}_h) \in \mathbb{R}_{++}^2$, a positive real $\widehat{\delta}_h$ satisfying Equation (147) exists. For any existing $(\widehat{\gamma}_h, \widehat{\delta}_h, \widehat{\eta}_h) \in \mathbb{R}_{++}^3$, positive real $\sigma_{a_h}^2$ and $\sigma_{b_h}^2$ satisfying Equations (89) and (90) exist. Thus, whenever a positive real solution $\widehat{\gamma}_h$ of Equation (93) exists, the corresponding point $(\widehat{\gamma}_h, \widehat{\delta}_h, \sigma_{a_h}^2, \sigma_{b_h}^2, \widehat{\eta}_h) \in \mathbb{R}_{++}^5$ satisfying the necessary and sufficient condition (93), (87), (147), (89), and (90) exists. This proves the sufficiency.

Finally, suppose that we obtain a solution satisfying Equations (86)–(90) in the domain (91). Then, Equation (87) implies that

$$\gamma_h > \widehat{\eta}_h.$$

Moreover, ignoring the positive terms $\sigma^2/c_{b_h}^2$ and $\sigma^2/c_{a_h}^2$ in Equation (86), we have

$$\widehat{\eta}_h > \widehat{\gamma}_h.$$

Therefore, Equation (96) holds. ■

G.8 Proof of Lemma 15

Assume that $\gamma_h^2 > \sigma^2 M$. Then, the second inequality in Equation (98) holds if and only if

$$\left(1 - \frac{\sigma^2 L}{\gamma_h^2}\right)\left(1 - \frac{\sigma^2 M}{\gamma_h^2}\right)\gamma_h^2 - \frac{\sigma^4}{c_{a_h}^2 c_{b_h}^2} > 0.$$

The left-hand side can be factorized as

$$\gamma_h^{-2}\left(\gamma_h^2 - \left(\kappa + \sqrt{\kappa^2 - LM\sigma^4}\right)\right)\left(\gamma_h^2 - \left(\kappa - \sqrt{\kappa^2 - LM\sigma^4}\right)\right) > 0, \tag{148}$$

where

$$\kappa = \frac{(L+M)\sigma^2}{2} + \frac{\sigma^4}{2c_{a_h}^2 c_{b_h}^2}.$$

Since

$$\kappa - \sqrt{\kappa^2 - LM\sigma^4} < M\sigma^2 < \kappa + \sqrt{\kappa^2 - LM\sigma^4},$$

Equation (148) holds if and only if

$$\gamma_h^2 > \kappa + \sqrt{\kappa^2 - LM\sigma^4},$$

which leads to Equation (30). ■

G.9 Proof of Lemma 16

We show that the Hessian of the objective function (73) has at least one negative and one positive eigenvalues at the *null* stationary point, when any *positive* stationary point exists. We only focus on the 2-dimensional subspace spanned by (μ_{a_h}, μ_{b_h}). The partial derivatives of Equation (73) are given by

$$\frac{1}{2}\frac{\partial \mathcal{L}_h^{\mathrm{VB}}}{\partial \mu_{a_h}} = \frac{\mu_{a_h}}{c_{a_h}^2} + \left(\frac{-\gamma_h \mu_{b_h} + (\mu_{b_h}^2 + L\sigma_{b_h}^2)\mu_{a_h}}{\sigma^2}\right),$$
$$\frac{1}{2}\frac{\partial \mathcal{L}_h^{\mathrm{VB}}}{\partial \mu_{b_h}} = \frac{\mu_{b_h}}{c_{b_h}^2} + \left(\frac{-\gamma_h \mu_{a_h} + (\mu_{a_h}^2 + M\sigma_{a_h}^2)\mu_{b_h}}{\sigma^2}\right).$$

Then, the Hessian is given by

$$\begin{aligned}\frac{1}{2}\mathcal{H}^{\mathrm{VB}} &= \begin{pmatrix} \frac{1}{2}\frac{\partial^2 \mathcal{L}_h^{\mathrm{VB}}}{(\partial \mu_{a_h})^2} & \frac{1}{2}\frac{\partial^2 \mathcal{L}_h^{\mathrm{VB}}}{\partial \mu_{a_h}\partial \mu_{b_h}} \\ \frac{1}{2}\frac{\partial^2 \mathcal{L}_h^{\mathrm{VB}}}{\partial \mu_{a_h}\partial \mu_{b_h}} & \frac{1}{2}\frac{\partial^2 \mathcal{L}_h^{\mathrm{VB}}}{(\partial \mu_{b_h})^2} \end{pmatrix} \\ &= \sigma^2 \begin{pmatrix} \frac{\sigma^2}{c_{a_h}^2} + (\mu_{b_h}^2 + L\sigma_{b_h}^2) & -\gamma_h + 2\mu_{a_h}\mu_{b_h} \\ -\gamma_h + 2\mu_{a_h}\mu_{b_h} & \frac{\sigma^2}{c_{b_h}^2} + (\mu_{a_h}^2 + M\sigma_{a_h}^2) \end{pmatrix}. \end{aligned} \tag{149}$$

The determinant of Equation (149) is written as

$$\left|\frac{1}{2}\mathcal{H}^{\mathrm{VB}}\right| = \frac{1}{\sigma^4}\left(\frac{\sigma^2}{c_{a_h}^2}+(\mu_{b_h}^2+L\sigma_{b_h}^2)\right)\left(\frac{\sigma^2}{c_{b_h}^2}+(\mu_{a_h}^2+M\sigma_{a_h}^2)\right)-\frac{1}{\sigma^4}(2\mu_{a_h}\mu_{b_h}-\gamma_h)^2$$
$$= \frac{1}{\sigma_{a_h}^2\sigma_{b_h}^2}-\frac{1}{\sigma^4}(2\mu_{a_h}\mu_{b_h}-\gamma_h)^2, \tag{150}$$

where Equations (76) and (77) are used in the second equation.

The determinant (150) of the Hessian at the *null* stationary point, given by Equations (80)–(83), is written as

$$\left|\frac{1}{2}\mathring{\mathcal{H}}^{\mathrm{VB}}\right| = \frac{1}{\mathring{\sigma}_{a_h}^2\mathring{\sigma}_{b_h}^2}-\frac{1}{\sigma^4}\gamma_h^2. \tag{151}$$

Assume the existence of any *positive* stationary point, for which it holds that

$$\gamma_h^2 = \frac{\sigma^4}{\breve{\sigma}_{a_h}^2\breve{\sigma}_{b_h}^2}. \tag{152}$$

This is obtained by substituting Equation (75) into Equation (74) and dividing both sides by $\breve{\mu}_{a_h}\breve{\sigma}_{a_h}^2\breve{\sigma}_{b_h}^2/\sigma^4$ (>0). Note that Equation (152) is not required for the *null* stationary point where $\mathring{\mu}_{a_h}=0$. Substituting Equation (152) into Equation (151), we have

$$\left|\frac{1}{2}\mathring{\mathcal{H}}^{\mathrm{VB}}\right| = \frac{1}{\mathring{\sigma}_{a_h}^2\mathring{\sigma}_{b_h}^2}-\frac{1}{\breve{\sigma}_{a_h}^2\breve{\sigma}_{b_h}^2}. \tag{153}$$

Multiplying Equations (139) and (140) leads to

$$\sigma_{a_h}^2\sigma_{b_h}^2 = \frac{1}{4LM\widehat{\eta}_h^2}\left\{-\left(\widehat{\eta}_h^2-\sigma^2(M-L)\right)+\sqrt{\left(\eta_h^2+\sigma^2(L+M)\right)^2-4\sigma^4LM}\right\}$$
$$\times\left\{-\left(\widehat{\eta}_h^2+\sigma^2(M-L)\right)+\sqrt{\left(\eta_h^2+\sigma^2(L+M)\right)^2-4\sigma^4LM}\right\}$$
$$= \frac{1}{2LM}\left\{\widehat{\eta}_h^2+\sigma^2(L+M)-\sqrt{\left(\widehat{\eta}_h^2+\sigma^2(L+M)\right)^2-4\sigma^4LM}\right\},$$

which is decreasing with respect to $\widehat{\eta}_h$. Equation (79) implies that $\widehat{\eta}_h$ is larger at any *positive* stationary point than at the *null* stationary point. Therefore, it holds that $\mathring{\sigma}_{a_h}^2\mathring{\sigma}_{b_h}^2 > \breve{\sigma}_{a_h}^2\breve{\sigma}_{b_h}^2$, and Equation (153) is negative. This means that the Hessian $\mathring{\mathcal{H}}^{\mathrm{VB}}$ has one negative and one positive eigenvalues.

Consequently, the Hessian of the objective function (73) with respect to $(\mu_{a_h},\mu_{b_h},\sigma_{a_h}^2,\sigma_{b_h}^2)$ has at least one negative and one positive eigenvalues at the *null* stationary point, which proves the lemma. ■

G.10 Proof of Lemma 18

We rely on the monotonicity of the positive solution of the quadratic equation (97) with respect to q_1 and q_0; the positive solution $\widehat{\gamma}$ of (97) is a monotone decreasing function of q_1 and q_0 (see

Figure 11). Although Equation (93) is not really quadratic with respect to $\widehat{\gamma}_h$ because Equation (94) depends on $\widehat{\gamma}_h$, we can bound the positive solutions of Equation (93) by replacing the coefficients q_1 and q_0 with their bounds. Equation (93) might have multiple positive solutions if the left-hand side oscillates when crossing the horizontal axis in Fig.11. However, our approach bounds all the positive solutions, and Lemma 17 guarantees that the minimizers consist of some of them when Equation (98) holds.

First we derive an upper-bound of $\widehat{\gamma}_h^2$. Let us lower-bound Equation (94) by ignoring the positive term $4\sigma^4 LM/(c_{a_h}^2 c_{b_h}^2)$:

$$\begin{aligned} q_1(\widehat{\gamma}_h) &= \frac{-(M-L)^2(\gamma_h-\widehat{\gamma}_h)+(L+M)\sqrt{(M-L)^2(\gamma_h-\widehat{\gamma}_h)^2+\frac{4\sigma^4 LM}{c_{a_h}^2 c_{b_h}^2}}}{2LM} \\ &> \frac{-(M-L)^2(\gamma_h-\widehat{\gamma}_h)+(L+M)\sqrt{(M-L)^2(\gamma_h-\widehat{\gamma}_h)^2}}{2LM} \\ &= \left(1-\frac{L}{M}\right)(\gamma_h-\widehat{\gamma}_h). \end{aligned}$$

We also lower-bound Equation (95) by ignoring the positive term $\sigma^4/(c_{a_h}^2 c_{b_h}^2)$. Then we can obtain an upper-bound of $\widehat{\gamma}_h$:

$$\widehat{\gamma}_h < \widehat{\gamma}_h^{\mathrm{up}},$$

where $\widehat{\gamma}_h^{\mathrm{up}}$ is the larger solution of the following equation:

$$(\widehat{\gamma}_h^{\mathrm{up}})^2 + \left(\frac{M}{L}-1\right)\gamma_h\widehat{\gamma}_h^{\mathrm{up}} - \frac{M}{L}\left(1-\frac{\sigma^2 L}{\gamma_h^2}\right)\left(1-\frac{\sigma^2 M}{\gamma_h^2}\right)\gamma_h^2 = 0.$$

This can be factorized as

$$\left(\widehat{\gamma}_h^{\mathrm{up}} - \left(1-\frac{\sigma^2 M}{\gamma_h^2}\right)\gamma_h\right)\left(\widehat{\gamma}_h^{\mathrm{up}} + \frac{M}{L}\left(1-\frac{\sigma^2 L}{\gamma_h^2}\right)\gamma_h\right) = 0.$$

Thus, the larger solution of this equation,

$$\widehat{\gamma}_h^{\mathrm{up}} = \left(1-\frac{\sigma^2 M}{\gamma_h^2}\right)\gamma_h,$$

gives the upper-bound in Equation (28).

Similarly, we derive a lower-bound of $\widehat{\gamma}_h^2$. Let us upper-bound Equation (94) by using the relation $\sqrt{x^2+y^2} \leq \sqrt{x^2+y^2+2xy} \leq x+y$ for $x, y \geq 0$:

$$\begin{aligned} q_1(\widehat{\gamma}_h) &= \frac{-(M-L)^2(\gamma_h-\widehat{\gamma}_h)+(L+M)\sqrt{(M-L)^2(\gamma_h-\widehat{\gamma}_h)^2+\frac{4\sigma^4 LM}{c_{a_h}^2 c_{b_h}^2}}}{2LM} \\ &\leq \frac{-(M-L)^2(\gamma_h-\widehat{\gamma}_h)+(L+M)\left((M-L)(\gamma_h-\widehat{\gamma}_h)+\frac{2\sigma^2\sqrt{LM}}{c_{a_h}c_{b_h}}\right)}{2LM} \\ &= \left(1-\frac{L}{M}\right)(\gamma_h-\widehat{\gamma}_h)+\frac{2\sigma^2(L+M)\sqrt{LM}}{2LMc_{a_h}c_{b_h}} \\ &= \left(1-\frac{L}{M}\right)(\gamma_h-\widehat{\gamma}_h)+\frac{\sigma^2(L+M)}{\sqrt{LM}c_{a_h}c_{b_h}}. \end{aligned}$$

We also upper-bound Equation (95) by adding a non-negative term

$$\frac{(M-L)\sigma^2}{Lc_{a_h}c_{b_h}}\left(\frac{1}{c_{a_h}c_{b_h}}+\frac{\sigma^2\sqrt{LM}}{\gamma_h}\right).$$

Then we can obtain a lower-bound of $\widehat{\gamma}_h$:

$$\widehat{\gamma}_h \geq \widehat{\gamma}_h^{\mathrm{lo}},$$

where $\widehat{\gamma}_h^{\mathrm{lo}}$ is the larger solution of the following equation:

$$L(\widehat{\gamma}_h^{\mathrm{lo}})^2+\left((M-L)\gamma_h+\frac{\sigma^2(L+M)\sqrt{M/L}}{c_{a_h}c_{b_h}}\right)\widehat{\gamma}_h^{\mathrm{lo}} + \frac{M^2\sigma^4}{Lc_{a_h}^2c_{b_h}^2}+\frac{\sigma^4M(M-L)\sqrt{M/L}}{\gamma_h c_{a_h}c_{b_h}}-M\left(1-\frac{\sigma^2L}{\gamma_h^2}\right)\left(1-\frac{\sigma^2M}{\gamma_h^2}\right)\gamma_h^2=0.$$

This can be factorized as

$$\left(\widehat{\gamma}_h^{\mathrm{lo}}-\left(1-\frac{\sigma^2M}{\gamma_h^2}\right)\gamma_h+\frac{\sigma^2\sqrt{M/L}}{c_{a_h}c_{b_h}}\right)\left(L\widehat{\gamma}_h^{\mathrm{lo}}+M\left(1-\frac{\sigma^2L}{\gamma_h^2}\right)\gamma_h+\frac{\sigma^2M\sqrt{M/L}}{c_{a_h}c_{b_h}}\right)=0.$$

Thus, the larger solution of this equation,

$$\widehat{\gamma}_h^{\mathrm{lo}}=\left(1-\frac{\sigma^2M}{\gamma_h^2}\right)\gamma_h-\frac{\sigma^2\sqrt{M/L}}{c_{a_h}c_{b_h}},$$

gives the lower-bound in Equation (28).

The coefficient of the second term of Equation (146),

$$\frac{\sigma^2}{c_{a_h}c_{b_h}}\left(\frac{c_{b_h}\widehat{\delta}_h}{c_{a_h}}+\frac{c_{a_h}}{c_{b_h}\widehat{\delta}_h}\right),$$

is minimized when

$$\widehat{\delta}_h=\frac{c_{a_h}}{c_{b_h}}.$$

Then we can obtain another upper-bound of $\widehat{\gamma}_h$:

$$\widehat{\gamma}_h \leq \widehat{\gamma}_h'^{\mathrm{up}},$$

where $\widehat{\gamma}_h'^{\mathrm{up}}$ is the larger solution of the following equation:

$$(\widehat{\gamma}_h'^{\mathrm{up}})^2+\left(\frac{2\sigma^2}{c_{a_h}c_{b_h}}\right)\widehat{\gamma}_h'^{\mathrm{up}}+\frac{\sigma^4}{c_{a_h}^2c_{b_h}^2}-\left(1-\frac{\sigma^2L}{\gamma_h^2}\right)\left(1-\frac{\sigma^2M}{\gamma_h^2}\right)\gamma_h^2=0.$$

This can be factorized as

$$\left(\widehat{\gamma}_h'^{\mathrm{up}} - \sqrt{\left(1-\frac{\sigma^2 L}{\gamma_h^2}\right)\left(1-\frac{\sigma^2 M}{\gamma_h^2}\right)}\gamma_h + \frac{\sigma^2}{c_{a_h}c_{b_h}}\right) \times \left(\widehat{\gamma}_h'^{\mathrm{up}} + \sqrt{\left(1-\frac{\sigma^2 L}{\gamma_h^2}\right)\left(1-\frac{\sigma^2 M}{\gamma_h^2}\right)}\gamma_h + \frac{\sigma^2}{c_{a_h}c_{b_h}}\right) = 0.$$

Thus, the larger solution of this equation,

$$\widehat{\gamma}_h'^{\mathrm{up}} = \sqrt{\left(1-\frac{\sigma^2 L}{\gamma_h^2}\right)\left(1-\frac{\sigma^2 M}{\gamma_h^2}\right)}\gamma_h - \frac{\sigma^2}{c_{a_h}c_{b_h}},$$

gives the upper-bound in Equation (31). ■

G.11 Proof of Lemma 19

Consider the two-step minimization, (103) and (104). Lemma 17 implies that the minimizer of Equation (103) is the *null* stationary point for any given $(c_{a_h}^2, c_{b_h}^2)$ in $\mathring{\mathcal{R}}$. The *null* stationary point is explicitly given by Lemma 11. Substituting Equations (80)–(83) into Equation (106) gives

$$\mathring{\mathcal{L}}_h^{\mathrm{EVB}}(c_{a_h}^2, c_{b_h}^2) = M(-\log\lambda_{a,1} + \lambda_{a,1}) + L(-\log\lambda_{b,1} + \lambda_{b,1}) + \frac{LM\lambda_{a,0}\lambda_{b,0}}{\sigma^2}. \tag{154}$$

where

$$\lambda_{a,k}(c_{a_h}c_{b_h}) = \frac{1}{2M(c_{a_h}c_{b_h})^k}\left\{-\left(\frac{\sigma^2}{c_{a_h}c_{b_h}} - c_{a_h}c_{b_h}(M-L)\right) + \sqrt{\left(\frac{\sigma^2}{c_{a_h}c_{b_h}} - c_{a_h}c_{b_h}(M-L)\right)^2 + 4M\sigma^2}\right\},$$

$$\lambda_{b,k}(c_{a_h}c_{b_h}) = \frac{1}{2L(c_{a_h}c_{b_h})^k}\left\{-\left(\frac{\sigma^2}{c_{a_h}c_{b_h}} + c_{a_h}c_{b_h}(M-L)\right) + \sqrt{\left(\frac{\sigma^2}{c_{a_h}c_{b_h}} + c_{a_h}c_{b_h}(M-L)\right)^2 + 4L\sigma^2}\right\}.$$

Note that $\lambda_{a,k} > 0, \lambda_{b,k} > 0$ for any k, and that Equation (154) depends on $c_{a_h}^2$ and $c_{b_h}^2$ only through their product $c_{a_h}c_{b_h}$.

Consider a decreasing mapping $x = \sigma^2/(c_{a_h}^2 c_{b_h}^2)$ (>0). Then, $\lambda_{a,1}$ and $\lambda_{b,1}$ are written as

$$\lambda_{a,1}'(x) = 1 - \frac{(x+(L+M)) - \sqrt{(x+(L+M))^2 - 4ML}}{2M},$$

$$\lambda_{b,1}'(x) = 1 - \frac{(x+(L+M)) - \sqrt{(x+(L+M))^2 - 4ML}}{2L}.$$

Since they are increasing with respect to x, $\lambda_{a,1}$ and $\lambda_{b,1}$ are decreasing with respect to $c_{a_h}c_{b_h}$. Further, $\lambda_{a,1}$ and $\lambda_{b,1}$ are upper-bounded as

$$\lambda_{a,1}(c_{a_h}c_{b_h}) < \lim_{c_{a_h}c_{b_h}\to+0}\lambda_{a,1}(c_{a_h}c_{b_h}) = \lim_{x\to\infty}\lambda'_{a,1}(x) = 1,$$
$$\lambda_{b,1}(c_{a_h}c_{b_h}) < \lim_{c_{a_h}c_{b_h}\to+0}\lambda_{b,1}(c_{a_h}c_{b_h}) = \lim_{x\to\infty}\lambda'_{b,1}(x) = 1.$$

Since $(-\log\lambda+\lambda)$ is decreasing in the range $0<\lambda<1$, the first two terms in Equation (154) are increasing with respect to $c_{a_h}c_{b_h}$, and lower-bounded as

$$M(-\log\lambda_{a,1}+\lambda_{a,1}) > \lim_{c_{a_h}c_{b_h}\to+0}M(-\log\lambda_{a,1}+\lambda_{a,1}) = M, \tag{155}$$
$$L(-\log\lambda_{b,1}+\lambda_{b,1}) > \lim_{c_{a_h}c_{b_h}\to+0}L(-\log\lambda_{b,1}+\lambda_{b,1}) = L. \tag{156}$$

Similarly, using the same decreasing mapping, we have

$$\lambda'_{a,0}(x)\cdot\lambda'_{b,0}(x) = \frac{\sigma^2}{2LM}\left((x+(L+M)) - \sqrt{(x+(L+M))^2-4LM}\right).$$

Since this is decreasing with respect to x and lower-bounded by zero, $\lambda_{a,0}\lambda_{b,0}$ is increasing with respect to $c_{a_h}c_{b_h}$ and lower-bounded as

$$\lambda_{a,0}(c_{a_h}c_{b_h})\cdot\lambda_{b,0}(c_{a_h}c_{b_h}) > \lim_{c_{a_h}c_{b_h}\to+0}\lambda_{a,0}(c_{a_h}c_{b_h})\cdot\lambda_{b,0}(c_{a_h}c_{b_h}) = \lim_{x\to\infty}\lambda'_{a,0}(x)\cdot\lambda'_{b,0}(x) = 0.$$

Therefore, the third term in Equation (154) is increasing with respect to $c_{a_h}c_{b_h}$, and lower-bounded as

$$\frac{LM\lambda_{a,0}\lambda_{b,0}}{\sigma^2} > \lim_{c_{a_h}c_{b_h}\to+0}\frac{LM\lambda_{a,0}\lambda_{b,0}}{\sigma^2} = 0. \tag{157}$$

Now we have found that Equation (154) is increasing with respect to $c_{a_h}c_{b_h}$, because it consists of the increasing terms. Equations (114) and (115) minimize $c_{a_h}c_{b_h}$ over $\mathring{\mathcal{R}}_\varepsilon$ when Equation (43) is adopted. Therefore, they minimize Equation (154). Equations (110)–(113) are obtained by substituting Equations (114) and (115) into Equations (80)–(83). Since the infima (155)–(157) of the three terms of Equation (154) are obtained at the same time with the minimizer in the limit when $\varepsilon\to+0$, we have Equation (116). ■

G.12 Proof of Lemma 20

Existence of any *positive* stationary point lying in $\mathring{R}$ contradicts with Lemma 14. ■

G.13 Proof of Lemma 21

Assume that Equation (118) holds. Then, any global minimizer or point sequence giving the global infimum $\underline{\check{L}}_h^{\mathrm{EVB}}$ exists in $\check{R}$. Let us investigate the objective function (106). It is differentiable in the

domain (102), and lower-bounded as

$$\mathcal{L}_h^{\mathrm{EVB}}(\mu_{a_h},\mu_{b_h},\sigma_{a_h}^2,\sigma_{b_h}^2,c_{a_h}^2,c_{b_h}^2) \geq \mu_{a_h}^2\left(\frac{1}{c_{a_h}^2}+\frac{1}{\sigma^2}L\sigma_{b_h}^2\right)+\mu_{b_h}^2\left(\frac{1}{c_{b_h}^2}+\frac{1}{\sigma^2}M\sigma_{a_h}^2\right)$$
$$+M\left(\frac{\sigma_{a_h}^2}{c_{a_h}^2}-\log\frac{\sigma_{a_h}^2}{c_{a_h}^2}\right)+L\left(\frac{\sigma_{b_h}^2}{c_{b_h}^2}-\log\frac{\sigma_{b_h}^2}{c_{b_h}^2}\right)+\frac{1}{\sigma^2}\left(LM\sigma_{a_h}^2\sigma_{b_h}^2-\gamma_h^2\right). \tag{158}$$

Note that each term is lower-bounded by a finite value, since $(x-\log x)\geq 1$ for any $x>0$.

Since any sequence such that $c_{a_h}^2\to 0$ or $c_{b_h}^2\to 0$ goes into $\mathring{R}$, it cannot give $\check{\underline{L}}_h^{\mathrm{EVB}}$. Accordingly, we neglect such sequences. Then, we find that the lower-bound (158) goes to infinity when $\sigma_{a_h}^2\to 0$ or $\sigma_{b_h}^2\to 0$, because of the third and the fourth terms (note that $\lim_{x\to+0}(x-\log x)=\infty$). Further, it goes to infinity when $\sigma_{a_h}^2\to\infty$ or $\sigma_{b_h}^2\to\infty$, because of the fifth term. It also goes to infinity when $|\mu_{a_h}|\to\infty$ or $|\mu_{b_h}|\to\infty$, because of the first and the second terms. Finally, it goes to infinity when $c_{a_h}^2\to\infty$ or $c_{b_h}^2\to\infty$, because of the third and the fourth terms.

The above mean that the objective function (106) goes to infinity when approaching to any point on the domain boundary included in $\check{R}$. Consequently, the minimizers consist of stationary points in $\check{R}$. According to Lemma 14 and Lemma 16, the *null* stationary points in $\check{R}$ are saddle points. Therefore, the minimizers consist of *positive* stationary points. ■

G.14 Proof of Lemma 22

Substituting Equation (75) into Equation (74) gives

$$\gamma_h^2=\frac{\sigma^4}{\sigma_{a_h}^2\sigma_{b_h}^2}. \tag{159}$$

Substituting Equations (76) and (77) into Equation (159), we have

$$\gamma_h^2=\left(\mu_{a_h}^2+M\sigma_{a_h}^2+\frac{\sigma^2}{c_{b_h}^2}\right)\left(\mu_{b_h}^2+L\sigma_{b_h}^2+\frac{\sigma^2}{c_{b_h}^2}\right). \tag{160}$$

Substituting Equations (119) and (120) into Equation (160) gives

$$\gamma_h^2=\left(Mc_{a_h}^2+\frac{\sigma^2}{c_{b_h}^2}\right)\left(Lc_{b_h}^2+\frac{\sigma^2}{c_{a_h}^2}\right).$$

From this, we have

$$LMc_{a_h}^4c_{b_h}^4-\left(\gamma_h^2-(L+M)\sigma^2\right)c_{a_h}^2c_{b_h}^2+\sigma^4=0. \tag{161}$$

Solving Equation (161) with respect to $c_{a_h}^2c_{b_h}^2$, we obtain two solutions:

$$c_{a_h}^2c_{b_h}^2=\frac{\left(\gamma_h^2-(L+M)\sigma^2\right)\pm\sqrt{\left(\gamma_h^2-(L+M)\sigma^2\right)^2-4LM\sigma^4}}{2LM}. \tag{162}$$

On the other hand, because of the redundancy with respect to the transform (42), we can fix the ratio of the hyperparameters as in Equation (43). Thus, we have transformed the necessary and sufficient condition (74)–(77), (119), and (120) to (74)–(77), and (162). Since

$$\begin{aligned}&\sqrt{\left(\gamma_h^2-(L+M)\sigma^2\right)^2-4LM\sigma^4}\\&\qquad=\sqrt{\left(\gamma_h^2-(\sqrt{L}+\sqrt{M})^2\sigma^2\right)\left(\gamma_h^2-(\sqrt{M}-\sqrt{L})^2\sigma^2\right)}\end{aligned}$$

and

$$\sqrt{(\sqrt{M}-\sqrt{L})^2\sigma^2}<\sqrt{M\sigma^2},$$

the two solutions (162) are real and positive if and only if Equation (121) holds. This proves the necessity.

Suppose that Equation (121) holds. Then, the two solutions (162) exist. The inverse of the smaller solution (123) is written as

$$\frac{1}{\acute{c}_{a_h}^2\acute{c}_{b_h}^2}=\frac{\left(\gamma_h^2-(L+M)\sigma^2\right)+\sqrt{\left(\gamma_h^2-(L+M)\sigma^2\right)^2-4LM\sigma^4}}{2\sigma^4}. \tag{163}$$

This is upper-bounded as

$$\frac{1}{\acute{c}_{a_h}^2\acute{c}_{b_h}^2}<\frac{1}{\sigma^4}\left(\gamma_h^2-(L+M)\sigma^2\right).$$

Using this bound, we have

$$\begin{aligned}&\sqrt{\left(1-\frac{\sigma^2L}{\gamma_h^2}\right)\left(1-\frac{\sigma^2M}{\gamma_h^2}\right)}\gamma_h-\frac{\sigma^2}{\acute{c}_{a_h}\acute{c}_{b_h}}\\&\qquad>\sqrt{\gamma_h^2-(L+M)\sigma^2+\frac{LM\sigma^4}{\gamma_h^2}}-\sqrt{\gamma_h^2-(L+M)\sigma^2}\\&\qquad>0.\end{aligned}$$

This means that Equation (98) holds. The same holds for the larger solution (122), since

$$\frac{1}{\check{c}_{a_h}\check{c}_{b_h}}\leq\frac{1}{\acute{c}_{a_h}\acute{c}_{b_h}}.$$

Consequently, Lemma 14 guarantees the existence of at least one *positive* stationary point $(\check{\mu}_{a_h},\check{\mu}_{b_h},\check{\sigma}_{a_h}^2,\check{\sigma}_{b_h}^2)\in\mathbb{R}^2\times\mathbb{R}_{++}^2$ satisfying Equations (74)–(77), given any $(c_{a_h}^2,c_{b_h}^2)\in\mathbb{R}_{++}^2$ constructed from Equation (43) and either of the two solutions (162). Thus, we have shown the existence of at least one *positive* stationary point satisfying the necessary and sufficient condition (74)–(77), and (162) when Equation (121) holds. This proves the sufficiency. ■

G.15 Proof of Lemma 23

We show that, when Equation (125) holds, the Hessian of the objective function (106) has at least one negative and one positive eigenvalues at any *small positive* stationary point. We only focus on the 4-dimensional subspace spanned by $(\mu_{a_h}, \mu_{b_h}, c_{a_h}^2, c_{b_h}^2)$. The partial derivatives of the objective function (106) are

$$\begin{aligned}
\frac{1}{2}\frac{\partial \mathcal{L}_h^{\mathrm{EVB}}}{\partial \mu_{a_h}} &= \frac{\mu_{a_h}}{c_{a_h}^2} + \frac{-\gamma_h \mu_{b_h} + (\mu_{b_h}^2 + L\sigma_{b_h}^2)\mu_{a_h}}{\sigma^2}, \\
\frac{1}{2}\frac{\partial \mathcal{L}_h^{\mathrm{EVB}}}{\partial \mu_{b_h}} &= \frac{\mu_{b_h}}{c_{b_h}^2} + \frac{-\gamma_h \mu_{a_h} + (\mu_{a_h}^2 + M\sigma_{a_h}^2)\mu_{b_h}}{\sigma^2}, \\
\frac{1}{2}\frac{\partial \mathcal{L}_h^{\mathrm{EVB}}}{\partial c_{a_h}^2} &= \frac{1}{2}\left(\frac{M}{c_{a_h}^2} - \frac{(\mu_{a_h}^2 + M\sigma_{a_h}^2)}{c_{a_h}^4}\right), \\
\frac{1}{2}\frac{\partial \mathcal{L}_h^{\mathrm{EVB}}}{\partial c_{b_h}^2} &= \frac{1}{2}\left(\frac{L}{c_{b_h}^2} - \frac{(\mu_{b_h}^2 + L\sigma_{b_h}^2)}{c_{b_h}^4}\right).
\end{aligned}$$

Then, the Hessian is given by

$$\begin{aligned}
\frac{1}{2}\mathcal{H}^{\mathrm{EVB}} &= \begin{pmatrix}
\frac{1}{2}\frac{\partial^2 \mathcal{L}_h^{\mathrm{EVB}}}{(\partial \mu_{a_h})^2} & \frac{1}{2}\frac{\partial^2 \mathcal{L}_h^{\mathrm{EVB}}}{\partial \mu_{a_h}\partial \mu_{b_h}} & \frac{1}{2}\frac{\partial^2 \mathcal{L}_h^{\mathrm{EVB}}}{\partial \mu_{a_h}\partial c_{a_h}^2} & \frac{1}{2}\frac{\partial^2 \mathcal{L}_h^{\mathrm{EVB}}}{\partial \mu_{a_h}\partial c_{b_h}^2} \\
\frac{1}{2}\frac{\partial^2 \mathcal{L}_h^{\mathrm{EVB}}}{\partial \mu_{b_h}\partial \mu_{a_h}} & \frac{1}{2}\frac{\partial^2 \mathcal{L}_h^{\mathrm{EVB}}}{(\partial \mu_{b_h})^2} & \frac{1}{2}\frac{\partial^2 \mathcal{L}_h^{\mathrm{EVB}}}{\partial \mu_{b_h}\partial c_{a_h}^2} & \frac{1}{2}\frac{\partial^2 \mathcal{L}_h^{\mathrm{EVB}}}{\partial \mu_{b_h}\partial c_{b_h}^2} \\
\frac{1}{2}\frac{\partial^2 \mathcal{L}_h^{\mathrm{EVB}}}{\partial c_{a_h}^2\partial \mu_{a_h}} & \frac{1}{2}\frac{\partial^2 \mathcal{L}_h^{\mathrm{EVB}}}{\partial c_{a_h}^2\partial \mu_{b_h}} & \frac{1}{2}\frac{\partial^2 \mathcal{L}_h^{\mathrm{EVB}}}{(\partial c_{a_h}^2)^2} & \frac{1}{2}\frac{\partial^2 \mathcal{L}_h^{\mathrm{EVB}}}{\partial c_{a_h}^2\partial c_{b_h}^2} \\
\frac{1}{2}\frac{\partial^2 \mathcal{L}_h^{\mathrm{EVB}}}{\partial c_{b_h}^2\partial \mu_{a_h}} & \frac{1}{2}\frac{\partial^2 \mathcal{L}_h^{\mathrm{EVB}}}{\partial c_{b_h}^2\partial \mu_{b_h}} & \frac{1}{2}\frac{\partial^2 \mathcal{L}_h^{\mathrm{EVB}}}{\partial c_{b_h}^2\partial c_{a_h}^2} & \frac{1}{2}\frac{\partial^2 \mathcal{L}_h^{\mathrm{EVB}}}{(\partial c_{b_h}^2)^2}
\end{pmatrix} \\
&= \begin{pmatrix}
\frac{1}{c_{a_h}^2} + \frac{\mu_{b_h}^2 + L\sigma_{b_h}^2}{\sigma^2} & \frac{2\mu_{a_h}\mu_{b_h} - \gamma_h}{\sigma^2} & -\frac{\mu_{a_h}}{c_{a_h}^4} & 0 \\
\frac{2\mu_{a_h}\mu_{b_h} - \gamma_h}{\sigma^2} & \frac{1}{c_{b_h}^2} + \frac{\mu_{a_h}^2 + M\sigma_{a_h}^2}{\sigma^2} & 0 & -\frac{\mu_{b_h}}{c_{b_h}^4} \\
-\frac{\mu_{a_h}}{c_{a_h}^4} & 0 & \frac{2(\mu_{a_h}^2 + M\sigma_{a_h}^2) - Mc_{a_h}^2}{2c_{a_h}^6} & 0 \\
0 & -\frac{\mu_{b_h}}{c_{b_h}^4} & 0 & \frac{2(\mu_{b_h}^2 + L\sigma_{b_h}^2) - Lc_{b_h}^2}{2c_{b_h}^6}
\end{pmatrix}.
\end{aligned} \tag{164}$$

At any *positive* stationary point, Equations (74)–(77), (119), and (120) hold. Substituting Equations (76), (77), (119), and (120) into (164), we have

$$\frac{1}{2}\mathcal{H}^{\mathrm{EVB}} = \begin{pmatrix}
\frac{1}{\sigma_{a_h}^2} & \frac{\gamma_h - 2\mu_{a_h}\mu_{b_h}}{\sigma^2} & -\frac{\mu_{a_h}}{c_{a_h}^4} & 0 \\
\frac{\gamma_h - 2\mu_{a_h}\mu_{b_h}}{\sigma^2} & \frac{1}{\sigma_{b_h}^2} & 0 & -\frac{\mu_{b_h}}{c_{b_h}^4} \\
-\frac{\mu_{a_h}}{c_{a_h}^4} & 0 & \frac{M}{2c_{a_h}^4} & 0 \\
0 & -\frac{\mu_{b_h}}{c_{b_h}^4} & 0 & \frac{L}{2c_{b_h}^4}
\end{pmatrix}.$$

Its determinant is calculated as

$$\left|\frac{1}{2}\mathcal{H}^{\mathrm{EVB}}\right| = -\frac{\mu_{b_h}}{c_{b_h}^4}\begin{vmatrix} -\frac{\mu_{a_h}}{c_{a_h}^4} & 0 & \frac{M}{2c_{a_h}^4} \\ 0 & -\frac{\mu_{b_h}}{c_{b_h}^4} & 0 \\ \frac{1}{\sigma_{a_h}^2} & \frac{\gamma_h - 2\mu_{a_h}\mu_{b_h}}{\sigma^2} & -\frac{\mu_{a_h}}{c_{a_h}^4} \end{vmatrix} + \frac{L}{2c_{b_h}^4}\begin{vmatrix} \frac{1}{\sigma_{a_h}^2} & \frac{\gamma_h - 2\mu_{a_h}\mu_{b_h}}{\sigma^2} & -\frac{\mu_{a_h}}{c_{a_h}^4} \\ \frac{\gamma_h - 2\mu_{a_h}\mu_{b_h}}{\sigma^2} & \frac{1}{\sigma_{b_h}^2} & 0 \\ -\frac{\mu_{a_h}}{c_{a_h}^4} & 0 & \frac{M}{2c_{a_h}^4} \end{vmatrix}$$
$$= \frac{1}{c_{a_h}^4 c_{b_h}^4}\left(\frac{\mu_{a_h}^2\mu_{b_h}^2}{c_{a_h}^4 c_{b_h}^4} - \frac{M\mu_{b_h}^2}{2\sigma_{a_h}^2 c_{b_h}^4} - \frac{L\mu_{a_h}^2}{2\sigma_{b_h}^2 c_{a_h}^4} + \frac{LM}{4\sigma^4}\left(\frac{\sigma^4}{\sigma_{a_h}^2\sigma_{b_h}^2} - (\gamma_h - 2\mu_{a_h}\mu_{b_h})^2\right)\right).$$

Multiplying both sides of Equation (74) by μ_{a_h} gives

$$\mu_{a_h}^2 = \frac{\sigma_{a_h}^2}{\sigma^2}\gamma_h\widehat{\gamma}_h,$$

and therefore

$$\frac{\mu_{a_h}^2}{\sigma_{a_h}^2} = \frac{\gamma_h\widehat{\gamma}_h}{\sigma^2}. \tag{165}$$

Similarly from Equation (75), we obtain

$$\frac{\mu_{b_h}^2}{\sigma_{b_h}^2} = \frac{\gamma_h\widehat{\gamma}_h}{\sigma^2}. \tag{166}$$

By using Equations (78), (84), (159), (165), and (166), we obtain

$$\left|\frac{1}{2}\mathcal{H}^{\mathrm{EVB}}\right| = \frac{1}{c_{a_h}^4 c_{b_h}^4}\left(\frac{\widehat{\gamma}_h^2}{c_{a_h}^4 c_{b_h}^4} - \frac{\gamma_h\widehat{\gamma}_h}{2\sigma^2}\left(\frac{M\widehat{\delta}^{-2}}{c_{b_h}^4} + \frac{L\widehat{\delta}^2}{c_{a_h}^4}\right) + \frac{LM}{\sigma^4}\left(\widehat{\gamma}_h\gamma_h - \widehat{\gamma}_h^2\right)\right). \tag{167}$$

Since

$$\frac{M\widehat{\delta}^{-2}}{c_{b_h}^4} + \frac{L\widehat{\delta}^2}{c_{a_h}^4} \geq \frac{2\sqrt{LM}}{c_{a_h}^2 c_{b_h}^2}$$

for any $\widehat{\delta}^2 > 0$, Equation (167) is upper-bounded by

$$\left|\frac{1}{2}\mathcal{H}^{\mathrm{EVB}}\right| \leq \frac{1}{c_{a_h}^4 c_{b_h}^4}\left(\frac{\widehat{\gamma}_h^2}{c_{a_h}^4 c_{b_h}^4} - \frac{\gamma_h\widehat{\gamma}_h\sqrt{LM}}{\sigma^2 c_{a_h}^2 c_{b_h}^2} + \frac{LM}{\sigma^4}\left(\widehat{\gamma}_h\gamma_h - \widehat{\gamma}_h^2\right)\right)$$
$$= \frac{\widehat{\gamma}_h}{c_{a_h}^4 c_{b_h}^4}\left(\frac{1}{c_{a_h}^2 c_{b_h}^2} - \frac{\sqrt{LM}}{\sigma^2}\right)\left\{\left(\frac{1}{c_{a_h}^2 c_{b_h}^2} + \frac{\sqrt{LM}}{\sigma^2}\right)\widehat{\gamma}_h - \frac{\sqrt{LM}}{\sigma^2}\gamma_h\right\}. \tag{168}$$

At any *small positive* stationary point, Equation (123) is upper-bounded as

$$\acute{c}_{a_h}^2\acute{c}_{b_h}^2 < \frac{\sigma^2}{\sqrt{LM}}$$

when Equation (125) holds. Therefore, Equation (168) is written as

$$\left|\frac{1}{2}\acute{\mathcal{H}}^{\mathrm{EVB}}\right| \leq C\left\{\left(\frac{1}{\acute{c}_{a_h}^2\acute{c}_{b_h}^2} + \frac{\sqrt{LM}}{\sigma^2}\right)\widehat{\gamma}_h - \frac{\sqrt{LM}}{\sigma^2}\gamma_h\right\},$$

with a positive factor

$$C = \frac{\widehat{\gamma}_h}{\acute{c}_{a_h}^4 \acute{c}_{b_h}^4} \left(\frac{1}{\acute{c}_{a_h}^2 \acute{c}_{b_h}^2} - \frac{\sqrt{LM}}{\sigma^2} \right).$$

Using Equation (31), we have

$$\begin{aligned}
\left| \frac{1}{2} \acute{\mathcal{H}}^{\mathrm{EVB}} \right| &\leq C \left\{ \left(\frac{1}{\acute{c}_{a_h}^2 \acute{c}_{b_h}^2} + \frac{\sqrt{LM}}{\sigma^2} \right) \left(\sqrt{\left(1 - \frac{L\sigma^2}{\gamma_h^2}\right)\left(1 - \frac{M\sigma^2}{\gamma_h^2}\right)} \gamma_h - \frac{\sigma^2}{\acute{c}_{a_h} \acute{c}_{b_h}} \right) \right. \\
&\qquad \left. - \frac{\sqrt{LM}}{\sigma^2} \gamma_h \right\} \\
&= C \left\{ - \frac{\sigma^2}{\acute{c}_{a_h}^3 \acute{c}_{b_h}^3} + \sqrt{\left(1 - \frac{L\sigma^2}{\gamma_h^2}\right)\left(1 - \frac{M\sigma^2}{\gamma_h^2}\right)} \frac{\gamma_h}{\acute{c}_{a_h}^2 \acute{c}_{b_h}^2} - \frac{\sqrt{LM}}{\acute{c}_{a_h} \acute{c}_{b_h}} \right. \\
&\qquad \left. - \frac{\sqrt{LM}}{\sigma^2} \left(1 - \sqrt{\left(1 - \frac{L\sigma^2}{\gamma_h^2}\right)\left(1 - \frac{M\sigma^2}{\gamma_h^2}\right)} \right) \gamma_h \right\} \\
&< \frac{C}{\acute{c}_{a_h} \acute{c}_{b_h}} \left(- \frac{\sigma^2}{\acute{c}_{a_h}^2 \acute{c}_{b_h}^2} + \sqrt{\left(1 - \frac{L\sigma^2}{\gamma_h^2}\right)\left(1 - \frac{M\sigma^2}{\gamma_h^2}\right)} \frac{\gamma_h}{\acute{c}_{a_h} \acute{c}_{b_h}} - \sqrt{LM} \right).
\end{aligned}$$

At the last inequality, we neglected the negative last term in the curly braces.

Using Equation (163), we have

$$\left| \frac{1}{2} \acute{\mathcal{H}}^{\mathrm{EVB}} \right| < -C'(f(\gamma_h) - g(\gamma_h)), \tag{169}$$

where

$$\begin{aligned}
C' &= \frac{\gamma_h^2 C}{2\sigma^2 \acute{c}_{a_h} \acute{c}_{b_h}}, \\
f(\gamma_h) &= \left(1 - \frac{(\sqrt{M} - \sqrt{L})^2 \sigma^2}{\gamma_h^2}\right) + \sqrt{\left(1 - \frac{(L+M)\sigma^2}{\gamma_h^2}\right)^2 - \frac{4LM\sigma^4}{\gamma_h^4}}, \\
g(\gamma_h) &= \sqrt{2\left(1 - \frac{L\sigma^2}{\gamma_h^2}\right)\left(1 - \frac{M\sigma^2}{\gamma_h^2}\right)} \\
&\qquad \times \sqrt{\left(1 - \frac{(L+M)\sigma^2}{\gamma_h^2}\right) + \sqrt{\left(1 - \frac{(L+M)\sigma^2}{\gamma_h^2}\right)^2 - \frac{4LM\sigma^4}{\gamma_h^4}}}.
\end{aligned}$$

Since C', $f(\gamma_h)$, and $g(\gamma_h)$ are positive, the right-hand side of Equation (169) is negative if $f^2(\gamma_h) - g^2(\gamma_h) > 0$. This is shown below.

$$
\begin{aligned}
f^2(\gamma_h) - g^2(\gamma_h) &= \left(\left(1 - \frac{(\sqrt{M}-\sqrt{L})^2\sigma^2}{\gamma_h^2}\right) + \sqrt{\left(1 - \frac{(L+M)\sigma^2}{\gamma_h^2}\right)^2 - \frac{4LM\sigma^4}{\gamma_h^4}} \right)^2 \\
&- 2\left(1 - \frac{L\sigma^2}{\gamma_h^2}\right)\left(1 - \frac{M\sigma^2}{\gamma_h^2}\right)\left(\left(1 - \frac{(L+M)\sigma^2}{\gamma_h^2}\right) + \sqrt{\left(1 - \frac{(L+M)\sigma^2}{\gamma_h^2}\right)^2 - \frac{4LM\sigma^4}{\gamma_h^4}} \right) \\
&= 2\frac{\sqrt{LM}\sigma^2}{\gamma_h^2}\left(2 - \frac{\sqrt{LM}\sigma^2}{\gamma_h^2}\right) \\
&\quad \times \left(\left(1 - \frac{(L+M)\sigma^2}{\gamma_h^2}\right) + \sqrt{\left(1 - \frac{(L+M)\sigma^2}{\gamma_h^2}\right)^2 - \frac{4LM\sigma^4}{\gamma_h^4}} \right) \\
&> 0.
\end{aligned}
$$

Consequently, it holds that $|\acute{\mathcal{H}}^{\mathrm{EVB}}| < 0$. This means that $\acute{\mathcal{H}}^{\mathrm{EVB}}$ has at least one negative and one positive eigenvalues. Therefore, the Hessian of the objective function (106) with respect to $(\mu_{a_h}, \mu_{b_h}, \sigma_{a_h}^2, \sigma_{b_h}^2, c_{a_h}^2, c_{b_h}^2)$ has at least one negative and one positive eigenvalues at any *small positive* stationary point, when Equation (125) holds. This proves the lemma. ■

G.16 Proof of Lemma 25

Substituting Equations (106) and (116) into Equation (126), we have

$$
\begin{aligned}
\Delta_h(\breve{\mu}_{a_h}, \breve{\mu}_{b_h}, \breve{\sigma}_{a_h}^2, \breve{\sigma}_{b_h}^2, \breve{c}_{a_h}^2, \breve{c}_{b_h}^2) &= \mathcal{L}_h^{\mathrm{EVB}}(\breve{\mu}_{a_h}, \breve{\mu}_{b_h}, \breve{\sigma}_{a_h}^2, \breve{\sigma}_{b_h}^2, \breve{c}_{a_h}^2, \breve{c}_{b_h}^2) - (L+M) \\
&= M\log\frac{\breve{c}_{a_h}^2}{\breve{\sigma}_{a_h}^2} + L\log\frac{\breve{c}_{b_h}^2}{\breve{\sigma}_{b_h}^2} + \frac{\breve{\mu}_{a_h}^2 + M\breve{\sigma}_{a_h}^2}{\breve{c}_{a_h}^2} + \frac{\breve{\mu}_{b_h}^2 + L\breve{\sigma}_{b_h}^2}{\breve{c}_{b_h}^2} \\
&\quad + \frac{1}{\sigma^2}\left(-2\gamma_h\breve{\mu}_{a_h}\breve{\mu}_{b_h} + \left(\breve{\mu}_{a_h}^2 + M\breve{\sigma}_{a_h}^2\right)\left(\breve{\mu}_{b_h}^2 + L\breve{\sigma}_{b_h}^2\right)\right) - (L+M). \qquad (170)
\end{aligned}
$$

Substituting Equations (119) and (120) into Equation (170), we have

$$
\Delta_h = M\log\left(\frac{\breve{\mu}_{a_h}^2}{M\breve{\sigma}_{a_h}^2} + 1\right) + L\log\left(\frac{\breve{\mu}_{b_h}^2}{L\breve{\sigma}_{b_h}^2} + 1\right) + \frac{1}{\sigma^2}\left(-2\gamma_h\breve{\mu}_{a_h}\breve{\mu}_{b_h} + LM\breve{c}_{a_h}^2\breve{c}_{b_h}^2\right). \qquad (171)
$$

Substituting Equations (165) and (166) into Equation (171) and using Equation (78), we have

$$
\Delta_h = M\log\left(\frac{\gamma_h}{M\sigma^2}\widehat{\gamma}_h + 1\right) + L\log\left(\frac{\gamma_h}{L\sigma^2}\widehat{\gamma}_h + 1\right) + \frac{1}{\sigma^2}\left(-2\gamma_h\widehat{\gamma}_h + LM\breve{c}_{a_h}^2\breve{c}_{b_h}^2\right). \qquad (172)
$$

Using the bounds (28), Equation (172) is upper-bounded as

$$\begin{aligned}
\Delta_h &< M\log\left(\frac{\gamma_h^2}{M\sigma^2}\left(1-\frac{M\sigma^2}{\gamma_h^2}\right)+1\right)+L\log\left(\frac{\gamma_h^2}{L\sigma^2}\left(1-\frac{M\sigma^2}{\gamma_h^2}\right)+1\right)\\
&\qquad+\frac{1}{\sigma^2}\left(-2\gamma_h\left(\left(1-\frac{\sigma^2 M}{\gamma_h^2}\right)\gamma_h-\frac{\sigma^2\sqrt{M/L}}{\breve{c}_{a_h}\breve{c}_{b_h}}\right)+LM\breve{c}_{a_h}^2\breve{c}_{b_h}^2\right)\\
&= M\log\left(\frac{\gamma_h^2}{M\sigma^2}\right)+L\log\left(\frac{\gamma_h^2}{L\sigma^2}-\frac{M}{L}+1\right)\\
&\qquad+\frac{1}{\sigma^2}\left(-2\gamma_h\left(\gamma_h-\frac{\sigma^2 M}{\gamma_h}-\frac{\sigma^2\sqrt{M/L}}{\breve{c}_{a_h}\breve{c}_{b_h}}\right)+LM\breve{c}_{a_h}^2\breve{c}_{b_h}^2\right)\\
&= M\log\left(\frac{\gamma_h^2}{M\sigma^2}\right)+L\log\left(\frac{\gamma_h^2}{L\sigma^2}-\frac{M}{L}+1\right)+2M+\frac{2\sqrt{M/L}}{\breve{c}_{a_h}\breve{c}_{b_h}}\gamma_h-\frac{2\gamma_h^2}{\sigma^2}+\frac{LM\breve{c}_{a_h}^2\breve{c}_{b_h}^2}{\sigma^2}.
\end{aligned}$$

Since $\sqrt{x^2-y^2} > x-y$ for $x > y > 0$, Equation (122) yields

$$\breve{c}_{a_h}^2\breve{c}_{b_h}^2 \geq \frac{\gamma_h^2-(L+M+\sqrt{LM})\sigma^2}{LM}. \tag{173}$$

Ignoring the positive term $4LM\sigma^4$ in Equation (122), we obtain

$$\breve{c}_{a_h}^2\breve{c}_{b_h}^2 < \frac{\gamma_h^2-(L+M)\sigma^2}{LM}. \tag{174}$$

Equations (173) and (174) result in

$$\sqrt{\frac{\gamma_h^2-(L+M+\sqrt{LM})\sigma^2}{LM}} \leq \breve{c}_{a_h}\breve{c}_{b_h} < \sqrt{\frac{\gamma_h^2-(L+M)\sigma^2}{LM}}.$$

Using these bounds, we obtain

$$\begin{aligned}
\Delta_h &< M\log\left(\frac{\gamma_h^2}{M\sigma^2}\right)+L\log\left(\frac{\gamma_h^2}{L\sigma^2}-\frac{M}{L}+1\right)+2M+\frac{2\sqrt{M/L}}{\sqrt{\frac{\gamma_h^2-(L+M+\sqrt{LM})\sigma^2}{LM}}}\gamma_h\\
&\qquad-\frac{2\gamma_h^2}{\sigma^2}+\gamma_h^2-(L+M)\\
&= M\log\left(\frac{\gamma_h^2}{M\sigma^2}\right)+L\log\left(\frac{\gamma_h^2}{L\sigma^2}-\frac{M}{L}+1\right)+M-L+\frac{2M}{\sqrt{1-\frac{(L+M+\sqrt{LM})\sigma^2}{\gamma_h^2}}}-\frac{\gamma_h^2}{\sigma^2}.
\end{aligned}$$

Using Equations (128), (129), and (130), we obtain Equation (127). ■

G.17 Proof of Lemma 26

For $0 < \alpha \leq 1$ and $\beta \geq 7$, Equation (128) is increasing with respect to α, because

$$\begin{aligned}
\frac{\partial \psi(\alpha,\beta)}{\partial \alpha} &= \log\left(\frac{\beta-1+\alpha}{\alpha}\right) - \left(\frac{\beta-1}{\beta-1+\alpha}\right) - 1 + \frac{(\sqrt{\alpha}+1/2)}{\beta\sqrt{\alpha}\left(1-\frac{(\alpha+\sqrt{\alpha}+1)}{\beta}\right)^{3/2}} \\
&> \log\left(\frac{\beta-1}{\alpha}+1\right) - 2 + \frac{1}{\beta} \\
&\geq \log(\beta) - 2 + \frac{1}{\beta} \\
&> 0.
\end{aligned}$$

Here, we used the numerical estimation that $\log(\beta) - 2 + 1/\beta \approx 0.0888$ when $\beta = 7$, and the fact that $\log(\beta) - 2 + 1/\beta$ is increasing with respect to β when $\beta > 1$.

For $0 < \alpha \leq 1$ and $\beta > 3$, Equation (128) is decreasing with respect to β, because

$$\begin{aligned}
\frac{\partial \psi(\alpha,\beta)}{\partial \beta} &= \frac{1}{\beta} + \frac{\alpha}{(\beta-1+\alpha)} - \frac{\frac{(\alpha+\sqrt{\alpha}+1)}{\beta^2}}{2\left(1-\frac{(\alpha+\sqrt{\alpha}+1)}{\beta}\right)^{3/2}} - 1 \\
&< \frac{1}{\beta} + \frac{\alpha}{(\beta-1+\alpha)} - 1 \\
&= -\frac{(\beta-1+\sqrt{\alpha})(\beta-1-\sqrt{\alpha})}{\beta(\beta-1+\alpha)} \\
&< 0.
\end{aligned}$$

Consequently, if $\psi(1,\widetilde{\beta}) < 0$, it holds that $\psi(\alpha,\beta) < 0$ for any $0 < \alpha \leq 1$ and $\beta \geq \widetilde{\beta}$. The fact that $\psi(1,7) \approx -0.462 < 0$ completes the proof. ■

G.18 Proof of Lemma 29

Since the upper-bound in Equation (28) does not depend on $(c_{a_h}^2, c_{b_h}^2)$, Equation (46) holds.

Since the lower-bound in Equation (28) is nondecreasing with respect to $c_{a_h}c_{b_h}$, substituting Equation (173) into Equation (28) yields

$$\widehat{\gamma}_h \geq \max\left\{0, \left(1-\frac{\sigma^2 M}{\gamma_h^2}\right)\gamma_h - \frac{\sigma^2 M}{\sqrt{\gamma_h^2 - (L+M+\sqrt{LM})\sigma^2}}\right\}.$$

It holds that

$$-\frac{\sigma^2 M}{\gamma_h} > -\frac{\sigma^2 M}{\sqrt{\gamma_h^2 - (L+M+\sqrt{LM})\sigma^2}} > -\frac{\sigma^2 M}{\gamma_h - \sqrt{(L+M+\sqrt{LM})\sigma^2}},$$

where the positive term $(L+M+\sqrt{LM})\sigma^2$ is subtracted in the first inequality and the relation $\sqrt{x^2-y^2} > x-y$ for $x > y > 0$ is used in the second inequality. Then we have

$$\widehat{\gamma}_h > \max\left\{0, \gamma_h - \frac{2\sigma^2 M}{\gamma_h - \sqrt{(L+M+\sqrt{LM})\sigma^2}}\right\},$$

which leads to Equation (47).

Substituting Equation (174) into Equation (31), we obtain

$$\begin{aligned}\widehat{\gamma}_h &< \sqrt{\left(1-\frac{\sigma^2 L}{\gamma_h^2}\right)\left(1-\frac{\sigma^2 M}{\gamma_h^2}\right)}\gamma_h - \frac{\sigma^2\sqrt{LM}}{\sqrt{\gamma_h^2-(L+M)\sigma^2}} \\ &< \sqrt{\left(1-\frac{\sigma^2 L}{\gamma_h^2}\right)\left(1-\frac{\sigma^2 M}{\gamma_h^2}\right)}\gamma_h - \frac{\sigma^2\sqrt{LM}}{\gamma_h},\end{aligned}$$

where the positive term $(L+M)\sigma^2$ is ignored in the second inequality. This gives Equation (48), and completes the proof. ■

Appendix H. Illustration of EVB Objective Function

Here we illustrate the EVB objective function (106). Let us consider a partially minimized objective function:

$$\tilde{\mathcal{L}}_h^{\mathrm{EVB}}(c_{a_h}c_{b_h}) = \min_{(\mu_{a_h},\mu_{b_h},\sigma_{a_h}^2,\sigma_{b_h}^2)} \mathcal{L}_h^{\mathrm{EVB}}(\mu_{a_h},\mu_{b_h},\sigma_{a_h}^2,\sigma_{b_h}^2,c_{a_h}c_{b_h},c_{a_h}c_{b_h}). \tag{175}$$

According to Lemma 19, the infimum at the *null* local minimizer is given by

$$\lim_{c_{a_h}c_{b_h}\to 0} \tilde{\mathcal{L}}_h^{\mathrm{EVB}}(c_{a_h}c_{b_h}) = \mathring{\underline{\mathcal{L}}}_h^{\mathrm{EVB}} = L+M. \tag{176}$$

Figure 13 depicts the partially minimized objective function (175) when $L=M=H=1, \sigma^2=1$, and $V = 1.5, 2.0, 2.1, 2.7$. Corollary 1 provides the exact values for drawing these graphs. The *large* and the *small positive* stationary points, specified by Equations (122) and (123), respectively, are also plotted in the graphs if they exist. When

$$V = 1.5 \left(< 2 = (\sqrt{L}+\sqrt{M})\sigma\right),$$

Equation (121) does not hold. In this case, the objective function (175) has no stationary point as Lemma 22 states (the upper-left graph of Figure 13). The curve is identical for $0 \le V < 2.0$.

When $V = 2.0$ (the upper-right graph), Equation (124) holds. In this case, the objective function (175) has a stationary point at $c_{a_h}c_{b_h} = 1$. This corresponds to the coincident *large* and *small positive* stationary point. Still no local minimum exists.

When $V = 2.1$ (the lower-left graph), Equation (125) holds. In this case, there exists a *large positive* stationary point (which is a local minimum) at $c_{a_h}c_{b_h} \approx 1.37$, as well as a *small positive* stationary point (which is a local maximum) at $c_{a_h}c_{b_h} \approx 0.73$. However, we see that

$$\tilde{\mathcal{L}}_h^{\mathrm{EVB}}(1.37) \approx 2.24 > 2 = \mathring{\underline{\mathcal{L}}}_h^{\mathrm{EVB}}.$$

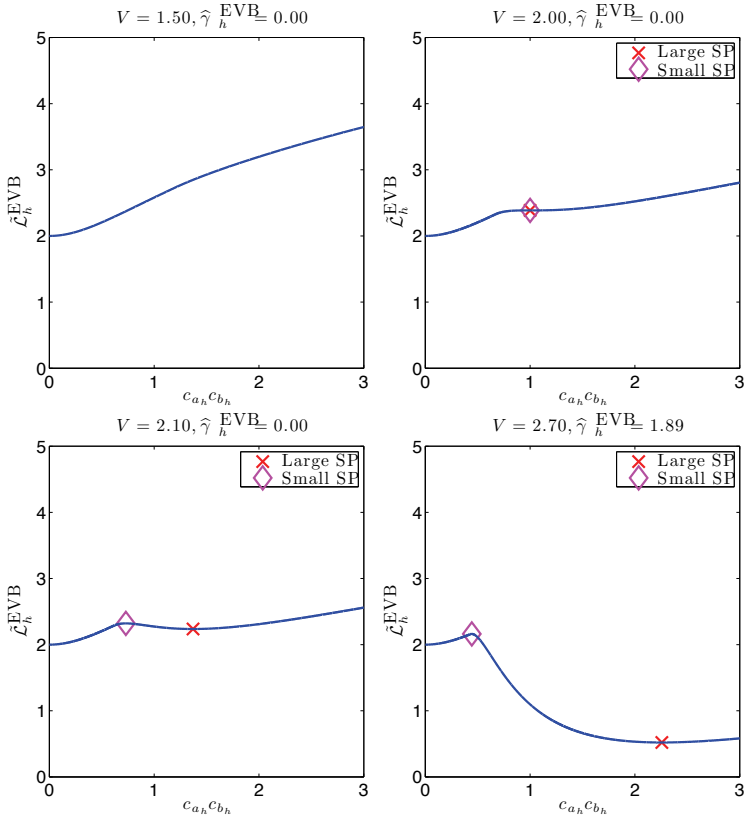


Figure 13: Illustration of the partially minimized objective function (175) when $L = M = H = 1$, $\sigma^2 = 1$, and $V = 1.5, 2.0, 2.1, 2.7$. The convergence $\tilde{L}_h^{\mathrm{EVB}}(c_{a_h} c_{b_h}) \to L + M \ (= 2)$ as $c_{a_h} c_{b_h} \to 0$ is observed (see Equation (176)). 'Large SP' and 'Small SP' indicate the *large* and the *small positive* stationary points, respectively.

Therefore, the *null* local minimizer ($c_{a_h} c_{b_h} \to 0$) is still global, resulting in $\widehat{\gamma}_h^{\mathrm{EVB}} = 0$.

When $V = 2.7$ (the lower-right graph), $\gamma_h \geq \sqrt{7M} \cdot \sigma$ holds. As Lemma 28 states, a *large positive* stationary point at $c_{a_h} c_{b_h} \approx 2.26$ gives the global minimum:

$$\tilde{L}_h^{\mathrm{EVB}}(2.26) \approx 0.52 < 2 = \mathring{\underline{L}}_h^{\mathrm{EVB}},$$

resulting in a *positive* output $\widehat{\gamma}_h^{\mathrm{EVB}} \approx 1.89$.

Appendix I. Derivation of Equations (57) and (58)

Let $p(\boldsymbol{v}|\boldsymbol{\theta})$ be a model distribution, where $\boldsymbol{v}$ is a random variable and $\boldsymbol{\theta} \in \mathbb{R}^d$ is a d-dimensional parameter vector. The *Jeffreys non-informative prior* (Jeffreys, 1946) is defined as

$$\phi^{\mathrm{Jef}}(\boldsymbol{\theta}) \propto \sqrt{|\mathcal{F}|}, \tag{177}$$

where $\mathcal{F} \in \mathbb{R}^{d \times d}$ is the Fisher information matrix defined by

$$\mathcal{F}_{jk} = \int \frac{\partial \log p(\boldsymbol{v}|\boldsymbol{\theta})}{\partial \theta_j} \frac{\partial \log p(\boldsymbol{v}|\boldsymbol{\theta})}{\partial \theta_k} p(\boldsymbol{v}|\boldsymbol{\theta}) d\boldsymbol{v}. \tag{178}$$

Let us first derive the Jeffreys prior for the non-factorizing model:

$$p_U(V|U) \propto \exp\left(-\frac{1}{2\sigma^2}(V-U)^2\right). \tag{179}$$

In this model, the parameter vector is one-dimensional: $\theta = U$. Since

$$\frac{\partial \log p_U(V|U)}{\partial U} = \frac{V-U}{\sigma^2},$$

the Fisher information (178) is given by

$$\mathcal{F}_U = \frac{1}{\sigma^2}.$$

This is constant over the parameter space. Therefore, the Jeffreys prior (177) for the model (179) is given by Equation (57).

Let us move on to the MF model:

$$p_{A,B}(V|A,B) \propto \exp\left(-\frac{1}{2\sigma^2}(V-AB)^2\right). \tag{180}$$

In this model, the parameter vector is $\boldsymbol{\theta} = (A,B)$. Since

$$\frac{\partial \log p_{A,B}(Y|A,B)}{\partial A} = \frac{1}{\sigma^2}(Y-AB)B,$$
$$\frac{\partial \log p_{A,B}(Y|A,B)}{\partial B} = \frac{1}{\sigma^2}(Y-AB)A,$$

the Fisher information matrix is given by

$$\mathcal{F}_{A,B} = \frac{1}{\sigma^2}\begin{pmatrix} B^2 & AB \\ AB & A^2 \end{pmatrix},$$

whose eigenvalues are $\sigma^{-2}\sqrt{A^2+B^2}$ and 0.

The common (over the parameter space) zero-eigenvalue comes from the invariance of the MF model (180) under the transform $(A,B) \to (sA, s^{-1}B)$ for any $s > 0$. Neglecting it, we re-define the Jeffreys prior by

$$\phi^{\mathrm{Jef}}(\boldsymbol{\theta}) \propto \sqrt{\textstyle\prod_{j=1}^{d-1} \lambda_j},$$

where λ_j is the j-th largest eigenvalue of the Fisher information matrix. Thus, we obtain Equation (58).

■

References

T. W. Anderson. *An Introduction to Multivariate Statistical Analysis*. Wiley, New York, second edition, 1984.

H. Attias. Inferring parameters and structure of latent variable models by variational Bayes. In *Proceedings of the Fifteenth Conference Annual Conference on Uncertainty in Artificial Intelligence (UAI-99)*, pages 21–30, San Francisco, CA, 1999. Morgan Kaufmann.

P. Baldi and S. Brunak. *Bioinformatics: The Machine Learning Approach*. MIT Press, Cambridge, MA, USA, 1998.

P. F. Baldi and K. Hornik. Learning in linear neural networks: A survey. *IEEE Transactions on Neural Networks*, 6(4):837–858, 1995.

A. J. Baranchik. Multiple regression and estimation of the mean of a multivariate normal distribution. Technical Report 51, Department of Statistics, Stanford University, Stanford, CA, USA, 1964.

J. Besag. On the statistical analysis of dirty pictures. *Journal of the Royal Statistical Society B*, 48: 259–302, 1986.

C. M. Bishop. *Pattern Recognition and Machine Learning*. Springer, New York, NY, USA, 2006.

J. F. Cai, E. J. Candes, and Z. Shen. A singular value thresholding algorithm for matrix completion. *SIAM Journal on Optimization*, 20(4):1956–1982, 2010.

O. Chapelle and Z. Harchaoui. A machine learning approach to conjoint analysis. In *Advances in Neural Information Processing Systems*, volume 17, pages 257–264, 2005.

W. Chu and Z. Ghahramani. Probabilistic models for incomplete multi-dimensional arrays. In *Proceedings of International Conference on Artificial Intelligence and Statistics*, 2009.

A. Cichocki, R. Zdunek, A. H. Phan, and S. Amari. *Nonnegative Matrix and Tensor Factorizations: Applications to Exploratory Multi-way Data Analysis and Blind Source Separation*. John Wiley & Sons, West Sussex, UK, 2009.

M. J. Daniels and R. E. Kass. Shrinkage estimators for covariance matrices. *Biometrics*, 57(4): 1173–1184, 2001.

B. Efron and C. Morris. Stein's estimation rule and its competitors—an empirical Bayes approach. *Journal of the American Statistical Association*, 68:117–130, 1973.

S. Funk. Try this at home. http://sifter.org/˜simon/journal/20061211.html, 2006.

A. Gelman. Parameterization and Bayesian modeling. *Journal of the American Statistical Association*, 99:537–545, 2004.

Y. Y. Guo and N. Pal. A sequence of improvements over the James-Stein estimator. *Journal of Multivariate Analysis*, 42(2):302–317, 1992.

K. Hayashi, J. Hirayama, and S. Ishii. Dynamic exponential family matrix factorization. In T. Theeramunkong, B. Kijsirikul, N. Cercone, and T.-B. Ho, editors, *Advances in Knowledge Discovery and Data Mining*, volume 5476 of *Lecture Notes in Computer Science*, pages 452–462, Berlin, 2009. Springer.

H. Hotelling. Relations between two sets of variates. *Biometrika*, 28(3–4):321–377, 1936.

A. Hyvärinen, J. Karhunen, and E. Oja. *Independent Component Analysis*. Wiley, New York, 2001.

W. James and C. Stein. Estimation with quadratic loss. In *Proceedings of the 4th Berkeley Symposium on Mathematical Statistics and Probability*, volume 1, pages 361–379, Berkeley, CA., USA, 1961. University of California Press.

H. Jeffreys. An invariant form for the prior probability in estimation problems. In *Proceedings of the Royal Society of London. Series A, Mathematical and Physical Sciences*, volume 186, pages 453–461, 1946.

J. A. Konstan, B. N. Miller, D. Maltz, J. L. Herlocker, L. R. Gordon, and J. Riedl. Grouplens: Applying collaborative filtering to Usenet news. *Communications of the ACM*, 40(3):77–87, 1997.

S. Kullback and R. A. Leibler. On information and sufficiency. *Annals of Mathematical Statistics*, 22:79–86, 1951.

O. Ledoit and M. Wolf. A well-conditioned estimator for large dimensional covariance matrices. *Journal of Multivariate Analysis*, pages 365–411, 2004.

D. D. Lee and S. Seung. Learning the parts of objects by non-negative matrix factorization. *Nature*, 401(6755):788–791, 1999.

E. L. Lehmann. *Theory of Point Estimation*. Wiley, New York, 1983.

Y. J. Lim and T. W. Teh. Variational Bayesian approach to movie rating prediction. In *Proceedings of KDD Cup and Workshop*, 2007.

D. J. C. MacKay. Bayesian interpolation. *Neural Computation*, 4(2):415–447, 1992.

D. J. C. MacKay. *Information Theory, Inference, and Learning Algorithms*. Cambridge University Press, Cambridge, UK, 2003.

A. W. Marshall, I. Olkin, and B. C. Arnold. *Inequalities: Theory of Majorization and Its Applications, Second Edition*. Springer, 2009.

S. Nakajima and M. Sugiyama. Implicit regularization in variational Bayesian matrix factorization. In A. T. Joachims and J. Fürnkranz, editors, *Proceedings of 27th International Conference on Machine Learning (ICML2010)*, Haifa, Israel, Jun. 21–25 2010.

S. Nakajima and S. Watanabe. Variational Bayes solution of linear neural networks and its generalization performance. *Neural Computation*, 19(4):1112–1153, 2007.

R. M. Neal. *Bayesian Learning for Neural Networks*. Springer, 1996.

A. Paterek. Improving regularized singular value decomposition for collaborative filtering. In *Proceedings of KDD Cup and Workshop*, 2007.

T. Raiko, A. Ilin, and J. Karhunen. Principal component analysis for large scale problems with lots of missing values. In J. Kok, J. Koronacki, R. Lopez de Mantras, S. Matwin, D. Mladenic, and A. Skowron, editors, *Proceedings of the 18th European Conference on Machine Learning*, volume 4701 of *Lecture Notes in Computer Science*, pages 691–698, Berlin, 2007. Springer-Verlag.

G. R. Reinsel and R. P. Velu. *Multivariate Reduced-Rank Regression: Theory and Applications*. Springer, New York, 1998.

J. D. M. Rennie and N. Srebro. Fast maximum margin matrix factorization for collaborative prediction. In *Proceedings of the 22nd International Conference on Machine learning*, pages 713–719, 2005.

J. Rissanen. Stochastic complexity and modeling. *Annals of Statistics*, 14(3):1080–1100, 1986.

R. Rosipal and N. Krämer. Overview and recent advances in partial least squares. In C. Saunders, M. Grobelnik, S. Gunn, and J. Shawe-Taylor, editors, *Subspace, Latent Structure and Feature Selection Techniques*, volume 3940 of *Lecture Notes in Computer Science*, pages 34–51, Berlin, 2006. Springer.

R. Salakhutdinov and A. Mnih. Probabilistic matrix factorization. In J. C. Platt, D. Koller, Y. Singer, and S. Roweis, editors, *Advances in Neural Information Processing Systems 20*, pages 1257–1264, Cambridge, MA, 2008. MIT Press.

B. Schölkopf and A. J. Smola. *Learning with Kernels*. MIT Press, Cambridge, MA, 2002.

P. Y.-S. Shao and W. E. Strawderman. Improving on the James-Stein positive-part estimator. *The Annals of Statistics*, 22:1517–1538, 1994.

N. Srebro and T. Jaakkola. Weighted low rank approximation. In T. Fawcett and N. Mishra, editors, *Proceedings of the Twentieth International Conference on Machine Learning*. AAAI Press, 2003.

N. Srebro, J. Rennie, and T. Jaakkola. Maximum margin matrix factorization. In *Advances in NIPS*, volume 17, 2005.

C. Stein. Inadmissibility of the usual estimator for the mean of a multivariate normal distribution. In *Proc. of the 3rd Berkeley Symp. on Math. Stat. and Prob.*, pages 197–206, 1956.

C. Stein. Estimation of a covariance matrix. In *Rietz Lecture, 39th Annual Meeting IMS*, 1975.

W. E. Strawderman. Proper Bayes minimax estimators of the multivariate normal mean. *Annals of Mathematical Statistics*, 42:385–388, 1971.

D. Tao, M. Song, X. Li, J. Shen, J. Sun, X. Wu, C. Faloutsos, and S. J. Maybank. Tensor approach for 3-D face modeling. *IEEE Transactions on Circuits and Systems for Video Technology*, 18(10): 1397–1410, 2008.

K. Watanabe and S. Watanabe. Stochastic complexities of Gaussian mixtures in variational Bayesian approximation. *Journal of Machine Learning Research*, 7:625–644, 2006.

S. Watanabe. Algebraic analysis for nonidentifiable learning machines. *Neural Computation*, 13 (4):899–933, 2001.

S. Watanabe. *Algebraic Geometry and Statistical Learning*. Cambridge University Press, Cambridge, UK, 2009.

D. Wipf and S. Nagarajan. A new view of automatric relevance determination. In J. C. Platt, D. Koller, Y. Singer, and S. Roweis, editors, *Advances in Neural Information Processing Systems 20*, pages 1625–1632, Cambridge, MA, 2008. MIT Press.

H. Wold. Estimation of principal components and related models by iterative least squares. In P. R. Krishnaiah, editor, *Multivariate Analysis*, pages 391–420. Academic Press, New York, NY, USA, 1966.

K. J. Worsley, J-B. Poline, K. J. Friston, and A. C. Evanss. Characterizing the response of PET and fMRI data using multivariate linear models. *NeuroImage*, 6(4):305–319, 1997.

K. Yamazaki and S. Watanabe. Singularities in mixture models and upper bounds of stochastic complexity. *Neural Networks*, 16(7):1029–1038, 2003.

K. Yu, V. Tresp, and A. Schwaighofer. Learning Gaussian processes from multiple tasks. In *Proceedings of the Twenty-Second International Conference on Machine learning*, pages 1012–1019, 2005.

S. Yu, J. Bi, and J. Ye. Probabilistic interpretations and extensions for a family of 2D PCA-style algorithms. In *KDD Workshop on Data Mining using Matrices and Tensors*, 2008.

Journal of Machine Learning Research 12 (2011) 2649-2680 Submitted 6/09; Revised 5/11; Published 9/11

Bayesian Co-Training

Shipeng Yu SHIPENG.YU@SIEMENS.COM
Balaji Krishnapuram BALAJI.KRISHNAPURAM@SIEMENS.COM
Business Intelligence and Analytics
Siemens Medical Solutions USA, Inc.
51 Valley Stream Parkway
Malvern, PA 19355, USA

Rómer Rosales ROMERR@YAHOO-INC.COM
Yahoo! Labs
4401 Great America Pkwy
Santa Clara, CA 95054, USA

R. Bharat Rao BHARAT.RAO@SIEMENS.COM
Business Intelligence and Analytics
Siemens Medical Solutions USA, Inc.
51 Valley Stream Parkway
Malvern, PA 19355, USA

Editor: Carl Edward Rasmussen

Abstract

Co-training (or more generally, co-regularization) has been a popular algorithm for semi-supervised learning in data with two feature representations (or views), but the fundamental assumptions underlying this type of models are still unclear. In this paper we propose a Bayesian undirected graphical model for co-training, or more generally for semi-supervised multi-view learning. This makes explicit the previously unstated assumptions of a large class of co-training type algorithms, and also clarifies the circumstances under which these assumptions fail. Building upon new insights from this model, we propose an improved method for co-training, which is a novel co-training kernel for Gaussian process classifiers. The resulting approach is convex and avoids local-maxima problems, and it can also automatically estimate how much each view should be trusted to accommodate noisy or unreliable views. The Bayesian co-training approach can also elegantly handle data samples with missing views, that is, some of the views are not available for some data points at learning time. This is further extended to an active sensing framework, in which the missing (sample, view) pairs are actively acquired to improve learning performance. The strength of active sensing model is that one actively sensed (sample, view) pair would improve the joint multi-view classification on all the samples. Experiments on toy data and several real world data sets illustrate the benefits of this approach.

Keywords: co-training, multi-view learning, semi-supervised learning, Gaussian processes, undirected graphical models, active sensing

1. Introduction

In machine learning, data samples may sometimes be characterized in multiple ways. For instance in web page classification, the web pages can be described both in terms of the textual content in each page and the hyperlink structure between them; for cancer diagnosis where the goal is to determine

if the patient has cancer or not, multiple medical imaging techniques (such as CT, Ultrasound and MRI) might be considered to collect complete characteristic of the patient from different perspectives. For learning under such a setting, it has been shown in Dasgupta et al. (2001) that the error rate on unseen test samples can be upper bounded by the disagreement between the classification-decisions obtained from independent characterizations (i.e., *views*) of the data. Thus, in the web page example, *misclassification rate* can be indirectly minimized by reducing the *rate of disagreement* between hyperlink-based and content-based classifiers, provided these characterizations are independent conditional on the class label.

As a completely new learning principle, multi-view consensus learning has been the subject of a large body of research recently. This type of methods were originally developed for semi-supervised learning, where class labels are expensive to obtain but unlabeled data are cheap and abundantly available, such as in web page classification. When the data samples can be characterized in multiple views, the disagreement between the class labels suggested by different views can be computed even when using unlabeled data. Therefore, a natural strategy for using unlabeled data to minimize the misclassification rate is to enforce *consistency* between the classification decisions based on several independent characterizations of the unlabeled samples. For brevity, unless otherwise specified, we shall use the term *co-training* to describe the entire genre of methods that rely upon this intuition, although strictly it should only refer to the original algorithm of Blum and Mitchell (1998).

In this pioneering paper, Blum and Mitchell introduced an iterative, alternating co-training method, which works in a bootstrap mode by repeatedly adding pseudo-labeled unlabeled samples into the pool of labeled samples, retraining the classifiers for each view, and pseudo-labeling additional unlabeled samples where at least one view is confident about its decision. The paper provided PAC-style guarantees that if (a) there exist weakly useful classifiers on each view of the data, and (b) these characterizations of the sample are conditionally independent given the class label, then the co-training algorithm can use the unlabeled data to learn arbitrarily strong classifiers. Later Balcan et al. (2004) tried to reduce the strong theoretical requirements, and they showed that co-training would be useful if (a) there exist low error rate classifiers on each view, (b) these classifiers never make mistakes in classification when they are confident about their decisions, and (c) the two views are not too highly correlated, in the sense that there would be at least some cases where one view makes confident classification decisions while the classifier on the other view does not have much confidence in its own decision. While each of these theoretical guarantees is intriguing and theoretically interesting, they are also rather unrealistic in many application domains. The assumption that classifiers do not make mistakes when they are confident and that of class conditional independence are rarely satisfied in practice. Empirical studies of co-training on many applications show mixed results. See, for instance, Pierce and Cardie (2001) and Kiritchenko and Matwin (2002); Hwa et al. (2003).

A strongly related algorithm is the co-EM algorithm from Nigam and Ghani (2000), which extends the original bootstrap approach of the co-training algorithm to operate simultaneously on all unlabeled samples in an iterative batch mode. Brefeld and Scheffer (2004) used this idea with SVMs as base classifiers, and subsequently in unsupervised learning in Bickel and Scheffer (2005). However, co-EM also suffers from local maxima problems, and while each iteration's optimization step is clear, the co-EM is not really an expectation maximization algorithm (i.e., it lacks a clearly defined overall log-likelihood that monotonically improves across iterations).

In recent years, some co-training algorithms jointly optimize an objective function which includes misclassification penalties (i.e., loss terms) for classifiers from each view, and a regulariza-

tion term that penalizes lack of agreement between the classification decisions of the different views. This *co-regularization* approach has become the dominant strategy for exploiting the intuition behind multi-view consensus learning, rendering obsolete earlier alternating-optimization strategies. Krishnapuram et al. (2004) proposed an approach for two-view consensus learning based on simultaneously learning multiple classifiers by maximizing an objective function which penalized misclassifications by any individual classifier, and included a regularization term that penalized a high level of disagreement between different views. This co-regularization framework improves upon the co-training and co-EM algorithms by maximizing a convex objective function; however the algorithm still depends on an alternating optimization that optimizes one view at a time. This approach was later adapted to two-view spectral clustering in de Sa (2005). The two-view co-regularization approach was subsequently adopted by Sindhwani et al. (2005), Brefeld et al. (2006), Sindhwani and Rosenberg (2008) and Farquhar et al. (2005) for semi-supervised classification and regression based on the reproducing kernel Hilbert space (RKHS). In these approaches a new co-regularization term is added to the objective function which is based on the disagreement of the two views. Representer theorem still holds and solutions can be easily derived by direct optimization. However, it is unclear how to set the regularization parameters (i.e., to control the weight of the co-regularization term). Theoretical analysis of this and other types of algorithms can be found in Balcan and Blum (2006), Sridharan and Kakade (2008), Wang and Zhou (2007) and Wang and Zhou (2010).

Much of these previous work on co-training has been somewhat ad-hoc in nature. Although some algorithms were empirically successful in specific applications, it was not always clear what precise assumptions were made, what was being optimized overall or why they worked well. In this paper we propose a principled undirected graphical model for co-training which we call the *Bayesian co-training*, and show that co-regularization algorithms provide one way for maximum-likelihood (ML) learning under this probabilistic model. By explicitly highlighting previously unstated assumptions, Bayesian co-training provides a deeper understanding of the co-regularization framework, and we are also able to discuss certain fundamental limitations of multi-view consensus learning. Summarizing our algorithmic contributions, we show that co-regularization is exactly equivalent to the use of a novel *co-training kernel* for *support vector machines* (SVMs) and *Gaussian processes* (GP), thus allowing one to leverage the large body of available literature for these algorithms. The kernel is intrinsically *non-stationary*, that is, the level of similarity between any pair of samples depends on *all* the available samples, whether labeled or unlabeled, thus promoting semi-supervised learning. Therefore, this approach is significantly simpler and more efficient than the alternating-optimization that is used in previous co-regularization implementations. Furthermore, we can automatically estimate how much each view should be trusted, and thus accommodate noisy or unreliable views.

The basic idea of Bayesian co-training was published in a short conference paper by Yu et al. (2008). In the current paper we have all the derivation details and more discussions to its related models. More importantly, we extend the Bayesian co-training model to handle data samples with missing views (i.e., some views are missing for certain data samples), and introduce a novel application called the *active sensing*. This makes the current paper significantly different from its conference version.

Active sensing aims to efficiently choose, among all the missing features (grouped in views), what views *and* samples to additionally acquire (or sense) to improve the overall learning performance. This is different from the typical *active learning*, which addresses the problem of efficiently choosing data samples to be labeled in order to improve overall learning performance. From a can-

cer diagnosis perspective, active learning is equivalent to choosing patients to do a biopsy such that the tumor is correctly diagnosed (benign/malignant), whereas active sensing is targeting at collecting (the not-yet-been-collected) medical imaging features (of, e.g., CT, Ultrasound and MRI) from some patients such that all the patients can be better diagnosed. This is important, since a patient does not undergo all possible tests at once (due to various side effects such as radiation and contrast), but these tests are selected based on the evidence collected up to a particular point. This is normally referred to as *differential diagnosis*. Another example is in land mine detection in a sensor network. We may have different types of sensors (as different views) deployed at one location, but some sensors may not be available for all locations due to high cost. So active sensing is to decide which location and which type of sensor we should additionally consider to achieve better detection accuracy. Formulated within the Bayesian co-training framework, two approaches will be discussed for efficiently choosing the (sample, view) pair, based on the mutual information (involving various random variables) and on the predictive uncertainty, respectively.

This active sensing problem is similar to active feature acquisition—see, for example, Melville et al. (2004) and Bilgic and Getoor (2007)—but there is a clear difference. Previous feature acquisition only considers one sample at a time, that is, when one sample is in consideration, the other samples will not be affected. But in active sensing, one actively acquired (sample, view) pair will improve the classification performance of *all* the unlabeled samples via a co-training setting. A related yet different problem was considered in Krause et al. (2008) to identify the optimal spatial locations for placing a single type of sensor to model spatially varying phenomena; however, this work addressed the use of a single type of sensor, and do not consider the scenario of multiple views.

The rest of the paper is organized as follows. We introduce the Bayesian co-training model in Section 2, covering both the undirected graphical model and various marginalizations. Co-training kernel will be discussed in detail to highlight the insight of the approach. The model is extended to handle missing views in Section 4, and this provides the basics for the active sensing solution. The active sensing problem is discussed in Section 5, in which we provide two methods for deciding which incomplete samples should be further characterized, and which sensors should be deployed on them. Experimental results are provided in Section 6, including both some toy problems and real world problems on web page classification and differential diagnosis. We conclude with a brief discussion and future work in Section 7.

2. Bayesian Co-Training

We start from an undirected graphical model for single-view learning with Gaussian processes, and then present Bayesian co-training which is a new undirected graphical model for multi-view learning.

2.1 Single-View Learning with Gaussian Processes

A Gaussian process (GP) defines a nonparametric prior over functions in Bayesian statistics (Rasmussen and Williams, 2006). A random, real-valued function $f : \mathbb{R}^d \to \mathbb{R}$ follows a GP, denoted by $f \sim \mathcal{GP}(h,\kappa)$, if for any finite number of data points $\mathrm{x}_1,\ldots,\mathrm{x}_n \in \mathbb{R}^d$, $\mathrm{f} = \{f(\mathrm{x}_i)\}_{i=1}^n$ follows a multivariate Gaussian distribution $\mathcal{N}(\mathrm{h},\mathrm{K})$ with mean vector $\mathrm{h} = \{h(\mathrm{x}_i)\}_{i=1}^n$ and covariance matrix defined as $\mathrm{K} = \{\kappa(\mathrm{x}_i,\mathrm{x}_j)\}_{i,j=1}^n$. The functions h and κ are called the mean function and the covariance function, respectively. Conventionally, the mean function is fixed as $h \equiv 0$, and the co-

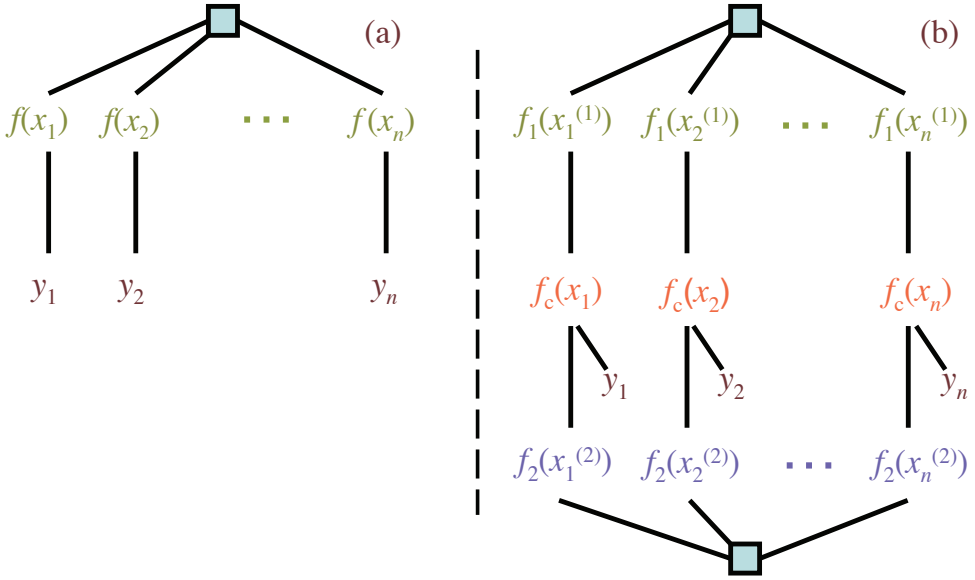


Figure 1: Factor graph for (a) one-view and (b) two-view models.

variance function κ is assumed to take a parametric (and usually stationary) form (e.g., the squared exponential function $\kappa(\mathrm{x}_i,\mathrm{x}_j) = \exp(-\frac{1}{2\rho^2}\|\mathrm{x}_i-\mathrm{x}_j\|^2)$ with $\rho > 0$ a *width* parameter).

In a single-view, supervised learning scenario, an output or target y_i is given for each observation x_i (e.g., for regression $y_i \in \mathbb{R}$ and for classification $y_i \in \{-1,+1\}$). In the GP model we assume there is a latent function f underlying the output,

$$p(y_i|\mathrm{x}_i) = \int p(y_i|f,\mathrm{x}_i)\,p(f)\,df = \int p(y_i|f(\mathrm{x}_i))\,p(f)\,df,$$

with the GP prior $p(f) = \mathcal{GP}(h,\kappa)$. Given the latent function f, for regression $p(y_i|f(\mathrm{x}_i))$ takes a Gaussian noise model $\mathcal{N}(y_i|f(\mathrm{x}_i),\sigma^2)$, with $\sigma > 0$ a parameter for the noise level; for classification $p(y_i|f(\mathrm{x}_i))$ takes the form of a sigmoid function $\lambda(y_i f(\mathrm{x}_i))$. For instance for GP logistic regression, we have $\lambda(z) = (1+\exp(-z))^{-1}$. See Rasmussen and Williams (2006) for more details on this.

The dependency structure of the single-view GP model can be shown as an undirected graph as in Figure 1(a). The maximal cliques of the graphical model are the fully connected nodes $\{f(\mathrm{x}_1),\ldots,f(\mathrm{x}_n)\}$ and the pairs $\{y_i, f(\mathrm{x}_i)\}$, $i = 1,\ldots,n$. Therefore, the joint probability of random variables $\mathrm{f} = \{f(\mathrm{x}_i)\}$ and $\mathrm{y} = \{y_i\}$ is defined as

$$p(\mathrm{f},\mathrm{y}) = \frac{1}{Z}\psi(\mathrm{f})\prod_{i=1}^{n}\psi(y_i, f(\mathrm{x}_i)),$$

with potential functions $\psi(\mathrm{f}) = \exp(-\frac{1}{2}\mathrm{f}^\top \mathrm{K}^{-1}\mathrm{f})$, and[1]

$$\psi(y_i, f(\mathrm{x}_i)) = \begin{cases} \exp(-\frac{1}{2\sigma^2}\|y_i - f(\mathrm{x}_i)\|^2) & \text{for regression,} \\ \lambda(y_i f(\mathrm{x}_i)) & \text{for classification.} \end{cases} \quad (1)$$

The normalization factor Z hereafter is defined such that the joint probability sums to 1.

1. The definition of ψ in this paper has been overloaded to simplify notation, but its meaning should be clear from the function arguments.

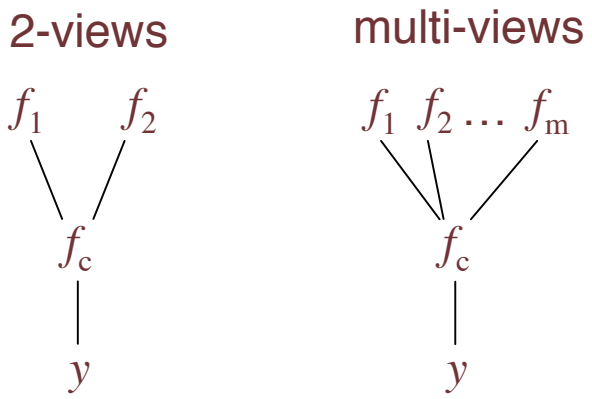


Figure 2: Factor graph in the functional space for 2-view and multi-view learning.

2.2 Undirected Graphical Model for Multi-View Learning

In multi-view learning, suppose we have m different views of a same set of n data samples. Let $\mathrm{x}_i^{(j)} \in \mathbb{R}^{d_j}$ be the features for the ith sample obtained using the jth view, where d_j is the dimensionality of the input space for view j. Note that subscripts index the data sample, and superscripts (with round brackets) index the view. Then the vector $\mathrm{x}_i \triangleq (\mathrm{x}_i^{(1)}, \ldots, \mathrm{x}_i^{(m)})$ is the complete representation of the ith data sample, and $\mathrm{x}^{(j)} \triangleq (\mathrm{x}_1^{(j)}, \ldots, \mathrm{x}_n^{(j)})$ represents all sample observations for the jth view. As in the single-view learning, let $\mathrm{y} = [y_1, \ldots, y_n]^\top$ be the output where y_i is the single output assigned to the ith data point.

One can certainly concatenate the multiple views of the data into a single view, and apply a single-view GP model. But the basic idea of multi-view learning is to introduce *one function per view*, which only uses the features from that specific view to make predictions. Multi-view learning then jointly optimizes these functions such that they come to a consensus. From a GP perspective, let f_j denote the latent function for the jth view (i.e., using features only from view j), and let $f_j \sim \mathcal{GP}(0, \kappa_j)$ be its GP prior in view j with covariance function κ_j. Since one data sample i has only one single label y_i even though it has multiple features from the multiple views (i.e., latent function value $f_j(\mathrm{x}_i^{(j)})$ for view j), the label y_i should depend on *all* of these latent function values for data sample i.

The challenge here is to make this dependency explicit in a graphical model. We tackle this problem by introducing a new latent function, the *consensus function* f_c, to ensure conditional independence between the output y and the m latent functions $\{f_j\}$ for the m views. See Figure 1(b) for the undirected graphical model for multi-view learning. At the functional level, the output y depends *only* on f_c, and latent functions $\{f_j\}$ depend on each other *only via* the consensus function f_c (see Figure 2 for the factor graphs for 2-view and multi-view cases). That is, the joint probability is defined as:

$$p(y, f_c, f_1, \ldots, f_m) = \frac{1}{Z}\psi(y, f_c)\prod_{j=1}^{m}\psi(f_j, f_c), \tag{2}$$

with some potential functions ψ. In the ground network where we have n data samples, let $\mathrm{f}_c = \{f_c(\mathrm{x}_i)\}_{i=1}^n$ and $\mathrm{f}_j = \{f_j(\mathrm{x}_i^{(j)})\}_{i=1}^n$ be the functional values for the consensus view and the jth view,

respectively. The graphical model leads to the following factorization:

$$p(\mathrm{y}, \mathrm{f}_c, \mathrm{f}_1, \ldots, \mathrm{f}_m) = \frac{1}{Z} \prod_{i=1}^{n} \psi(y_i, f_c(\mathrm{x}_i)) \prod_{j=1}^{m} \psi(\mathrm{f}_j) \psi(\mathrm{f}_j, \mathrm{f}_c). \tag{3}$$

Here the *within-view potential* $\psi(\mathrm{f}_j)$ specifies the dependency structure within each view j, and the *consensus potential* $\psi(\mathrm{f}_j, \mathrm{f}_c)$ describes how each latent function f_j is related to the consensus function f_c. With a GP prior for each of the m views, we can define the following potentials:

$$\psi(\mathrm{f}_j) = \exp\left(-\frac{1}{2} \mathrm{f}_j^\top \mathrm{K}_j^{-1} \mathrm{f}_j \right), \quad \psi(\mathrm{f}_j, \mathrm{f}_c) = \exp\left(-\frac{\|\mathrm{f}_j - \mathrm{f}_c\|^2}{2\sigma_j^2} \right), \tag{4}$$

where K_j is the covariance matrix of view j, that is, $\mathrm{K}_j(\mathrm{x}_k, \mathrm{x}_\ell) = \kappa_j(\mathrm{x}_k^{(j)}, \mathrm{x}_\ell^{(j)})$, and $\sigma_j > 0$ is a scalar which quantifies how apart the latent function f_j is from the consensus function f_c. It is seen that the within-view potentials only rely on the *intrinsic structure* of each view, that is, through the covariance matrix in a GP setting. Finally, the *output potential* $\psi(y_i, f_c(\mathrm{x}_i))$ is defined the same as that in (1) for regression or for classification.

The most important potential function in Bayesian co-training is the consensus potential, which simply defines an isotropic multivariate Gaussian for the difference of f_j and f_c, that is, $\mathrm{f}_j - \mathrm{f}_c \sim \mathcal{N}(0, \sigma_j^2 \mathrm{I})$. This can also be interpreted as assuming a conditional isotropic Gaussian for f_j with the consensus f_c being the mean. Alternatively if f_c is of interest, the joint consensus potentials effectively define a conditional Gaussian prior for $\mathrm{f}_c, \mathrm{f}_c|\mathrm{f}_1, \ldots, \mathrm{f}_m$, as $\mathcal{N}(\mu_c, \sigma_c^2 \mathrm{I})$ where

$$\mu_c = \sigma_c^2 \sum_j \frac{\mathrm{f}_j}{\sigma_j^2}, \qquad \sigma_c^2 = \left(\sum_j \frac{1}{\sigma_j^2} \right)^{-1}. \tag{5}$$

One can easily verify that this is a product of Gaussian distributions, with each Gaussian being $\mathcal{N}(\mathrm{f}_c|\mathrm{f}_j, \sigma_j^2 \mathrm{I})$.[2] This indicates that, given the latent functions $\{\mathrm{f}_j\}_{j=1}^m$, the posterior mean of the consensus function f_c is a *weighted average* of these latent functions, and the weight is given by the inverse variance (i.e., the precision) of each consensus potential. The higher the variance, the smaller the contribution to the consensus function. In the following we call σ_j^2 the *view variance* for view j. In this paper these view variances are taken as parameters of the Bayesian co-training model, but one can also assign a prior (e.g., a Gamma prior) to them and treat them instead as hidden variables. We will discuss the consensus potential and the view variances in more details in Section 3.

In (3) we assume the output y is available for all the n data samples. More generally we consider *semi-supervised* multi-view learning, in which only a subset of data samples have outputs available. This is actually the setting for which co-training and multi-view learning were originally motivated (Blum and Mitchell, 1998). Formally, let n_l be the number of data samples which have outputs available, and let n_u be the number of data samples which do not. We still keep $n = n_l + n_u$ to be the total number of data samples. Under this setting, we only have outputs available for n_l samples, that is, $\mathrm{y}_l = [y_1, \ldots, y_{n_l}]^\top$.

In the functional space, the undirected graphical model for semi-supervised multi-view learning is the same as in Figure 2. The joint probability is also the same as in (2). In the ground network,

2. Note that this conditional Gaussian for f_c has a normalization factor which depends on $\mathrm{f}_1, \ldots, \mathrm{f}_m$.

since the output vector y_l is only of length n_l, the joint probability is now:

$$p(\mathrm{y}_l, \mathrm{f}_c, \mathrm{f}_1, \ldots, \mathrm{f}_m) = \frac{1}{Z} \prod_{i=1}^{n_l} \psi(y_i, f_c(\mathrm{x}_i)) \prod_{j=1}^{m} \psi(\mathrm{f}_j) \psi(\mathrm{f}_j, \mathrm{f}_c). \tag{6}$$

Note that the product of output potentials contains only that of the n_l labeled data samples, and that $\mathrm{f}_c = \{f_c(\mathrm{x}_i)\}_{i=1}^n$ and $\mathrm{f}_j = \{f_j(\mathrm{x}_i^{(j)})\}_{i=1}^n$ are still of length n. Unlabeled data samples contribute to the joint probability via the within-view potentials $\psi(\mathrm{f}_j)$ and consensus potentials $\psi(\mathrm{f}_j, \mathrm{f}_c)$. All the potentials are defined similarly as in (4). In the following we will mainly discuss this more interesting setting.

3. Inference and Learning in Bayesian Co-Training

In this section we discuss inference and learning in the proposed model, assuming first that there is no missing data in any of the views (the setting with missing data will be discussed in Section 4). Instead of working with the undirected graphical model directly, we show different types of marginalizations under this model. The standard inference task is that of inferring y from the observed data, that is, obtaining $p(\mathrm{y})$; however, in order to gain insight into the proposed model and co-training, we explore different marginalizations. All marginalizations lead to standard Gaussian process inference with different latent function at consideration, but interestingly, these different marginalizations show different insights of the proposed undirected graphical model. One advantage of the marginalizations is that it allows us to see that many existing multi-view learning models are actually special cases of the proposed framework. In addition, this Bayesian interpretation helps us understand both the benefits and the limitations of co-training. For clarity we put the derivations into Appendix A.

3.1 Marginal 1: Co-Regularized Multi-View Learning

Our first marginalization focuses on the joint probability distribution of the m latent functions, when the consensus function f_c is integrated out. This would lead to a GP model in which the latent functions are the view specific functions $\mathrm{f}_1, \ldots, \mathrm{f}_m$. Taking the integral of (3) over f_c (and ignoring the output potential for the moment), we obtain the joint marginal distribution as follows after some mathematics (for derivations see Appendix A.1):

$$p(\mathrm{f}_1, \ldots, \mathrm{f}_m) = \frac{1}{Z} \exp\left\{ -\frac{1}{2} \sum_{j=1}^{m} \mathrm{f}_j^\top \mathrm{K}_j^{-1} \mathrm{f}_j - \frac{1}{2} \sum_{j<k} \left[\frac{\|\mathrm{f}_j - \mathrm{f}_k\|^2}{\sigma_j^2 \sigma_k^2} \Big/ \sum_\ell \frac{1}{\sigma_\ell^2} \right] \right\}. \tag{7}$$

It can be seen that the negation of the logarithm of this marginal recovers the regularization terms in the *co-regularized multi-view learning* (see, e.g., Sindhwani et al., 2005; Brefeld et al., 2006). In particular, we have

$$\begin{aligned} -\log p(\mathrm{f}_1, \ldots, \mathrm{f}_m) &= \frac{1}{2} \sum_{j=1}^{m} \mathrm{f}_j^\top \mathrm{K}_j^{-1} \mathrm{f}_j + \frac{1}{2} \sum_{j<k} \left[\frac{\|\mathrm{f}_j - \mathrm{f}_k\|^2}{\sigma_j^2 \sigma_k^2} \Big/ \sum_\ell \frac{1}{\sigma_\ell^2} \right] + \log Z \\ &= \frac{1}{2} \sum_{j=1}^{m} \Omega_j(\mathrm{f}_j) + \frac{1}{2} \frac{1}{\sum_\ell \frac{1}{\sigma_\ell^2}} \sum_{j<k} L(\mathrm{f}_j, \mathrm{f}_k) + \log Z, \end{aligned}$$

where $\Omega_j(\mathrm{f}_j) \triangleq \mathrm{f}_j^\top \mathrm{K}_j^{-1}\mathrm{f}_j$ regularizes the functional space of each individual view j, and the loss function $L(\mathrm{f}_j,\mathrm{f}_k) \triangleq \|\mathrm{f}_j-\mathrm{f}_k\|^2/\sigma_j^2\sigma_k^2$ measures the disagreement of every pair of the function outputs, inversely weighted by the product of the corresponding variances. The higher the variance σ_j^2 of view j, the less the contribution view j brings to the overall loss. We refer to this as *variance-sensitive co-regularized multi-view learning*. Note that unlike the formulation in Brefeld et al. (2006) where the disagreements are only with respect to the unlabeled data, here we regularize the disagreements of all data samples. From the GP perspective, (7) actually defines a *joint multi-view prior* for the m latent functions, $(\mathrm{f}_1,\ldots,\mathrm{f}_m) \sim \mathcal{N}(0,\Lambda^{-1})$, where Λ is a $mn \times mn$ precision matrix with block-wise definition:

$$\Lambda(j,j) = \mathrm{K}_j^{-1} + \frac{1}{\sum_\ell \frac{1}{\sigma_\ell^2}} \sum_{k\neq j} \frac{1}{\sigma_j^2\sigma_k^2}\mathrm{I}, \quad \Lambda(j,j') = -\frac{1}{\sum_\ell \frac{1}{\sigma_\ell^2}} \frac{1}{\sigma_j^2\sigma_{j'}^2}\mathrm{I}, \quad j' \neq j. \tag{8}$$

It is seen that the block-wise precision matrix for view j has contributions from all the other views.

When we take into account the observed output variable y, we can also easily derive the joint marginal of y with all the latent functions $\mathrm{f}_1,\ldots,\mathrm{f}_m$. For instance for regression, the marginal distribution turns out to be (recall that σ^2 is the variance parameter in the output potential for regression):

$$p(\mathrm{y},\mathrm{f}_1,\ldots,\mathrm{f}_m) = \frac{1}{Z}\exp\left\{ -\frac{1}{2\rho\sigma^2}\sum_j \frac{\sum_{i=1}^n (y_i - f_j(\mathrm{x}_i))^2}{\sigma_j^2} \right.$$
$$\left. -\frac{1}{2}\sum_j \mathrm{f}_j^\top \mathrm{K}_j^{-1}\mathrm{f}_j - \frac{1}{2\rho}\sum_{j<k} \frac{\|\mathrm{f}_j-\mathrm{f}_k\|^2}{\sigma_j^2\sigma_k^2} \right\}. \tag{9}$$

Here $\rho \triangleq \frac{1}{\sigma^2} + \sum_j \frac{1}{\sigma_j^2}$ is the sum of all the inverse variances, including the regression variance. Maximizing this marginal distribution is equivalent to solving a minimization problem in co-regularized multi-view learning with least square loss. It is seen that the least square loss with respect to the jth latent function f_j is inversely weighted by the variance σ_j^2, which indicates again that a higher variance leads to less contribution to the total loss.

3.2 Marginal 2: The Co-Training Kernel

The joint multi-view kernel defined in (8) is interesting, but it has a large dimension and is difficult to work with. A more interesting kernel can be obtained if we instead integrate out all the m latent functions $\mathrm{f}_1,\ldots,\mathrm{f}_m$ in (3). This leads to a standard (transductive) Gaussian process model, with f_c being the latent function realizations, and GP prior being $p(\mathrm{f}_c) = \mathcal{N}(0,\mathrm{K}_c)$ where

$$\mathrm{K}_c = \left[\sum_j (\mathrm{K}_j + \sigma_j^2\mathrm{I})^{-1}\right]^{-1}. \tag{10}$$

See Appendix A.2 for the derivation. This indicates that by marginalization, we can transfer the multi-view problem into a single-view problem with respect to the consensus function f_c, without loss of information. The new kernel matrix K_c is derived via all the m kernels from the m views, and note that each entry (i,j) in K_c depends not only on the features of the corresponding data items x_i and x_j, but also on all the other labeled and unlabeled data points (as seen in (10) through matrix inverse). This is the result of the multi-view dependency in the graphical model in Bayesian

co-training, and it also means that this kernel lacks the marginalization property and can only be used in a transductive setting.

This kernel definition is crucial to Bayesian co-training, and in the following we call K_c the *co-training kernel* for multi-view learning. This marginalization reveals the previously unclear insight of how the kernels from different views are combined together in a multi-view learning framework. This allows us to transform a multi-view learning problem into a single-view problem, and simply use the co-training kernel K_c to solve GP classification or regression. Since this marginalization is equivalent to (7),[3] we end up with solutions that are largely similar to any other co-regularization algorithm, but however a key difference is the Bayesian treatment contrasting previous ML-optimization methods.

Formulation (10) can also be viewed as a *kernel design* for transductive multi-view learning, namely, the inverse of the co-training kernel is the sum of the inverse of all individual kernels, corrected by the view specific variance term. Higher variance leads to less contribution to the overall co-training kernel. In a transductive setting where the data are partially labeled, the co-training kernel between labeled data is also dependent on the unlabeled data. Hence the proposed co-training kernel, by the design in (10), can be used for semi-supervised GP learning (Zhu et al., 2003).

Additional benefits of the co-training kernel include the following:

- With fixed hyperparameters (e.g., σ_j^2), the co-training kernel avoids repeated alternating optimizations with respect to the different views f_j, and directly works with a single consensus view f_c. This reduces both time complexity and space complexity (since we only maintain K_c in memory) of multi-view learning.

- While other alternating optimization algorithms might converge to local minima (because they optimize, not integrate), the single consensus view guarantees the *global optimal inference solution* for multi-view learning since it marginalizes other latent functions and leads to a standard GP inference model.

- Even if all the individual kernels are stationary, K_c is in general *non-stationary*. This is because the inverse-covariances are added and then inverted again.

3.3 Marginal 3: Individual View Learning with Side-Information

In Bayesian co-training model we can also focus on one particular view j by marginalizing all the other views and the consensus view. This is particularly interesting if there is one view that is of the main interest (e.g., it provides the most useful features, or it has the least missing features), and we want to understand how the other views influence this view in the inference process. This can be done by integrating out the other latent functions $\mathrm{f}_k, k \neq j$, in (7), and it will lead to another GP formulation with f_j being the latent function. Since (7) represents a jointly Gaussian distribution, we obtain $\mathrm{f}_j \sim \mathcal{N}(0, \mathrm{C}_j)$, where

$$\mathrm{C}_j^{-1} = \mathrm{K}_j^{-1} + \left[\sigma_j^2 \mathrm{I} + \sum_{k \neq j} \left(\mathrm{K}_k + \sigma_k^2 \mathrm{I} \right)^{-1} \right]^{-1}. \tag{11}$$

3. The equivalence is in the sense that both marginalizations are based on the same underlying graphical model, and any optimal solution derived from these marginalizations should be a solution which optimizes the likelihood of the graphical model.

See Appendix A.3 for the derivation. This can be intuitively understood as that the precision matrix of the individual view, C_j^{-1}, is the sum of its original precision matrix and the contributions from other views, weighted by the inverse of the variance. Therefore if σ_k^2 is big for some view k, its contribution to the other views will be compromised. Hence, if one particular view is of interest, we can encode the additional information from the other views into the kernel for the interested view.

Another benefit of this marginalization is the possibility of introducing an inductive inference scheme (rather than transductive as in Section 3.2)—given a new test data x_*, we try to make a prediction of y_* if the jth view $\mathrm{x}_*^{(j)}$ is available. Inspired by Yu et al. (2005), let us define $\alpha_j = [\alpha_{j1}, \ldots, \alpha_{jn}]^\top \in \mathbb{R}^n$ such that $f_j(\mathrm{x}) = \sum_{i=1}^n \alpha_{ji} \kappa_j(\mathrm{x}^{(j)}, \mathrm{x}_i^{(j)})$ (this is also motivated by the Representer theorem). On the training data, this yields $\mathrm{f}_j = \mathrm{K}_j \alpha_j$. From (11) we can see that this re-parameterization leads to a co-training prior for α_j as $\alpha_j \sim \mathcal{N}(0, \mathrm{K}_j^{-1} \mathrm{C}_j \mathrm{K}_j^{-1})$. At testing time when we have the posterior of α_j, y_* can be approximated by $f_j(\mathrm{x}_*) = \sum_{i=1}^n \alpha_{ji} \kappa_j(\mathrm{x}_*^{(j)}, \mathrm{x}_i^{(j)})$. This approach is particularly interesting in the case that one of the views is known to be predictive (i.e., the other views are "side" information to help this primary view), or test data often come with features only in a specific view (since the features from the other views would be disregarded at testing time).

3.4 Optimization of Hyperparameters

One of the advantages of Bayesian co-training is that each view j has a view-specific variance term σ_j^2 to quantify how far the latent function f_j is apart from the consensus view f_c. In particular, a larger value of σ_j^2 implies less confidence on the observation of evidence provided by the jth view. In the perspective of kernel design, this leads to a lesser weight on the kernel K_j. Thus when some views of the data are better at predicting the output than the others, they are weighted more while forming consensus opinions. These variance terms are hyperparameters of the Bayesian co-training model.

To optimize these variance terms together with other hyperparameters involved in each covariance function (e.g., parameter $\rho > 0$ in the Gaussian kernel $\kappa(\mathrm{x}_i, \mathrm{x}_j) = \exp(-\rho \|\mathrm{x}_i - \mathrm{x}_j\|^2)$), we can use the *type II maximum likelihood* method (sometimes called evidence approximation), which maximizes the marginal likelihood with respect to each of these hyperparameters. For simplicity we put the derivation and detailed equations in Appendix B. For more details on the type II maximum likelihood in the GP setting, please refer to Rasmussen and Williams (2006).

3.5 Discussions

The proposed undirected graphical model provides better understanding of multi-view learning algorithms. In each of the marginalizations, we end up with a standard GP model for some latent functions (i.e., $\{\mathrm{f}_1, \ldots, \mathrm{f}_m\}$ in Marginal 1, f_c in Marginal 2, and f_j in Marginal 3). This simplifies learning and inference under the proposed model. Under a transductive setting, the co-training kernel in (10) indicates that *Bayesian co-training is equivalent to single-view learning with a specially designed (non-stationary) kernel*. This is also the preferable way of working with multi-view learning since it avoids alternating optimizations at the inference step.

The proposed graphical model also motivates new methods for unsupervised multi-view learning such as spectral clustering. While the similarity matrix of each view j is encoded in K_j, the

co-training kernel K_c encodes the similarity of two data samples *with multiple views*, and thus can be used directly in spectral clustering.

We would also like to point out the limitations of the proposed consensus-based learning, which are shared by co-training as proposed by Blum and Mitchell (1998) and many other multi-view learning algorithms. As mentioned before, the consensus-based potentials in (4) can be interpreted as defining a Gaussian prior (5) to f_c, where the mean is a *weighted average* of the m individual views. This averaging indicates that the value of f_c is never higher (or lower) than that of any single view. While the consensus-based potentials are intuitive and useful for many applications, they are limited for some real world problems where the evidence from different views should be *additive* (or enhanced) rather than averaging. For instance, when a radiologist is making a diagnostic decision about a lung cancer patient, he or she might look at both the CT image and the MRI image. If either of the two images gives a strong evidence of cancer by that image alone, he or she can make a decision based on a single view (and thus, ignoring the other image completely); if either of the images only gives a moderate evidence (i.e., from a single-view learner which ignores the other image), it would be beneficial to look at both images (i.e., to consider both views), and the final evidence of cancer after observing both images should be higher (or lower, depending on the specific scenario) than either of them if observed individually. It's clear that in this scenario the multiple views are *reinforcing* or *weakening* each other, not averaging. While all the previously proposed co-training and co-regularization algorithms have thus far been based on enforcing consensus between the views explicitly or implicitly, we make this clear from the graphical model perspective, and allow effective tailoring of the view importance from the training data. As part of future work, it would be interesting to explore the possibility of going beyond consensus-based multi-view learning.

4. Bayesian Co-Training with Missing Views

In the previous two sections we assume that the input data are complete, that is, all the views are observed for every data sample. However for many real-world problems, the features could be incomplete or missing for various reasons. For instance, in cancer diagnosis we cannot ask every patient to take all the available imaging tests (e.g., CT, PET, Ultrasound, MRI) for the final diagnosis, so some views (i.e., imaging tests) are missing for certain patients. In this section we extend Bayesian co-training to the case where there are missing (sample, view) pairs in the input data (which can happen both in labeled data and in unlabeled data). The three marginalizations will also be discussed. To the best of our knowledge, this is the first elegant framework to account for the missing views in the multi-view learning setting.

Let each view j be observed for a subset of $n_j \leq n$ samples, and let $\mathbb{I}_j$ denote the indices of these samples in the whole sample set (including labeled and unlabeled data). Note that under this notation, the single-view kernel matrix K_j for view j is of size $n_j \times n_j$, which are defined over the subset of samples denoted by indicator $\mathbb{I}_j$. From the co-training kernel perspective, the difficulty here is to combine the kernels of different sizes together from different views, if at all possible.

We start from the undirected graphical model and make necessary changes to the potentials to account for the missing views. The idea is to treat the missing view information as *hidden* in the graphical model. The undirected graphical model is shown in Figure 3 for Bayesian co-training

with missing views, which is very similar to Figure 1(b). The joint probability can be defined as:

$$p(\mathrm{y}_l, \mathrm{f}_c, \mathrm{f}_1, \ldots, \mathrm{f}_m) = \frac{1}{Z} \prod_{i=1}^{n_l} \psi(y_i, f_c(\mathrm{x}_i)) \prod_{j=1}^{m} \psi(\mathrm{f}_j) \psi(\mathrm{f}_j, \mathrm{f}_c), \tag{12}$$

where $\mathrm{f}_c = \{f_c(\mathrm{x}_i)\}_{i=1}^n \in \mathbb{R}^n$, and $\mathrm{f}_j = \{f_j(\mathrm{x}_i^{(j)})\}_{i \in \mathbb{I}_j} \in \mathbb{R}^{n_j}$. Note that f_j is only realized on a subset of samples and is of length n_j (instead of n). The *within-view potential* $\psi(\mathrm{f}_j)$ is defined via the GP prior, $\psi(\mathrm{f}_j) = \exp(-\frac{1}{2}\mathrm{f}_j^\top \mathrm{K}_j^{-1} \mathrm{f}_j)$, where $\mathrm{K}_j \in \mathbb{R}^{n_j \times n_j}$ is the covariance matrix for view j; the *consensus potential* $\psi(\mathrm{f}_j, \mathrm{f}_c)$ is defined as follows:

$$\psi(\mathrm{f}_j, \mathrm{f}_c) = \exp\left(-\frac{\|\mathrm{f}_j - \mathrm{f}_c(\mathbb{I}_j)\|^2}{2\sigma_j^2} \right), \tag{13}$$

in which $\mathrm{f}_c(\mathbb{I}_j)$ takes the length-n_j subset of vector f_c with indices given in $\mathbb{I}_j$. In other words, the consensus potentials is defined such that

$$\psi(f_j(\mathrm{x}_i), f_c(\mathrm{x}_i)) = \exp\left(-\frac{1}{2\sigma_j^2} \big(f_j(\mathrm{x}_i) - f_c(\mathrm{x}_i)\big)^2 \right), \quad i \in \mathbb{I}_j.$$

The idea here is to define the consensus potential for view j using only the data samples observed in view j. The other data samples with missing view information for view j are treated as hidden (or integrated out) in this potential definition. As before, $\sigma_j > 0$ quantifies how far the latent function f_j is apart from f_c. Note that the smaller n_j is, the less the contribution of view j to the overall graphical model.[4] Next we look at the three marginalizations to gain more insight about this graphical model.

4.1 Co-Regularization with Missing Views

It is straightforward to derive all the marginalizations of Bayesian co-training with missing views. For the co-regularization marginal, a simple calculation leads to the following joint distribution for the m latent functions:

$$p(\mathrm{f}_1, \ldots, \mathrm{f}_m) = \frac{1}{Z} \exp\left\{ -\frac{1}{2} \sum_{j=1}^{m} \mathrm{f}_j^\top \mathrm{K}_j^{-1} \mathrm{f}_j - \frac{1}{2} \sum_{j<k} \sum_{\mathrm{x} \in \mathbb{I}_j \wedge \mathbb{I}_k} \left[\frac{[f_j(\mathrm{x}) - f_k(\mathrm{x})]^2}{\sigma_j^2 \sigma_k^2} \Big/ \sum_{\ell : \mathrm{x} \in \mathbb{I}_\ell} \frac{1}{\sigma_\ell^2} \right] \right\}.$$

As in the Bayesian co-training with fully observed views, this provides an equivalent form to co-regularized multi-view learning. The first part regularizes the functional space of each view, and the second part constrains that every pair of views need to agree on the outputs for *co-observed* samples (inversely weighted by view variances and the sum of inverse variances of the views in which the sample is observed). This is very intuitive and naturally extends the joint distribution in (7). If view j and view k do not share any data sample (i.e., no data sample has features from both view j and view k), the view pair (j,k) will not contribute to the joint distribution.[5] A joint probability distribution involving output y_l can also be derived which takes a similar form as in (9).

4. Also note that after hyperparameter learning, σ_j might not fully represent how strongly each view j contributes to the consensus, since the contribution also depends on the number of available data n_j in the view j.
5. Note that view j and view k will still contribute to the overall distribution through other views that they share data samples with.

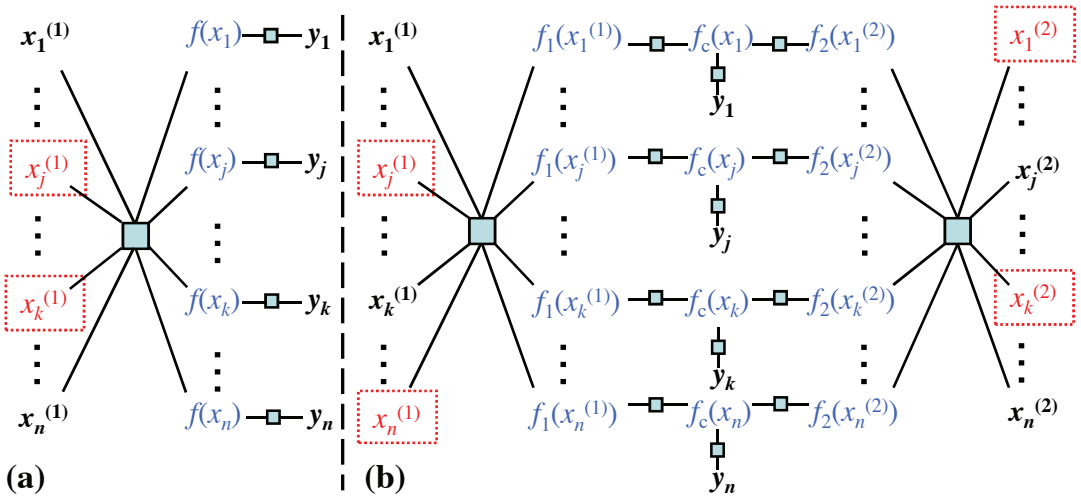


Figure 3: Factor graphs for Bayesian co-training with missing views, for (a) one-view and (b) two-view problems. Observed variables are marked as dark/bold, and unobserved ones are marked as red/non-bold, including functions f_1, f_2, f_c (blue/non-bold). Unobserved variables in a dotted box (such as $\mathrm{x}_j^{(1)}$) are potential observations for active sensing (see Section 5). All labels y are denoted as observed in the graph, but this is not required.

4.2 Co-Training Kernel with Missing Views

We can also derive a co-training kernel K_c by integrating out all the latent functions $\{\mathrm{f}_j\}$ in (12). This leads to a Gaussian prior $p(\mathrm{f}_c) = \mathcal{N}(0, \mathrm{K}_c)$, with

$$\mathrm{K}_c = \Lambda_c^{-1}, \qquad \Lambda_c = \sum_{j=1}^{m} \mathrm{A}_j,$$

where each A_j is a $n \times n$ matrix defined as

$$\mathrm{A}_j(\mathbb{I}_j, \mathbb{I}_j) = (\mathrm{K}_j + \sigma_j^2 \mathrm{I})^{-1}, \text{ and 0 otherwise.} \tag{14}$$

That is, A_j is an expansion of the one-view information matrix $(\mathrm{K}_j + \sigma_j^2 \mathrm{I})^{-1}$ to the full size $n \times n$, with the other (unindexed) entries filled with 0. It is easily seen that such a kernel K_c is indeed positive definite, as long as each one-view kernel K_j is positive definite and at least there are two views sharing one data sample. We also call Λ_c the *co-training precision matrix*. Very importantly, we note that *one additional observation of a (sample, view) pair will affect all the elements of the co-training kernel*. In other words, the kernel value for a pair of samples is potentially changed even when a third (unrelated) object is further characterized by an additional sensor.[6] This property motivates us to do active feature acquisition (or *active sensing*) in the Bayesian co-training framework. Section 5 will discuss this in detail.

6. Note that the marginalizations in Section 4.2 and Section 4.1 are still equivalent (since they come from the same underlying graphical model), despite the fact that additional (sample, view) pair influences the kernels (with dimension $nm \times nm$ in Section 4.1 and $n \times n$ in Section 4.2) differently in these two marginalizations.

4.3 Individual View Learning with Missing Views

If one particular view j is of interest, we can also integrate out the consensus view and all the other views, leading to a GP prior for view j, $\mathrm{f}_j \sim \mathcal{N}(0, \mathrm{C}_j)$, with the precision matrix being

$$\mathrm{C}_j^{-1} = \mathrm{K}_j^{-1} + \left[\sigma_j^2 \mathrm{I} + \Lambda_{c\setminus j}(\mathbb{I}_j, \mathbb{I}_j)^{-1}\right]^{-1}.$$

Here we extract the $(\mathbb{I}_j, \mathbb{I}_j)$ sub-matrix from the *leave-one-view-out* co-training precision matrix $\Lambda_{c\setminus j}$, which is defined as $\Lambda_{c\setminus j} = \sum_{k\neq j} \mathrm{A}_k$. Each A_k is defined as in (14). This marginalization allows us to, for example, measure how much benefit every other view brings to the interested view. An important fact to realize here is that *with an observed (sample, view) pair from another view k, even if this sample is not observed in the primarily interested view j, the kernel of the view j will still be affected so long as* $\mathbb{I}_j \wedge \mathbb{I}_k \neq \emptyset$. One can also introduce the inductive GP inference as in Section 3.3 under this setting.

4.4 Discussion

Bayesian co-training with missing views provides an elegant framework to combine information from multiple views or multiple data sources together, even when different subsets of data samples are measured in different views. For learning and inference, we still prefer using the co-training kernel with the second marginalization due to its simplicity.

We note that the definition of the consensus potentials in (13) implies that the influence of the different pairs of views has been factored into a product. As a consequence, the view-pairs are combined in a linear manner. A way to go beyond this is by using higher-order potentials.

A higher order potential definition $\psi(\mathrm{f}_1, ..., \mathrm{f}_m, \mathrm{f}_c)$, which combines $\mathrm{f}_1, ..., \mathrm{f}_m$ simultaneously, would produce a richer combination of views, but often at the expense of increased inference/computational complexity. It is not clear how to achieve this effect with standard co-training.

Since one observation of a (sample, view) pair will affect the overall co-training kernel, we can derive a framework for *active sensing*, which aims to actively select the best pair for feature acquisition or sensing. This active sensing problem is different from active learning where the goal is to select the best pair for labeling. We discuss this idea in detail in the next section.

5. Active Sensing in Bayesian Co-Training

In active sensing, we are interested in selecting the best unobserved (sample, view) pair for sensing, or for view acquisition, which will improve the overall classification performance. In this section we will focus on logistic regression loss for binary classification. For active sensing we mainly discuss an approach based on the mutual information framework, which measures the expected information gain after observing an additional (sample, view) pair. Another approach based on the predictive uncertainty is also briefly discussed in Section 5.5.

In the following let $\mathcal{D}_O$ and $\mathcal{D}_U$ denote the observed and unobserved (sample, view) pairs, respectively. Recall that under the second marginalization in which only the consensus function f_c is of primary interest, the Bayesian co-training model for binary classification reduces to

$$p(\mathrm{y}_l, \mathrm{f}_c) = \frac{1}{Z}\psi(\mathrm{f}_c)\prod_{i=1}^{n_l}\psi(y_i, f_c(\mathrm{x}_i)),$$

where y_l contains the binary labels for the n_l labeled samples, $\psi(\mathrm{f}_c)$ is defined via the co-training kernel as $\psi(\mathrm{f}_c) = \exp\left\{-\frac{1}{2}\mathrm{f}_c^\top \mathrm{K}_c^{-1}\mathrm{f}_c\right\}$, and $\psi(y_i, f_c(\mathrm{x}_i))$ is the output potential $\lambda(y_i f_c(\mathrm{x}_i))$ with $\lambda(\cdot)$ the logistic function. The log marginal likelihood of the output y_l under this model, conditioned on the input data $\mathrm{X} \triangleq \{\mathrm{x}_i^{(j)}\}$ and model parameters Θ, is:

$$\begin{aligned}\mathcal{L} \triangleq \log p(\mathrm{y}_l|\mathrm{X},\Theta) &= \log \int p(\mathrm{y}_l|\mathrm{f}_c,\Theta)p(\mathrm{f}_c|\mathrm{X},\Theta)\, d\mathrm{f}_c - \log Z \\ &= \log \int \prod_{i=1}^{n_l} \lambda(y_i f_c(\mathrm{x}_i)) \cdot \exp\left\{-\frac{1}{2}\mathrm{f}_c^\top \mathrm{K}_c^{-1}\mathrm{f}_c\right\}\, d\mathrm{f}_c - \log Z.\end{aligned}$$

5.1 Laplace Approximation

To calculate the mutual information we need to calculate the differential entropy of the consensus view function f_c. With co-training kernel and the logistic regression loss, Laplace approximation can be applied to approximate the *a posteriori* distribution of f_c as a Gaussian distribution. The *a posteriori* distribution of f_c, $p(\mathrm{f}_c|\mathcal{D}_O, \mathrm{y}_l, \Theta) \propto p(\mathrm{y}_l|\mathrm{f}_c,\Theta)p(\mathrm{f}_c|\mathcal{D}_O,\Theta)$, is approximately

$$\mathcal{N}(\hat{\mathrm{f}}_c, (\Delta_{\text{post}})^{-1}), \tag{15}$$

where $\hat{\mathrm{f}}_c$ is the maximum *a posteriori* (MAP) estimate of f_c, and the *a posteriori* precision matrix is

$$\Delta_{\text{post}} = \mathrm{K}_c^{-1} + \Phi, \tag{16}$$

with Φ the Hessian of the negative log-likelihood. It turns out that Φ is a diagonal matrix, with $\Phi(i,i) = \eta_i(1-\eta_i)$ where $\eta_i = \lambda(\hat{\mathrm{f}}_c(\mathrm{x}_i))$. The differential entropy of f_c under this Laplace approximation is

$$H(\mathrm{f}_c) = -\frac{n}{2}\log(2\pi e) - \frac{1}{2}\log\det(\Delta_{\text{post}}),$$

where $\det(\cdot)$ denotes the matrix determinant.

5.2 Mutual Information for Active Sensing

Remind that $\mathrm{x}_i^{(j)}$ denote the features in the jth view for the ith sample. In active sensing, the mutual information (MI) between the consensus view function f_c and the unobserved (sample, view) pair $\mathrm{x}_i^{(j)} \in \mathcal{D}_U$ is the *expected decrease in entropy of* f_c when $\mathrm{x}_i^{(j)}$ is observed,

$$I(\mathrm{f}_c, \mathrm{x}_i^{(j)}) = \mathbb{E}[H(\mathrm{f}_c)] - \mathbb{E}[H(\mathrm{f}_c|\mathrm{x}_i^{(j)})] = -\frac{1}{2}\log\det(\Delta_{\text{post}}) + \frac{1}{2}\mathbb{E}\left[\log\det(\Delta_{\text{post}}^{x(i,j)})\right],$$

where the expectation is with respect to $p(\mathrm{x}_i^{(j)}|\mathcal{D}_O, \mathrm{y}_l)$, the distribution of the unobserved (sample, view) pair given all the observed pairs and available outputs. $\Delta_{\text{post}}^{x(i,j)}$ is the *a posteriori* precision matrix, derived from (16), after one pair $\mathrm{x}_i^{(j)}$ is observed.

The maximum MI criterion has been used before to identify the "best" unlabeled sample in active learning (MacKay, 1992). Here we adopt this criterion and choose the unobserved pair which maximizes MI:

$$(i^*, j^*) = \arg\max_{\mathrm{x}_i^{(j)} \in \mathcal{D}_U} I(\mathrm{f}_c, \mathrm{x}_i^{(j)}) = \arg\max_{\mathrm{x}_i^{(j)} \in \mathcal{D}_U} \mathbb{E}\left[\log\det(\Delta_{\text{post}}^{x(i,j)})\right]. \tag{17}$$

5.3 Density Modeling

In order to calculate the expectation in (17), we need a conditional density model for the unobserved pairs, that is, $p(\mathrm{x}_i^{(j)}|\mathcal{D}_O, \mathrm{y}_l)$. This of course depends on the type of the features in each view, and for our applications we use a special Gaussian mixture model (GMM). This model has the nice property that all the marginals are still GMMs, and yet is not too flexible like the full GMM. One can certainly define other density models based on the applications.

For a m-view input data $\mathrm{x} = (\mathrm{x}^{(1)}, \ldots, \mathrm{x}^{(m)})$, let the joint input density be

$$p(\mathrm{x}^{(1)}, \ldots, \mathrm{x}^{(m)}) = p(y=+1)p(\mathrm{x}^{(1)}, \ldots, \mathrm{x}^{(m)}|y=+1) + p(y=-1)p(\mathrm{x}^{(1)}, \ldots, \mathrm{x}^{(m)}|y=-1),$$

and each conditional density takes a *component-wise factorized* GMM form, that is,

$$p(\mathrm{x}^{(1)}, \ldots, \mathrm{x}^{(m)}|y=+1) = \sum_c \pi_c^+ \prod_j \mathcal{N}(\mathrm{x}^{(j)}|\mu_c^{+(j)}, \Sigma_c^{+(j)}),$$
$$p(\mathrm{x}^{(1)}, \ldots, \mathrm{x}^{(m)}|y=-1) = \sum_c \pi_c^- \prod_j \mathcal{N}(\mathrm{x}^{(j)}|\mu_c^{-(j)}, \Sigma_c^{-(j)}).$$

Here, for the positive class, $\mu_c^{+(j)}$ and $\Sigma_c^{+(j)}$ are the mean and covariance matrix for view j in component c, and $\pi_c^+ > 0$, $\sum_c \pi_c^+ = 1$ are the mixture weights. For the negative class we use similar notations. Note that although the conditional density for each mixture component is decoupled for different views, the joint conditional density is not.[7] Under this model, the joint density $p(\mathrm{x}^{(1)}, \ldots, \mathrm{x}^{(m)})$ is also a GMM, and any marginal (conditioned on y or not) density is still a GMM, for example, $p(\mathrm{x}^{(j)}|y=+1) = \sum_c \pi_c^+ \mathcal{N}(\mathrm{x}^{(j)}|\mu_c^{+(j)}, \Sigma_c^{+(j)})$.

Now it is easy to calculate $p(\mathrm{x}_i^{(j)}|\mathcal{D}_O, \mathrm{y}_l)$. Let $\mathrm{x}_i^{(O)}$ be the set of observed views for x_i, we need to distinguish two different settings. When the label y_i is available, for example, $y_i = +1$, we have

$$p(\mathrm{x}_i^{(j)}|\mathcal{D}_O, \mathrm{y}_l) = p(\mathrm{x}_i^{(j)}|\mathrm{x}_i^{(O)}, y_i = +1) = \sum_c \pi_c^{+(j)}(\mathrm{x}_i^{(O)}) \cdot \mathcal{N}(\mathrm{x}_i^{(j)}|\mu_c^{+(j)}, \Sigma_c^{+(j)}), \tag{18}$$

which is again a GMM model, with the mixing weights being

$$\pi_c^{+(j)}(\mathrm{x}_i^{(O)}) = \pi_c^+ \frac{\prod_{k \in O} \mathcal{N}(\mathrm{x}_i^{(k)}|\mu_c^{+(k)}, \Sigma_c^{+(k)})}{p(\mathrm{x}_i^{(O)}|y_i = +1)}.$$

When the label y_i is not available, we need to integrate out the labeling uncertainty and compute

$$\begin{aligned} p(\mathrm{x}_i^{(j)}|\mathcal{D}_O, \mathrm{y}_l) &= p(\mathrm{x}_i^{(j)}|\mathrm{x}_i^{(O)}) \\ &= p(y_i = +1)p(\mathrm{x}_i^{(j)}|\mathrm{x}_i^{(O)}, y_i = +1) + p(y_i = -1)p(\mathrm{x}_i^{(j)}|\mathrm{x}_i^{(O)}, y_i = -1), \end{aligned}$$

which is a GMM model as well, as can be seen from (18).

7. A straightforward EM algorithm can be derived to estimate all these parameters. When labels are only available for a very limited number of samples, one might assume a full generative GMM model neglecting the dependency on labels (instead of a conditional GMM model).

5.4 Expectation Calculation

We are now ready to compute the expectation in (17). The *a posteriori* precision matrix after one (sample, view) pair $\mathrm{x}_i^{(j)}$ is observed, $\Delta_{\text{post}}^{x(i,j)}$, can be calculated as

$$\Delta_{\text{post}}^{x(i,j)} = (\mathrm{K}_c^{x(i,j)})^{-1} + \Phi = \mathrm{A}_j^{x(i,j)} + \sum_{k \neq j} \mathrm{A}_k + \Phi, \tag{19}$$

where $\mathrm{K}_c^{x(i,j)}$ and $\mathrm{A}_j^{x(i,j)}$ are the new K_c and A_j matrices after the new pair is observed. Based on (14), to calculate $\mathrm{A}_j^{x(i,j)}$ we need to recalculate the kernel for the jth view, K_j, after an additional pair $\mathrm{x}_i^{(j)}$ is observed. This is simply done by adding one row and column to the old K_j as:

$$\mathrm{K}_j^{x(i,j)} = \begin{bmatrix} \mathrm{K}_j & \mathrm{b}_j \\ \mathrm{b}_j^\top & a_j \end{bmatrix},$$

where $a_j = \kappa_j(\mathrm{x}_i^{(j)}, \mathrm{x}_i^{(j)}) \in \mathbb{R}$, and $\mathrm{b}_j \in \mathbb{R}^{n_j}$ has the ℓth entry as $\kappa_j(\mathrm{x}_\ell^{(j)}, \mathrm{x}_i^{(j)})$. Then from (14), the non-zero part of $\mathrm{A}_j^{x(i,j)}$ is calculated as

$$\left(\mathrm{K}_j^{x(i,j)} + \sigma_j^2 \mathrm{I}\right)^{-1} = \begin{bmatrix} \mathrm{K}_j + \sigma_j^2 \mathrm{I} & \mathrm{b}_j \\ \mathrm{b}_j^\top & a_j + \sigma_j^2 \end{bmatrix}^{-1} = \begin{bmatrix} \Gamma_j + \lambda_j \Gamma_j \mathrm{b}_j \mathrm{b}_j^\top \Gamma_j & -\lambda_j \Gamma_j \mathrm{b}_j \\ -\lambda_j \mathrm{b}_j^\top \Gamma_j & \lambda_j \end{bmatrix}, \tag{20}$$

using the block-matrix inverse formula, where $\Gamma_j = (\mathrm{K}_j + \sigma_j^2 \mathrm{I})^{-1}$ and $\lambda_j = \frac{1}{a_j + \sigma_j^2 - \mathrm{b}_j^\top \Gamma_j \mathrm{b}_j}$.

As seen from (19) and (20), it is difficult to directly calculate the expectation in (17). Since for any matrix Q, $\mathbb{E}[\log\det(\mathrm{Q})] \leq \log\det(\mathbb{E}[\mathrm{Q}])$ due to the concavity of $\log\det(\cdot)$, we alternatively take the upper bound $\log\det(\mathbb{E}[\Delta_{\text{post}}^{x(i,j)}])$ as the selection criteria and also take the risk that the best pair (i, j) that optimizes $\log\det(\mathbb{E}[\Delta_{\text{post}}^{x(i,j)}])$ doesn't necessarily optimize $\mathbb{E}[\log\det(\Delta_{\text{post}}^{x(i,j)})]$. From (19) and (20), this reduces to computing $\mathbb{E}[\lambda_j], \mathbb{E}[\lambda_j \mathrm{b}_j]$ and $\mathbb{E}[\lambda_j \mathrm{b}_j \mathrm{b}_j^\top]$, where the expectations are with respect to $p(\mathrm{x}_i^{(j)} | \mathcal{D}_O, \mathrm{y})$, a GMM model (cf. Section 5.3). In general one needs to calculate these expectations numerically, as different kernel functions lead to different integrals. As another approximation one might assume each of the GMM component is a point-mass such that the mean is used for the calculation.

5.5 Discussion

The mutual information based approach directly measures the expected information gain for every (sample, view) pair. A different (and simpler) approach is based on the predictive uncertainty, in which the most *uncertain* sample (after the current classifier is trained) is selected for view acquisition. This approach was taken for a different problem in Melville et al. (2004). This uncertainty (i.e., predictive variance) is estimated as the diagonal entries of the *a posteriori* covariance matrix $(\Delta_{\text{post}})^{-1}$, as seen from (15). However it is not clear what view to acquire for this sample (if more than one view is missing for the sample). The advantage of this approach is that no density modeling is necessary for unobserved views.

6. Experiments

For the first part of the experiments we empirically evaluate some single-view and multi-view learning algorithms on several toy data and two real world data sets. We compare the proposed Bayesian

co-training models with the original co-training method proposed by Blum and Mitchell (1998), and several single-view learning algorithms. Since this co-training algorithm—sometimes we call it the *canonical co-training* algorithm—was proposed for classification problems, we focus on classification in this section and compare all the methods with the logistic regression loss. We show both problems where co-training works and does not work (i.e., is not better compared to the single-view learning counterpart).

In the second part we evaluate the active sensing algorithms in the Bayesian co-training setting. We are given a classification task with missing views, and at each iteration we are allowed to select an unobserved (sample, view) pair for sensing (i.e., feature acquisition). The proposed methods are compared with random sensing in which a random unobserved (sample, view) pair is selected for sensing.

6.1 Toy Examples for Bayesian Co-Training

First of all, we show some 2D toy classification problems to visualize the co-training result in Figure 4. We assume each of these 2D problems is a two-view problem, in which one view only contains one single feature. Canonical co-training is applied by iteratively training one classifier based on one view, adding the most confident unlabeled data from one view to the training pool of the other classifier, and retraining each classifier till convergence (i.e., no confident unlabeled data can be added further). In Bayesian co-training we use the squared exponential covariance function as mentioned in Section 2, and the width ρ is set to $1/\sqrt{2}$ which yields the optimal performance.

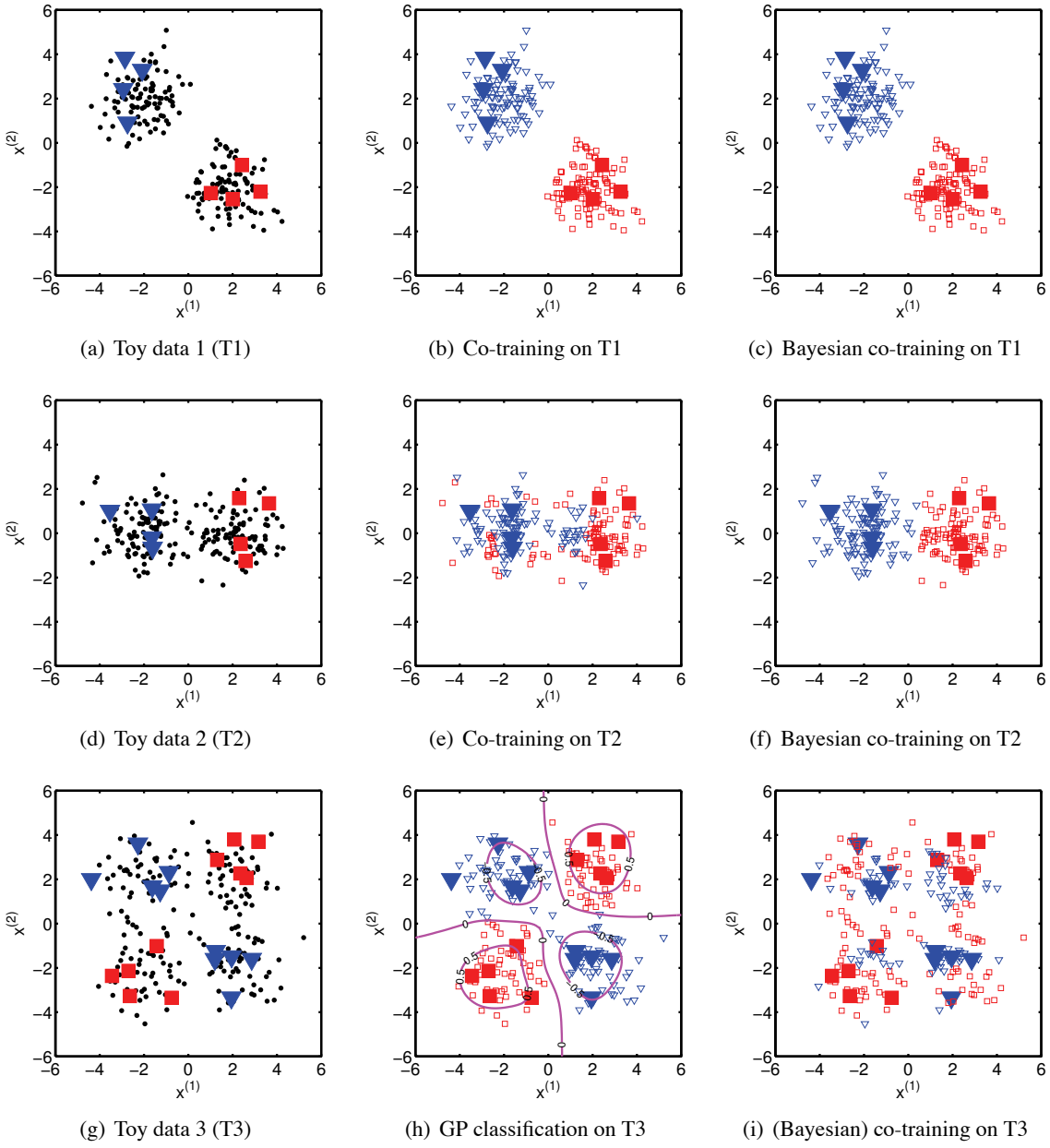
Our first example is a two-Gaussian case with mean $(2,-2)$ and $(-2,2)$, where either feature $x^{(1)}$ or $x^{(2)}$ can be used alone to fully solve the problem (Figure 4(a)). This is an ideal case for co-training, since: 1) each single view is sufficient to train a classifier, and 2) both views are conditionally independent given the class labels. Therefore we see that both canonical co-training and Bayesian co-training yield the same perfect result (Figure 4(b),(c)).

For the second toy data (Figure 4(d)) we assume the two Gaussians are aligned to the $x^{(1)}$-axis (with mean $(2,0)$ and $(-2,0)$). In this case the feature $x^{(2)}$ is totally irrelevant to the classification problem. The canonical co-training fails here (Figure 4(e)) since when we add labels using the $x^{(2)}$ feature , noisy labels will be introduced and expanded to future training. The Bayesian co-training model can handle this situation since we can adapt the weight of each view and penalize the feature $x^{(2)}$ (Figure 4(f)).

The third toy data follows an XOR shape where the data from four Gaussians (with mean $(2,2)$, $(-2,2)$, $(2,-2)$, $(-2,-2)$) lead to a binary classification problem that is not linearly separable (Figure 4(g)). In this case both the two assumptions mentioned above are violated, and neither canonical nor Bayesian co-training will work (Figure 4(i)).[8] On the other hand, a supervised GP classification model with squared exponential covariance function can easily recover the non-linear underlying structure (see Figure 4(h)). This indicates that the learning a multi-view classifier for this problem with the current co-training type algorithms will not succeed. From a kernel design perspective, the consensus based co-training kernel K_c is not suitable for this type of problem.

In summary, these toy problems indicate that when co-training works, Bayesian co-training performs better than or at least as well as canonical co-training models. But since Bayesian co-training is fundamentally a kernel design for a single-view supervised learning, it will not work when the problem calls for more flexible kernel form (e.g., in Figure 4(g)).

8. We also tried other types of covariance functions but they yield similar results.



(a) Toy data 1 (T1) (b) Co-training on T1 (c) Bayesian co-training on T1

(d) Toy data 2 (T2) (e) Co-training on T2 (f) Bayesian co-training on T2

(g) Toy data 3 (T3) (h) GP classification on T3 (i) (Bayesian) co-training on T3

Figure 4: Toy problems for co-training. (b)∼(c) show canonical and Bayesian co-training results on two-Gaussian data (a); (e)∼(f) show the results on two-Gaussian data (d); (h) shows GP classification result on four-Gaussian XOR data (g); (i) shows (Bayesian) co-training result on data (g). Square exponential covariance function was used with width 1 for GP classification and $1/\sqrt{2}$ for each feature in two-view learning. In the toy data big red-square/blue-triangle markers denote the $+1/-1$ labeled points, and black dots denote the unlabeled points.

	# TRAIN +2/-10		# TRAIN +4/-20	
MODEL	AUC	F1	AUC	F1
TEXT	0.5725 ± 0.0180	0.1359 ± 0.0565	0.5770 ± 0.0209	0.1443 ± 0.0705
INBOUND LINK	0.5451 ± 0.0025	0.3510 ± 0.0011	0.5479 ± 0.0035	0.3521 ± 0.0017
OUTBOUND LINK	0.5550 ± 0.0119	0.3552 ± 0.0053	0.5662 ± 0.0124	0.3600 ± 0.0059
TEXT+LINK	0.5730 ± 0.0177	0.1386 ± 0.0561	0.5782 ± 0.0218	0.1474 ± 0.0721
CO-TRAINED GPLR	0.6459 ± 0.1034	0.4001 ± 0.2186	0.6519 ± 0.1091	0.4042 ± 0.2321
BAYESIAN CO-TRAINING	$\mathbf{0.6536 \pm 0.0419}$	$\mathbf{0.4210 \pm 0.0401}$	$\mathbf{0.6880 \pm 0.0300}$	$\mathbf{0.4530 \pm 0.0293}$

Table 1: Results for Citeseer with different numbers of labeled training data (positive/negative). The first three lines are supervised learning results using only the single-view features. The fourth line shows the supervised learning results by combining features from all the three views. The fifth and sixth lines are the co-training results. Bold face indicates the best performance.

MODEL	# TRAIN +2/-2		# TRAIN +4/-4	
	AUC	F1	AUC	F1
TEXT	0.5767 ± 0.0430	0.4449 ± 0.1614	$\mathbf{0.6150 \pm 0.0594}$	0.5338 ± 0.1267
INBOUND LINK	0.5211 ± 0.0017	$\mathbf{0.5761 \pm 0.0013}$	0.5210 ± 0.0019	$\mathbf{0.5758 \pm 0.0015}$
TEXT+LINK	0.5766 ± 0.0429	0.4443 ± 0.1610	$\mathbf{0.6150 \pm 0.0594}$	0.5336 ± 0.1267
CO-TRAINED GPLR	0.5624 ± 0.1058	0.5437 ± 0.1225	0.5959 ± 0.0927	0.5737 ± 0.1203
BAYESIAN CO-TRAINING	$\mathbf{0.5794 \pm 0.0491}$	0.5562 ± 0.1598	0.6140 ± 0.0675	0.5742 ± 0.1298

Table 2: Results for WebKB with different numbers of labeled training data (positive/negative). The first two lines are supervised learning results using only the single-view features. The third line shows the supervised learning results by combining features from both views. The fourth and fifth lines are the co-training results. Bold face indicates the best performance.

6.2 Bayesian Co-Training for Web Page Classification

We use two sets of linked documents for our experiment. The main purpose of these empirical studies is to show the benefit of the proposed Bayesian co-training method compared to single-view learning and the canonical co-training algorithms, and also highlight the limitations of co-training type algorithms. As will be seen later, we show one case that co-training works, in which case Bayesian co-training yields the best performance; we also show one case that co-training does not improve over the single-view counterpart, in which case Bayesian co-training is slightly better than canonical co-training. As the co-training kernel based approach is equivalent to the adaptive co-regularized multi-view learning (since they are based on the same underlying graphical model), we do not include a separate line of results for the co-regularization methods.

The *Citeseer* data set contains 3,312 documents that belong to six classes. There are three natural views for each document: the text view consists of title and abstract of the paper; the two link views are inbound and outbound references. The bag-of-words features are extracted from each view, which amount to 3,703 for the text view, 1,107 for the inbound view and 903 for the outbound view. We pick up the largest class which contains 701 documents and test the one-vs-rest classification performance. The *WebKB* data set is a collection of 4,501 academic web pages

manually grouped into six classes (student, faculty, staff, department, course, project). There are two views containing the text on the page (24,480 features) and the anchor text (901 features) of all inbound links, respectively. We consider the binary classification problem "student" against "faculty", for which there are 1,641 and 1,119 documents, respectively. The preprocessed data sets are kindly shared by Steffen Bickel at `http://www.mpi-inf.mpg.de/~bickel/mvdata/`.

We compare the single-view learning methods based on logistic regression with Gaussian processes (using features in the single view such as TEXT, INBOUND LINK, and OUTBOUND LINK), concatenated-view method based on logistic regression with Gaussian processes (TEXT+LINK), and co-training methods CO-TRAINED GPLR (which stands for Co-Trained Gaussian Process Logistic Regression using canonical co-training) and BAYESIAN CO-TRAINING (using co-training kernel with logistic regression loss function). Linear kernels are used for all the competing methods since it is very robust from our experience in these experiments. For CO-TRAINED GPLR method, we repeat the procedure 50 times, and in each iteration we add the most predictable 1 positive sample and r negative samples into the training set where r depends on the number of negative/positive ratio of each training data set. The classifier we use is the Gaussian process classifier with logistic regression loss (or GPLR for short). For BAYESIAN CO-TRAINING, we use the co-training kernel approach with the same GPLR classifier. Performance is evaluated using AUC score and F1 measure. We vary the number of labeled training documents as seen in Table 1 and 2 (with ratio proportional to the true positive/negative ratio). Single-view learning methods use only the labeled data, and co-training algorithms are allowed to use all the unlabeled data in the training process. The experiments are repeated 20 times and the prediction means and standard deviations are shown in Table 1 and 2.

It can be seen that for the binary classification problem in Citeseer data set, the co-training methods are better than the single-view methods. In this case BAYESIAN CO-TRAINING is better than CO-TRAINED GPLR and achieves the best performance. For WebDB, however, CO-TRAINED GPLR is not as good as the single-view counterparts, and thus BAYESIAN CO-TRAINING is also worse than the purely supervised methods though it is slightly better than CO-TRAINED GPLR. This is maybe because the TEXT and LINK features are not independent given the class labels (especially when two classes "faculty" and "staff" might share features). CO-TRAINED GPLR has higher standard deviations than other methods due to the possibility of adding noisy labels. We have also tried other number of iterations but 50 seems to give an overall best performance.

Note that the single-view learning with TEXT almost achieves the same performance as concatenated-view method. This might be because the number of text features are much more than the link features (e.g., for WebKB there are 24,480 text features and only 901 link features). So these multiple views are very unbalanced and should be taken into account in co-training with different weights. Bayesian co-training provides a natural way of doing it.

6.3 Active Sensing on Toy Data

We show some empirical results on active sensing in this and the following subsections. Suppose we are given a classification task with missing views, and at each iteration we are allowed to select an unobserved (sample, view) pair for sensing (i.e., feature acquisition). We compare the classification performance on unlabeled data using the following three sensing approaches:

- **Active Sensing MI**: The pair is selected based on the mutual information criteria (17).

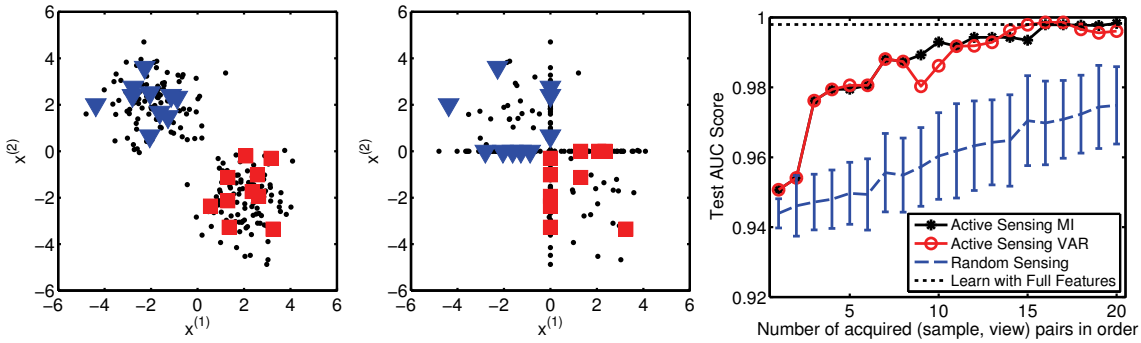


Figure 5: Toy data for active sensing (left). Big red-square/blue-triangle markers denote $+1/-1$ labeled points, and black dots denote unlabeled points. Data are sampled from two Gaussians with mean $(2,-2)$, $(-2,2)$ and unit variance. After "hiding" one feature for some of the data points, the data look like (middle) with removed features replaced with 0. Comparison of active sensing with random sensing is shown on the right. The x-axis labels each acquired pair in order.

- **Active Sensing VAR**: A sample is selected first which has the maximal predictive variance and has missing views, and then one of the missing views is randomly selected for sensing.

- **Random Sensing**: A random unobserved (sample, view) pair is selected for sensing.

After the pair is acquired in each iteration, learning is done using the Bayesian co-training model (with missing views), as discussed in Section 4. Note that for all the three approaches, the acquired (sample, view) pair will affect all the samples in the next iteration (via the co-training kernel). In active sensing with MI, we use EM algorithm to learn the GMM structure with missing entries, and the GMM model is re-estimated after each pair is selected and filled in (this is fast thanks to the incremental updates in the EM algorithm).

We first illustrate active sensing with a toy example. Figure 5 (left) shows a well separated two-class problem which is similar to the one shown in Figure 4(a). To simulate our active sensing experiment, we randomly "hide" one of the two features of each sample with 40% probability each, and with 20% probability observe both features. The final incomplete training data are shown in Figure 5 (middle) with the incomplete samples shown along the first or second axis. It can be seen that only 2 fully observed positive and negative samples are available. For active sensing MI we use the Gaussian kernel with width 0.5, and let the GMM choose the number of clusters automatically (see, e.g., Corduneanu and Bishop, 2001). Standard transductive setting is applied where all the unlabeled data are available for co-training kernel calculation. In Figure 5 (right) we compare active sensing with random sensing, using AUC for the unlabeled data. This indicates that active sensing is much better than random sensing in improving the classification performance. The Bayes optimal accuracy (reachable when there is no missing data) is reached by the 16th query by active sensing whereas random sensing improves much slower with the number of acquired pairs. The two active sensing algorithms show similar results.

Features for NSCLC 2-years Survival Prediction

Feature	Description	View
GENDER	1-Male, 2-Female	1st
WHO	WHO performance status	1st
FEV1	Forced expiratory volume in 1 second	1st
GTV	Gross tumor volume	2nd
NPLN	Number of positive lymph node stations	2nd

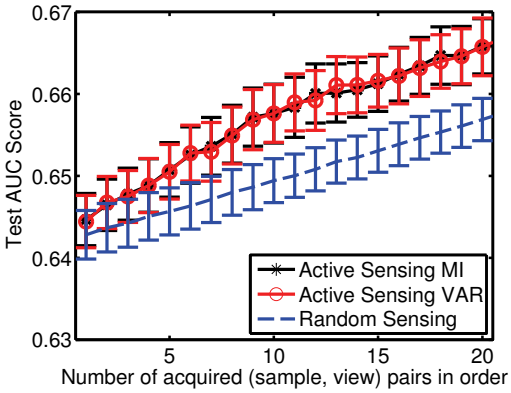


Figure 6: Experiments on NSCLC survival prediction. The features for the 2 views are listed in the left table, and the performance comparison of active sensing and random sensing is shown in the right figure. As baselines, training with full features (i.e., no sensing needed) yields 0.73; training with mean imputation (i.e., using the mean of each feature to fill in the missing entries) yields 0.62.

6.4 Active Sensing in Survival Prediction for Lung Cancer

We consider 2-year survival prediction for advanced non-small cell lung cancer (NSCLC) patients treated with (chemo-)radiotherapy. This is currently a very challenging problem in clinical research, since the prognosis of this group of patients is very poor (less than 40% survive two years). Currently most models in the literature rely on various clinical factors of the patient such as gender and the WHO performance status. Very recently, imaging-related factors such as the size of the tumor and the number of positive lymph node stations are shown to be better predictors (Dehing-Oberije et al., 2009). However, it is expensive to obtain the images and to manually measure these factors. Therefore we study how to select the best set of patients to go through imaging to get additional features. All the relevant factors are listed in Figure 6 (left) with short descriptions. These factors are all known to be predictive based on Dehing-Oberije et al. (2009). From Bayesian co-training point of view we have 2 views, with 3 features in the first (clinical feature) view and 2 features in the second (imaging-based feature) view.

Our study contains 233 advanced NSCLC patients treated at the MAASTRO Clinic in the Netherlands from 2002 to 2006, among which 77 survived 2 years (labeled +1). All the features are available for these patients, and are normalized to have zero mean and unit variance before training. We randomly choose 30% of the patients as training samples (with labels known), and the rest 70% as unlabeled samples. We use linear kernel for each view, and let the GMM algorithm automatically choose the number of clusters. As the active sensing setup, the first view is available for all the patients, and the second view is available only for randomly chosen 50% patients. So our goal is to sequentially select patients to acquire features in view 2, such that the overall classifier performance is maximized. Figure 6 (right) shows the test AUC scores (with error-bars) of active sensing and random sensing, with different number of acquired pairs. Performance is averaged over 20 runs with randomly chosen 50% patients at the start. Active sensing in general yields better performance, and is significantly better after 5 first pairs. Active sensing based on MI and VAR again yield very

similar results. We have also tested other experimental settings, and the comparison is not sensitive to this setup.

6.5 Active Sensing in pCR Prediction for Rectal Cancer

Our second example is to predict tumor response after chemo-radiotherapy for locally advanced rectal cancer. This is important in individualizing treatment strategies, since patients with a pathologic complete response (pCR) after therapy, that is, with no evidence of viable tumor on pathologic analysis, would need less invasive surgery or another radiotherapy strategy instead of resection. Most available models combine clinical factors such as gender and age, and pre-treatment imaging-based factors such as tumor length and SUV_{max} (from CT/PET imaging), but it is expected that adding imaging data collected *after* therapy would lead to a better predictive model (though with a higher cost). In this study we show how to effectively select patients to go through pre-treatment and post-treatment imaging to better predict pCR.

We use the data from Capirci et al. (2007) which contains 78 prospectively collected rectal cancer patients. All patients underwent a CT/PET scan before treatment and 42 days after treatment, and 21 of them had pCR (labeled +1). We split all the features into 3 views (clinical, pre-treatment imaging, post-treatment imaging), and the features are listed in Figure 7 (left). For active sensing, we assume that all the (labeled or unlabeled) patients have view 1 features available, 70% of the patients have view 2 features available, and 40% of the patients have view 3 features available. This is to account for the fact that view 3 features are most expensive to get. All the other settings are the same as the NSCLC survival prediction study. Figure 7 (right) shows the performance comparison of active sensing with random sensing, and it is seen that after about 18 pair acquisitions, active sensing is significantly better than random sensing. Active sensing MI and VAR share a similar trend, and the MI based active sensing is overall better than VAR based active sensing. The difference is however not statistically significant. The optimal AUC (when there are no missing features) is shown as a dotted line, and we see that with around 34 actively acquired pairs, active sensing can almost achieve the optimum. It takes however much longer for random sensing to reach this performance.

7. Conclusion

This paper has two principal contributions. We have proposed a graphical model for combining multi-view data, and shown that previously derived co-regularization based training algorithms maximize the likelihood of this model. In the process, we showed that these algorithms have been making an intrinsic assumption of the form $p(f_c, f_1, f_2, \ldots, f_m) \propto \psi(f_c, f_1)\psi(f_c, f_2)\ldots\psi(f_c, f_m)$, even though it was not explicitly realized earlier. We also studied circumstances when this assumption proves unreasonable. Thus, our first contribution was to clarify the implicit assumptions and limitations in multi-view consensus learning in general, and co-regularization in particular.

Motivated by the insights from the graphical model, our second contribution was the development of alternative algorithms for co-regularization; in particular the development of a non-stationary co-training kernel. Unlike previously published co-regularization algorithms, our approach handles all the following in an elegant framework: (a) handles naturally more than 2 views; (b) automatically learns which views of the data should be trusted more while predicting class labels; (c) shows how to leverage previously developed methods for efficiently training GP/SVM; (d) clearly explains our assumptions, for example, what is being optimized *overall*; (e) does not suffer

Features for pCR Prediction in Rectal Cancer

Feature	Description	View
GENDER	1-Male, 2-Female	1st
AGE	Age in years	1st
STAGE	Staging of cancer	1st
LENGTH	Max diameter of the tumor	2nd
SUVPre	SUV_{max} before treatment	2nd
ΔSUV	Absolute difference of SUV_{max} before and after treatment	3rd
RI	Response Index, ΔSUV in %	3rd

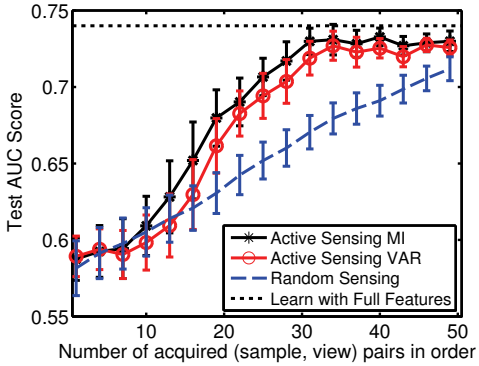


Figure 7: Experiments on pCR prediction for rectal cancer. The features for the 3 views are listed in the left table, and the performance comparison of active sensing and random sensing is shown in the right figure. As baselines, training with full features (i.e., no sensing needed) yields 0.74 (shown as a dotted line); training with mean imputation (i.e., using the mean of each feature to fill in the missing entries) yields 0.55 (not shown).

from local maxima problems; (f) is less computationally demanding in terms of both speed and memory requirements.

We also extend this framework to handle multi-view data with missing features, and introduce an active sensing framework which allows us to actively acquiring missing (sample, view) pairs to maximize performance. In the future we plan to study alternative potentials based on the proposed graphical model, and explore inductive multi-view learning in a more principled manner.

Appendix A. Derivations of the Marginalizations

In this appendix we provide the derivations of the various marginalizations of the Bayesian co-training model, described in Section 3. The joint probability of all the variables is defined as in (6) and is repeated here:

$$p(\mathrm{y}_l, \mathrm{f}_c, \mathrm{f}_1, \ldots, \mathrm{f}_m) = \frac{1}{Z} \prod_{i=1}^{n_l} \psi(y_i, f_c(\mathrm{x}_i)) \prod_{j=1}^{m} \psi(\mathrm{f}_j) \psi(\mathrm{f}_j, \mathrm{f}_c). \tag{21}$$

Recall that the following integration result is true for any $\mathrm{x} \in \mathbb{R}^p$, $\mathrm{b} \in \mathbb{R}^p$, and symmetric matrix $\mathrm{A} \in \mathbb{R}^{p \times p}$.

$$\int \exp\left\{-\frac{1}{2}\mathrm{x}^\top \mathrm{A}\mathrm{x} + \mathrm{b}^\top \mathrm{x}\right\} d\mathrm{x} = \sqrt{\det(2\pi \mathrm{A}^{-1})} \exp\left\{\frac{1}{2}\mathrm{b}^\top \mathrm{A}^{-1}\mathrm{b}\right\}. \tag{22}$$

A.1 Marginal 1: Co-Regularized Multi-View Learning

The first marginalization integrates out the latent consensus function f_c in (21). Ignoring the output consensus function $\psi(y_i, f_c(\mathrm{x}_i))$ for the moment, we derive the joint likelihood

$$\begin{aligned} p(\mathrm{f}_1,\ldots,\mathrm{f}_m) &= \frac{1}{Z}\int \prod_{j=1}^{m} \psi(\mathrm{f}_j)\psi(\mathrm{f}_j,\mathrm{f}_c)\, d\mathrm{f}_c \\ &= \frac{1}{Z}\int \prod_{j=1}^{m} \exp\left\{-\frac{1}{2}\mathrm{f}_j^\top \mathrm{K}_j^{-1}\mathrm{f}_j^\top - \frac{\|\mathrm{f}_j-\mathrm{f}_c\|^2}{2\sigma_j^2}\right\} d\mathrm{f}_c \\ &= \frac{1}{Z}\int \exp\left\{-\frac{1}{2}\sum_{j=1}^{m}\left[\mathrm{f}_j^\top \mathrm{K}_j^{-1}\mathrm{f}_j^\top + \frac{\|\mathrm{f}_j-\mathrm{f}_c\|^2}{\sigma_j^2}\right]\right\} d\mathrm{f}_c \\ &= \frac{1}{Z}\int \exp\left\{-\frac{1}{2}\mathrm{f}_c^\top \mathrm{A}\mathrm{f}_c + \mathrm{b}^\top \mathrm{f}_c + C\right\} d\mathrm{f}_c, \end{aligned}$$

in which we define

$$\mathrm{A} = \sum_j \frac{1}{\sigma_j^2}\mathrm{I}, \quad \mathrm{b} = \sum_j \frac{\mathrm{f}_j}{\sigma_j^2}, \quad C = -\frac{1}{2}\sum_j \left[\mathrm{f}_j^\top \mathrm{K}_j^{-1}\mathrm{f}_j + \frac{\|\mathrm{f}_j\|^2}{\sigma_j^2}\right]. \tag{23}$$

Note that C does not depend on f_c. Applying (22) and absorbing the constants into the normalization factor Z, we have

$$\begin{aligned} p(\mathrm{f}_1,\ldots,\mathrm{f}_m) &= \frac{1}{Z}\exp\left\{-\frac{1}{2}\sum_j \mathrm{f}_j^\top \mathrm{K}_j^{-1}\mathrm{f}_j - \frac{1}{2}\sum_j \frac{\|\mathrm{f}_j\|^2}{\sigma_j^2} + \frac{1}{2}\frac{1}{\sum_j \frac{1}{\sigma_j^2}}\left\|\sum_j \frac{\mathrm{f}_j}{\sigma_j^2}\right\|^2\right\} \\ &= \frac{1}{Z}\exp\left\{-\frac{1}{2}\sum_j \mathrm{f}_j^\top \mathrm{K}_j^{-1}\mathrm{f}_j - \frac{1}{2}\frac{1}{\sum_j \frac{1}{\sigma_j^2}}\left[\sum_j \frac{1}{\sigma_j^2}\cdot\sum_j \frac{\|\mathrm{f}_j\|^2}{\sigma_j^2} - \left\|\sum_j \frac{\mathrm{f}_j}{\sigma_j^2}\right\|^2\right]\right\} \\ &= \frac{1}{Z}\exp\left\{-\frac{1}{2}\sum_j \mathrm{f}_j^\top \mathrm{K}_j^{-1}\mathrm{f}_j - \frac{1}{2}\frac{1}{\sum_j \frac{1}{\sigma_j^2}}\sum_{j<k}\frac{\|\mathrm{f}_j-\mathrm{f}_k\|^2}{\sigma_j^2\sigma_k^2}\right\}. \end{aligned}$$

This recovers the marginal 1 as in (7). To see the GP view of this marginal as in (8), we just need to notice that (7) is a quadratic form of the joint latent functions $(\mathrm{f}_1,\ldots,\mathrm{f}_m)$, and relocate the terms in (7) in the GP format.

When the output potentials $\psi(y_i, f_c(\mathrm{x}_i))$ are taken into account, the whole derivation follows with the only difference that there is an additional term with respect to y in each summation in (23). So we obtain (9) as the joint marginal likelihood.

A.2 Marginal 2: The Co-Training Kernel

To get the co-training kernel we integrate out all the m latent functions in (21), leaving only f_c and y_l. We calculate the marginal distribution of y_l and f_c as follows:

$$\begin{aligned} p(\mathrm{y}_l,\mathrm{f}_c) &= \int p(\mathrm{y}_l,\mathrm{f}_c,\mathrm{f}_1,\ldots,\mathrm{f}_m)\, d\mathrm{f}_1\ldots d\mathrm{f}_m \\ &= \frac{1}{Z}\prod_{i=1}^{n_l}\psi(y_i, f_c(\mathrm{x}_i))\prod_{j=1}^{m}\int \psi(\mathrm{f}_j)\psi(\mathrm{f}_j,\mathrm{f}_c)\, d\mathrm{f}_j, \end{aligned} \tag{24}$$

and

$$\int \psi(\mathrm{f}_j)\psi(\mathrm{f}_j,\mathrm{f}_c)\,d\mathrm{f}_j = \int \exp\left\{-\frac{1}{2}\mathrm{f}_j^\top \mathrm{K}_j^{-1}\mathrm{f}_j - \frac{\|\mathrm{f}_j-\mathrm{f}_c\|^2}{2\sigma_j^2}\right\} d\mathrm{f}_j$$

$$= \int \exp\left\{-\frac{1}{2}\mathrm{f}_j^\top\left(\mathrm{K}_j^{-1}+\frac{1}{\sigma_j^2}\mathrm{I}\right)\mathrm{f}_j + \frac{\mathrm{f}_c^\top}{\sigma_j^2}\mathrm{f}_j - \frac{\|\mathrm{f}_c\|^2}{2\sigma_j^2}\right\} d\mathrm{f}_j \tag{25}$$

$$= \exp\left\{\frac{1}{2}\frac{\mathrm{f}_c^\top}{\sigma_j^2}\left(\mathrm{K}_j^{-1}+\frac{1}{\sigma_j^2}\mathrm{I}\right)^{-1}\frac{\mathrm{f}_c}{\sigma_j^2} - \frac{\|\mathrm{f}_c\|^2}{2\sigma_j^2}\right\} \tag{26}$$

$$= \exp\left\{-\frac{1}{2}\mathrm{f}_c^\top \mathrm{A}_j \mathrm{f}_c\right\}, \tag{27}$$

where

$$\mathrm{A}_j \triangleq \frac{1}{\sigma_j^2}\mathrm{I} - \frac{1}{\sigma_j^2}\left(\mathrm{K}_j^{-1}+\frac{1}{\sigma_j^2}\mathrm{I}\right)^{-1}\frac{1}{\sigma_j^2} = \left(\mathrm{K}_j+\sigma_j^2\mathrm{I}\right)^{-1}.$$

Note that from (25) to (26) we applied the integration result (22). Therefore, from (24) and (27) we have

$$p(\mathrm{y}_l,\mathrm{f}_c) = \frac{1}{Z}\prod_{i=1}^{n_l}\psi(y_i,f_c(\mathrm{x}_i))\exp\left\{-\frac{1}{2}\mathrm{f}_c^\top\left(\sum_j \mathrm{A}_j\right)\mathrm{f}_c\right\},$$

in which the output potentials are equivalent to the conditional density $p(\mathrm{y}_l|\mathrm{f}_c)$, and the big exponential term can be seen as a *prior term* for the consensus function f_c. This leads to the co-training Gaussian prior $p(\mathrm{f}_c)=\mathcal{N}(0,\mathrm{K}_c)$, with $\mathrm{K}_c=(\sum_j \mathrm{A}_j)^{-1}$ being the co-training kernel (10).

A.3 Marginal 3: Individual View Learning with Side-Information

The third marginalization leaves out only the latent function f_j and integrates out the consensus function f_c and all the other latent functions $\{\mathrm{f}_k\}_{k\neq j}$. Ignoring the output potentials for the moment, based on (27) and (22) we have

$$p(\mathrm{f}_j) = \int p(\mathrm{f}_c,\mathrm{f}_1,\ldots,\mathrm{f}_m)\,d\mathrm{f}_c\,d\mathrm{f}_1\ldots d\mathrm{f}_{j-1}\,d\mathrm{f}_{j+1}\ldots d\mathrm{f}_m$$

$$= \frac{1}{Z}\psi(\mathrm{f}_j)\int\left(\psi(\mathrm{f}_j,\mathrm{f}_c)\prod_{k\neq j}\int\psi(\mathrm{f}_k)\psi(\mathrm{f}_k,\mathrm{f}_c)\,d\mathrm{f}_k\right)d\mathrm{f}_c$$

$$= \frac{1}{Z}\psi(\mathrm{f}_j)\int\exp\left\{-\frac{\|\mathrm{f}_j-\mathrm{f}_c\|^2}{2\sigma_j^2} - \frac{1}{2}\mathrm{f}_c^\top\left(\sum_{k\neq j}\mathrm{A}_k\right)\mathrm{f}_c\right\}d\mathrm{f}_c$$

$$= \frac{1}{Z}\psi(\mathrm{f}_j)\int\exp\left\{-\frac{1}{2}\mathrm{f}_c^\top\left(\sum_{k\neq j}\mathrm{A}_k+\frac{1}{\sigma_j^2}\mathrm{I}\right)\mathrm{f}_c + \frac{\mathrm{f}_j^\top}{\sigma_j^2}\mathrm{f}_c - \frac{\|\mathrm{f}_j\|^2}{2\sigma_j^2}\right\}d\mathrm{f}_c$$

$$= \frac{1}{Z}\exp\left\{-\frac{1}{2}\mathrm{f}_j^\top\mathrm{K}_j^{-1}\mathrm{f}_j\right\}\exp\left\{\frac{1}{2}\frac{\mathrm{f}_j^\top}{\sigma_j^2}\left(\sum_{k\neq j}\mathrm{A}_k+\frac{1}{\sigma_j^2}\mathrm{I}\right)^{-1}\frac{\mathrm{f}_j}{\sigma_j^2} - \frac{\|\mathrm{f}_j\|^2}{2\sigma_j^2}\right\}$$

$$= \frac{1}{Z}\exp\left\{-\frac{1}{2}\mathrm{f}_j^\top\mathrm{C}_j^{-1}\mathrm{f}_j\right\},$$

where in the last line we define

$$\begin{aligned} \mathrm{C}_j^{-1} &= \mathrm{K}_j^{-1} + \frac{1}{\sigma_j^2}\mathrm{I} - \frac{1}{\sigma_j^2}\left(\sum_{k\neq j}\mathrm{A}_k + \frac{1}{\sigma_j^2}\mathrm{I}\right)^{-1}\frac{1}{\sigma_j^2} \\ &= \mathrm{K}_j^{-1} + \left(\sigma_j^2\mathrm{I} + \sum_{k\neq j}\mathrm{A}_k\right)^{-1}. \end{aligned}$$

This yields the Equation (11). If we consider the output potentials, a similar GP prior for f_j holds but takes a more sophisticated form.

Appendix B. Optimization of the View Variance Parameters

In this appendix we derive the equations to optimize the view variance σ_j^2 for each view j using the type II maximum likelihood. Under the second marginalization in which only the consensus function f_c is of primary interest, the Bayesian co-training model reduces to

$$p(\mathrm{y}_l, \mathrm{f}_c) = \frac{1}{Z}\psi(\mathrm{f}_c)\prod_{i=1}^{n_l}\psi(y_i, f_c(\mathrm{x}_i)),$$

where $\psi(y_i, f_c(\mathrm{x}_i))$ is the output potential as defined in (1), and $\psi(\mathrm{f}_c)$ is defined via the co-training kernel as

$$\psi(\mathrm{f}_c) = \frac{1}{Z}\exp\left\{-\frac{1}{2}\mathrm{f}_c^\top \mathrm{K}_c^{-1}\mathrm{f}_c\right\}. \tag{28}$$

Note that f_c is of length $n \geq n_l$. This defines a single-view learning problem, and we are effectively assigning a GP prior to f_c with the co-training kernel K_c. The log marginal likelihood of the output y_l under this model, conditioned on the input data $\mathrm{X} \triangleq \{\mathrm{x}_i^{(j)}\}$ and model parameters Θ, is:

$$\mathcal{L} \triangleq \log p(\mathrm{y}_l|\mathrm{X},\Theta) = \log\int p(\mathrm{y}_l|\mathrm{f}_c,\Theta)p(\mathrm{f}_c|\mathrm{X},\Theta)\,d\mathrm{f}_c. \tag{29}$$

In (29) all the probabilities are conditional probabilities, in which $p(\mathrm{y}_l|\mathrm{f}_c,\Theta)$ is defined via (1) and $p(\mathrm{f}_c|\mathrm{X},\Theta)$ is a Gaussian distribution defined via the co-training kernel (28). Here the model parameters Θ contain all the view variance parameters $\{\sigma_j^2\}$, all kernel parameters and other parameters involved in the output potentials. In type II maximum likelihood we maximize (29) with respect to these model parameters. In the following we derive the equations in the regression case, that is, the output potential is a Gaussian noise model. Similar but more complicated equations can be derived for classification case and readers please refer to Rasmussen and Williams (2006) for details.

When the outputs y_l are regression outputs, the integral in (29) can be computed analytically as

$$\mathcal{L} = -\frac{1}{2}\mathrm{y}_l^\top \mathrm{G}^{-1}\mathrm{y}_l - \frac{1}{2}\log\det\mathrm{G} - \frac{n}{2}\log 2\pi,$$

in which for simplicity we rename $\mathrm{G} \triangleq \mathrm{K}_c(1:n_l, 1:n_l) + \sigma^2\mathrm{I}$. Note that since y_l is only of length $n_l \leq n$, matrix G only involves the $n_l \times n_l$ sub-matrix of K_c. For each $\theta \in \Theta$, the partial derivative

of $\mathcal{L}$ with respect to θ is calculated as:

$$\begin{aligned}\frac{\partial \mathcal{L}}{\partial \theta} &= \frac{1}{2} y_l^\top G^{-1} \frac{\partial G}{\partial \theta} G^{-1} y_l - \frac{1}{2} \mathrm{tr}\left[G^{-1} \frac{\partial G}{\partial \theta}\right] \\ &= \frac{1}{2} \mathrm{tr}\left[\left(\alpha\alpha^\top - G^{-1}\right) \frac{\partial G}{\partial \theta}\right], \end{aligned} \tag{30}$$

where $\alpha = G^{-1} y_l$, and $\mathrm{tr}(\cdot)$ denote the matrix trace. We are now ready to calculate the partial derivative of $\mathcal{L}$ with respect to each view variance σ_j^2. We first compute the partial derivative of K_c with respect to σ_j^2 as:

$$\begin{aligned}\frac{\partial K_c}{\partial \sigma_j^2} &= \frac{\partial}{\partial \sigma_j^2}\left[\sum_j \left(K_j + \sigma_j^2 I\right)^{-1}\right]^{-1} \\ &= -K_c \cdot \frac{\partial}{\partial \sigma_j^2}\left(K_j + \sigma_j^2 I\right)^{-1} \cdot K_c \\ &= K_c \left(K_j + \sigma_j^2 I\right)^{-1} \cdot \frac{\partial}{\partial \sigma_j^2}\left(K_j + \sigma_j^2 I\right) \cdot \left(K_j + \sigma_j^2 I\right)^{-1} K_c \\ &= K_c \left(K_j + \sigma_j^2 I\right)^{-1} \left(K_j + \sigma_j^2 I\right)^{-1} K_c. \end{aligned}$$

Then if we name matrix $B_j \triangleq K_c (K_j + \sigma_j^2 I)^{-1} (K_j + \sigma_j^2 I)^{-1} K_c$, we have

$$\frac{\partial G}{\partial \sigma_j^2} = \frac{\partial}{\partial \sigma_j^2} K_c(1:n_l, 1:n_l) = B_j(1:n_l, 1:n_l). \tag{31}$$

This equation follows since we have

$$\begin{aligned}\frac{\partial}{\partial \sigma_j^2} K_c(1:n_l, 1:n_l) &= \frac{\partial}{\partial \sigma_j^2} \begin{pmatrix} I_{n_l} & 0 \end{pmatrix} \cdot K_c \cdot \begin{pmatrix} I_{n_l} \\ 0 \end{pmatrix} \\ &= \begin{pmatrix} I_{n_l} & 0 \end{pmatrix} \cdot \frac{\partial}{\partial \sigma_j^2} K_c \cdot \begin{pmatrix} I_{n_l} \\ 0 \end{pmatrix} \\ &= \begin{pmatrix} I_{n_l} & 0 \end{pmatrix} \cdot B_j \cdot \begin{pmatrix} I_{n_l} \\ 0 \end{pmatrix} \\ &= B_j(1:n_l, 1:n_l). \end{aligned}$$

Note that even though we only need to consider the top left corner of matrix B_j in the derivative calculation, each entry in this sub-matrix depends both on labeled data and on unlabeled data. This provides some additional insight since even with f_c integrated out, the marginal likelihood still depends on unlabeled data, so as the optimization of the hyperparameters σ_j^2.

With (30) and (31) we can calculate $\partial \mathcal{L} / \partial \sigma_j^2$ and then use conjugate gradients to find the optimal σ_j^2. Since the derivatives for the different σ_j^2 are coupled, one needs to iteratively optimize each σ_j^2 until convergence. The partial derivative for σ^2 can be easily computed as $\frac{\partial G}{\partial \sigma^2} = I_{n_l}$. Similarly one can derive the partial derivatives for other kernel parameters inside each kernel K_j and we omit the details.

References

M. Balcan and A. Blum. A PAC-style model for learning from labeled and unlabeled data. In *Semi-Supervised Learning*, pages 111–126. MIT Press, 2006.

M. Balcan, A. Blum, and K. Yang. Co-training and expansion: Towards bridging theory and practice. In *NIPS*, 2004.

S. Bickel and T. Scheffer. Estimation of mixture models using Co-EM. In *ECML*, 2005.

M. Bilgic and L. Getoor. VOILA: Efficient feature-value acquisition for classification. In *AAAI*, 2007.

A. Blum and T. Mitchell. Combining labeled and unlabeled data with co-training. In *COLT*, 1998.

U. Brefeld and T. Scheffer. Co-EM support vector learning. In *ICML*, 2004.

U. Brefeld, T. Gärtner, T. Scheffer, and S. Wrobel. Efficient co-regularised least squares regression. In *ICML*, pages 137–144, 2006.

C. Capirci, L. Rampin, P. Erba, F. Galeotti, G. Crepaldi, E. Banti, M. Gava, S. Fanti, G. Mariani, P. Muzzio, and D. Rubello. Sequential FDG-PET/CT reliably predicts response of locally advanced rectal cancer to neo-adjuvant chemo-radiation therapy. *Eur J Nucl Med Mol Imaging*, 34, 2007.

A. Corduneanu and C. M. Bishop. Variational Bayesian model selection for mixture distributions. In *Workshop AI and Statistics*, pages 27—34, 2001.

S. Dasgupta, M. Littman, and D. McAllester. PAC generalization bounds for co-training. In *NIPS*, 2001.

V. de Sa. Spectral clustering with two views. In *ICML Workshop on Learning With Multiple Views*, 2005.

C. Dehing-Oberije, S. Yu, D. De Ruysscher, S. Meerschout, K. van Beek, Y. Lievens, J. van Meerbeeck, W. de Neve, G. Fung, B. Rao, S. Krishnan, H. van der Weide, and P. Lambin. Development and external validation of prognostic model for 2-year survival of non-small-cell lung cancer patients treated with chemoradiotherapy. *Int J Radiat Oncol Biol Phys*, 2009.

J. Farquhar, D. Hardoon, H. Meng, J-S. Taylor, and S. Szedmak. Two view learning: SVM-2K, Theory and Practice. In *NIPS*, 2005.

R. Hwa, M. Osborne, A.Sarkar, and M. Steedman. Corrected co-training for statistical parsers. In *ICML Workshop The Continuum from Labeled to Unlabeled Data*, 2003.

S. Kiritchenko and S. Matwin. Email classification with co-training. Technical report, University of Ottawa, 2002.

A. Krause, A. Singh, and C. Guestrin. Near-optimal sensor placements in Gaussian processes: Theory, efficient algorithms and empirical studies. *JMLR*, 9:235–284, 2008.

B. Krishnapuram, D. Williams, Y. Xue, A. Hartemink, L. Carin, and M. Figueiredo. On semi-supervised classification. In *NIPS*, 2004.

D. MacKay. Information-based objective functions for active data selection. *Neural Computation*, 4:590–604, 1992.

P. Melville, M. Saar-Tsechansky, F. Provost, and R. Mooney. Active feature-value acquisition for classifier induction. In *IEEE International Conference on Data Mining*, 2004.

K. Nigam and R. Ghani. Analyzing the effectiveness and applicability of co-training. In *Workshop on information and knowledge management*, 2000.

D. Pierce and C. Cardie. Limitations of co-training for natural language learning from large datasets. In *EMNLP-2001*, 2001.

C. E. Rasmussen and C. K. I. Williams. *Gaussian Processes for Machine Learning*. MIT Press, 2006.

V. Sindhwani and D. S. Rosenberg. An RKHS for multi-view learning and manifold co-regularization. In *ICML*, 2008.

V. Sindhwani, P. Niyogi, and M. Belkin. A co-regularization approach to semi-supervised learning with multiple views. *ICML Workshop on Learning With Multiple Views*, 2005.

K. Sridharan and S. M. Kakade. An information theoretic framework for multi-view learning. In *COLT*, 2008.

W. Wang and Z.-H. Zhou. Analyzing co-training style algorithms. In *European Conference on Machine Learning*, 2007.

W. Wang and Z.-H. Zhou. A new analysis of co-training. In *International Conference on Machine Learning*, 2010.

K. Yu, V. Tresp, and A. Schwaighofer. Learning Gaussian processes from multiple tasks. In *International Conference on Machine Learning*, 2005.

S. Yu, B. Krishnapuram, R. Rosales, H. Steck, and B. Rao. Bayesian co-training. In *NIPS*, 2008.

X. Zhu, J. Lafferty, and Z. Ghahramani. Semi-supervised learning: from Gaussian fields to Gaussian processes. Technical report, CMU-CS-03-175, 2003.

Journal of Machine Learning Research 12 (2011) 2681-2720 Submitted 4/11; Revised 8/11; Published 9/11

Convex and Network Flow Optimization for Structured Sparsity

Julien Mairal[*†] JULIEN@STAT.BERKELEY.EDU
Department of Statistics
University of California
Berkeley, CA 94720-1776, USA

Rodolphe Jenatton[*†] RODOLPHE.JENATTON@INRIA.FR
Guillaume Obozinski[†] GUILLAUME.OBOZINSKI@INRIA.FR
Francis Bach[†] FRANCIS.BACH@INRIA.FR
INRIA - SIERRA Project-Team
Laboratoire d'Informatique de l'Ecole Normale Supérieure (INRIA/ENS/CNRS UMR 8548)
23, avenue d'Italie 75214 Paris CEDEX 13, France

Editor: Hui Zou

Abstract

We consider a class of learning problems regularized by a structured sparsity-inducing norm defined as the sum of ℓ_2- or ℓ_∞-norms over groups of variables. Whereas much effort has been put in developing fast optimization techniques when the groups are disjoint or embedded in a hierarchy, we address here the case of general overlapping groups. To this end, we present two different strategies: On the one hand, we show that the proximal operator associated with a sum of ℓ_∞-norms can be computed exactly in polynomial time by solving a *quadratic min-cost flow problem*, allowing the use of accelerated proximal gradient methods. On the other hand, we use proximal splitting techniques, and address an equivalent formulation with non-overlapping groups, but in higher dimension and with additional constraints. We propose efficient and scalable algorithms exploiting these two strategies, which are significantly faster than alternative approaches. We illustrate these methods with several problems such as CUR matrix factorization, multi-task learning of tree-structured dictionaries, background subtraction in video sequences, image denoising with wavelets, and topographic dictionary learning of natural image patches.

Keywords: convex optimization, proximal methods, sparse coding, structured sparsity, matrix factorization, network flow optimization, alternating direction method of multipliers

1. Introduction

Sparse linear models have become a popular framework for dealing with various unsupervised and supervised tasks in machine learning and signal processing. In such models, linear combinations of small sets of variables are selected to describe the data. Regularization by the ℓ_1-norm has emerged as a powerful tool for addressing this variable selection problem, relying on both a well-developed theory (see Tibshirani, 1996; Chen et al., 1999; Mallat, 1999; Bickel et al., 2009; Wainwright, 2009, and references therein) and efficient algorithms (Efron et al., 2004; Nesterov, 2007; Beck and Teboulle, 2009; Needell and Tropp, 2009; Combettes and Pesquet, 2010).

∗. These authors contributed equally.
†. When most of this work was conducted, all authors were affiliated to INRIA, WILLOW Project-Team.

The ℓ_1-norm primarily encourages sparse solutions, regardless of the potential structural relationships (e.g., spatial, temporal or hierarchical) existing between the variables. Much effort has recently been devoted to designing sparsity-inducing regularizations capable of encoding higher-order information about the patterns of non-zero coefficients (Cehver et al., 2008; Jenatton et al., 2009; Jacob et al., 2009; Zhao et al., 2009; He and Carin, 2009; Huang et al., 2009; Baraniuk et al., 2010; Micchelli et al., 2010), with successful applications in bioinformatics (Jacob et al., 2009; Kim and Xing, 2010), topic modeling (Jenatton et al., 2010a, 2011) and computer vision (Cehver et al., 2008; Huang et al., 2009; Jenatton et al., 2010b). By considering sums of norms of appropriate subsets, or *groups*, of variables, these regularizations control the sparsity patterns of the solutions. The underlying optimization is usually difficult, in part because it involves nonsmooth components.

Our first strategy uses proximal gradient methods, which have proven to be effective in this context, essentially because of their fast convergence rates and their ability to deal with large problems (Nesterov, 2007; Beck and Teboulle, 2009). They can handle differentiable loss functions with Lipschitz-continuous gradient, and we show in this paper how to use them with a regularization term composed of a sum of ℓ_∞-norms. The second strategy we consider exploits proximal splitting methods (see Combettes and Pesquet, 2008, 2010; Goldfarg and Ma, 2009; Tomioka et al., 2011; Qin and Goldfarb, 2011; Boyd et al., 2011, and references therein), which builds upon an equivalent formulation with non-overlapping groups, but in a higher dimensional space and with additional constraints.[1] More precisely, we make four main contributions:

- We show that the *proximal operator* associated with the sum of ℓ_∞-norms with overlapping groups can be computed efficiently and exactly by solving a *quadratic min-cost flow* problem, thereby establishing a connection with the network flow optimization literature.[2] This is the main contribution of the paper, which allows us to use proximal gradient methods in the context of structured sparsity.
- We prove that the dual norm of the sum of ℓ_∞-norms can also be evaluated efficiently, which enables us to compute duality gaps for the corresponding optimization problems.
- We present proximal splitting methods for solving structured sparse regularized problems.
- We demonstrate that our methods are relevant for various applications whose practical success is made possible by our algorithmic tools and efficient implementations. First, we introduce a new CUR matrix factorization technique exploiting structured sparse regularization, built upon the links drawn by Bien et al. (2010) between CUR decomposition (Mahoney and Drineas, 2009) and sparse regularization. Then, we illustrate our algorithms with different tasks: video background subtraction, estimation of hierarchical structures for dictionary learning of natural image patches (Jenatton et al., 2010a, 2011), wavelet image denoising

1. The idea of using this class of algorithms for solving structured sparse problems was first suggested to us by Jean-Christophe Pesquet and Patrick-Louis Combettes. It was also suggested to us later by Ryota Tomioka, who briefly mentioned this possibility in Tomioka et al. (2011). It can also briefly be found in Boyd et al. (2011), and in details in the work of Qin and Goldfarb (2011) which was conducted as the same time as ours. It was also used in a related context by Sprechmann et al. (2010) for solving optimization problems with hierarchical norms.
2. Interestingly, this is not the first time that network flow optimization tools have been used to solve sparse regularized problems with proximal methods. Such a connection was recently established by Chambolle and Darbon (2009) in the context of total variation regularization, and similarly by Hoefling (2010) for the fused Lasso. One can also find the use of maximum flow problems for non-convex penalties in the work of Cehver et al. (2008) which combines Markov random fields and sparsity.

with a structured sparse prior, and topographic dictionary learning of natural image patches (Hyvärinen et al., 2001; Kavukcuoglu et al., 2009; Garrigues and Olshausen, 2010).

Note that this paper extends a shorter version published in Advances in Neural Information Processing Systems (Mairal et al., 2010b), by adding new experiments (CUR matrix factorization, wavelet image denoising and topographic dictionary learning), presenting the proximal splitting methods, providing the full proofs of the optimization results, and adding numerous discussions.

1.1 Notation

Vectors are denoted by bold lower case letters and matrices by upper case ones. We define for $q \geq 1$ the ℓ_q-norm of a vector $\mathbf{x}$ in $\mathbb{R}^m$ as $\|\mathbf{x}\|_q \triangleq (\sum_{i=1}^m |\mathbf{x}_i|^q)^{1/q}$, where $\mathbf{x}_i$ denotes the i-th coordinate of $\mathbf{x}$, and $\|\mathbf{x}\|_\infty \triangleq \max_{i=1,\ldots,m} |\mathbf{x}_i| = \lim_{q\to\infty} \|\mathbf{x}\|_q$. We also define the ℓ_0-pseudo-norm as the number of nonzero elements in a vector:[3] $\|\mathbf{x}\|_0 \triangleq \#\{i \text{ s.t. } \mathbf{x}_i \neq 0\} = \lim_{q\to 0^+} (\sum_{i=1}^m |\mathbf{x}_i|^q)$. We consider the Frobenius norm of a matrix $\mathbf{X}$ in $\mathbb{R}^{m\times n}$: $\|\mathbf{X}\|_{\mathrm{F}} \triangleq (\sum_{i=1}^m \sum_{j=1}^n \mathbf{X}_{ij}^2)^{1/2}$, where $\mathbf{X}_{ij}$ denotes the entry of $\mathbf{X}$ at row i and column j. Finally, for a scalar y, we denote $(y)_+ \triangleq \max(y,0)$. For an integer $p > 0$, we denote by $2^{\{1,\ldots,p\}}$ the powerset composed of the 2^p subsets of $\{1,\ldots,p\}$.

The rest of this paper is organized as follows: Section 2 presents structured sparse models and related work. Section 3 is devoted to proximal gradient algorithms, and Section 4 to proximal splitting methods. Section 5 presents several experiments and applications demonstrating the effectiveness of our approach and Section 6 concludes the paper.

2. Structured Sparse Models

We are interested in machine learning problems where the solution is not only known beforehand to be sparse—that is, the solution has only a few non-zero coefficients, but also to form non-zero patterns with a specific structure. It is indeed possible to encode additional knowledge in the regularization other than just sparsity. For instance, one may want the non-zero patterns to be structured in the form of non-overlapping groups (Turlach et al., 2005; Yuan and Lin, 2006; Stojnic et al., 2009; Obozinski et al., 2010), in a tree (Zhao et al., 2009; Bach, 2009; Jenatton et al., 2010a, 2011), or in overlapping groups (Jenatton et al., 2009; Jacob et al., 2009; Huang et al., 2009; Baraniuk et al., 2010; Cehver et al., 2008; He and Carin, 2009), which is the setting we are interested in here.

As for classical non-structured sparse models, there are basically two lines of research, that either (A) deal with nonconvex and combinatorial formulations that are in general computationally intractable and addressed with greedy algorithms or (B) concentrate on convex relaxations solved with convex programming methods.

2.1 Nonconvex Approaches

A first approach introduced by Baraniuk et al. (2010) consists in imposing that the sparsity pattern of a solution (i.e., its set of non-zero coefficients) is in a predefined subset of groups of variables $\mathcal{G} \subseteq 2^{\{1,\ldots,p\}}$. Given this a priori knowledge, a greedy algorithm (Needell and Tropp, 2009) is used

3. Note that it would be more proper to write $\|\mathbf{x}\|_0^0$ instead of $\|\mathbf{x}\|_0$ to be consistent with the traditional notation $\|\mathbf{x}\|_q$. However, for the sake of simplicity, we will keep this notation unchanged in the rest of the paper.

to address the following nonconvex structured sparse decomposition problem

$$\min_{\mathbf{w}\in\mathbb{R}^p} \frac{1}{2}\|\mathbf{y}-\mathbf{X}\mathbf{w}\|_2^2 \text{ s.t. } \text{Supp}(\mathbf{w}) \in \mathcal{G} \text{ and } \|\mathbf{w}\|_0 \leq s,$$

where s is a specified sparsity level (number of nonzeros coefficients), $\mathbf{y}$ in $\mathbb{R}^m$ is an observed signal, $\mathbf{X}$ is a design matrix in $\mathbb{R}^{m\times p}$ and $\text{Supp}(\mathbf{w})$ is the support of $\mathbf{w}$ (set of non-zero entries).

In a different approach motivated by the minimum description length principle (see Barron et al., 1998), Huang et al. (2009) consider a collection of groups $\mathcal{G} \subseteq 2^{\{1,\ldots,p\}}$, and define a "coding length" for every group in $\mathcal{G}$, which in turn is used to define a coding length for every pattern in $2^{\{1,\ldots,p\}}$. Using this tool, they propose a regularization function $\text{cl} : \mathbb{R}^p \to \mathbb{R}$ such that for a vector $\mathbf{w}$ in $\mathbb{R}^p$, $\text{cl}(\mathbf{w})$ represents the number of bits that are used for encoding $\mathbf{w}$. The corresponding optimization problem is also addressed with a greedy procedure:

$$\min_{\mathbf{w}\in\mathbb{R}^p} \frac{1}{2}\|\mathbf{y}-\mathbf{X}\mathbf{w}\|_2^2 \text{ s.t. } \text{cl}(\mathbf{w}) \leq s,$$

Intuitively, this formulation encourages solutions $\mathbf{w}$ whose sparsity patterns have a small coding length, meaning in practice that they can be represented by a union of a small number of groups. Even though they are related, this model is different from the one of Baraniuk et al. (2010).

These two approaches are encoding a priori knowledge on the shape of non-zero patterns that the solution of a regularized problem should have. A different point of view consists of modelling the zero patterns of the solution—that is, define groups of variables that should be encouraged to be set to zero together. After defining a set $\mathcal{G} \subseteq 2^{\{1,\ldots,p\}}$ of such groups of variables, the following penalty can naturally be used as a regularization to induce the desired property

$$\psi(\mathbf{w}) \triangleq \sum_{g\in\mathcal{G}} \eta_g \delta^g(\mathbf{w}), \text{ with } \delta^g(\mathbf{w}) \triangleq \begin{cases} 1 & \text{if there exists } j \in g \text{ such that } \mathbf{w}_j \neq 0, \\ 0 & \text{otherwise,} \end{cases}$$

where the η_g's are positive weights. This penalty was considered by Bach (2010), who showed that the convex envelope of such nonconvex functions (more precisely strictly positive, non-increasing submodular functions of $\text{Supp}(\mathbf{w})$, see Fujishige, 2005) when restricted on the unit ℓ_∞-ball, are in fact types of structured sparsity-inducing norms which are the topic of the next section.

2.2 Convex Approaches with Sparsity-Inducing Norms

In this paper, we are interested in convex regularizations which induce structured sparsity. Generally, we consider the following optimization problem

$$\min_{\mathbf{w}\in\mathbb{R}^p} f(\mathbf{w}) + \lambda\Omega(\mathbf{w}), \tag{1}$$

where $f : \mathbb{R}^p \to \mathbb{R}$ is a convex function (usually an empirical risk in machine learning and a data-fitting term in signal processing), and $\Omega : \mathbb{R}^p \to \mathbb{R}$ is a structured sparsity-inducing norm, defined as

$$\Omega(\mathbf{w}) \triangleq \sum_{g\in\mathcal{G}} \eta_g \|\mathbf{w}_g\|, \tag{2}$$

where $\mathcal{G} \subseteq 2^{\{1,\ldots,p\}}$ is a set of groups of variables, the vector $\mathbf{w}_g$ in $\mathbb{R}^{|g|}$ represents the coefficients of $\mathbf{w}$ indexed by g in $\mathcal{G}$, the scalars η_g are positive weights, and $\|.\|$ denotes the ℓ_2- or ℓ_∞-norm. We now consider different cases:

- When $\mathcal{G}$ is the set of singletons—that is $\mathcal{G} \triangleq \{\{1\},\{2\},\ldots,\{p\}\}$, and all the η_g are equal to one, Ω is the ℓ_1-norm, which is well known to induce sparsity. This leads for instance to the Lasso (Tibshirani, 1996) or equivalently to basis pursuit (Chen et al., 1999).
- If $\mathcal{G}$ is a partition of $\{1,\ldots,p\}$, that is, the groups do not overlap, variables are selected in groups rather than individually. When the coefficients of the solution are known to be organized in such a way, explicitly encoding the a priori group structure in the regularization can improve the prediction performance and/or interpretability of the learned models (Turlach et al., 2005; Yuan and Lin, 2006; Roth and Fischer, 2008; Stojnic et al., 2009; Huang and Zhang, 2010; Obozinski et al., 2010). Such a penalty is commonly called group-Lasso penalty.
- When the groups overlap, Ω is still a norm and sets groups of variables to zero together (Jenatton et al., 2009). The latter setting has first been considered for hierarchies (Zhao et al., 2009; Kim and Xing, 2010; Bach, 2009; Jenatton et al., 2010a, 2011), and then extended to general group structures (Jenatton et al., 2009). Solving Equation (1) in this context is a challenging problem which is the topic of this paper.

Note that other types of structured-sparsity inducing norms have also been introduced, notably the approach of Jacob et al. (2009), which penalizes the following quantity

$$\Omega'(\mathbf{w}) \triangleq \min_{\xi=(\xi^g)_{g\in\mathcal{G}}\in\mathbb{R}^{p\times|\mathcal{G}|}} \sum_{g\in\mathcal{G}} \eta_g \|\xi^g\| \text{ s.t. } \mathbf{w} = \sum_{g\in\mathcal{G}} \xi^g \text{ and } \forall g,\ \mathrm{Supp}(\xi^g) \subseteq g.$$

This penalty, which is also a norm, can be seen as a convex relaxation of the regularization introduced by Huang et al. (2009), and encourages the sparsity pattern of the solution to be a union of a small number of groups. Even though both Ω and Ω' appear under the terminology of "structured sparsity with overlapping groups", they have in fact significantly different purposes and algorithmic treatments. For example, Jacob et al. (2009) consider the problem of selecting genes in a gene network which can be represented as the union of a few predefined pathways in the graph (groups of genes), which overlap. In this case, it is natural to use the norm Ω' instead of Ω. On the other hand, we present a matrix factorization task in Section 5.3, where the set of zero-patterns should be a union of groups, naturally leading to the use of Ω. Dealing with Ω' is therefore relevant, but out of the scope of this paper.

2.3 Convex Optimization Methods Proposed in the Literature

Generic approaches to solve Equation (1) mostly rely on subgradient descent schemes (see Bertsekas, 1999), and interior-point methods (Boyd and Vandenberghe, 2004). These generic tools do not scale well to large problems and/or do not naturally handle sparsity (the solutions they return may have small values but no "true" zeros). These two points prompt the need for dedicated methods.

To the best of our knowledge, only a few recent papers have addressed problem Equation (1) with dedicated optimization procedures, and in fact, only when Ω is a linear combination of ℓ_2-norms. In this setting, a first line of work deals with the non-smoothness of Ω by expressing the norm as the minimum over a set of smooth functions. At the cost of adding new variables (to describe the set of smooth functions), the problem becomes more amenable to optimization. In particular, reweighted-ℓ_2 schemes consist of approximating the norm Ω by successive quadratic

upper bounds (Argyriou et al., 2008; Rakotomamonjy et al., 2008; Jenatton et al., 2010b; Micchelli et al., 2010). It is possible to show for instance that

$$\Omega(\mathbf{w}) = \min_{(z_g)_{g\in\mathcal{G}}\in\mathbb{R}_+^{|\mathcal{G}|}} \frac{1}{2}\Big\{ \sum_{g\in\mathcal{G}} \frac{\eta_g^2\|\mathbf{w}_g\|_2^2}{z_g} + z_g \Big\}.$$

Plugging the previous relationship into Equation (1), the optimization can then be performed by alternating between the updates of $\mathbf{w}$ and the additional variables $(z_g)_{g\in\mathcal{G}}$.[4] When the norm Ω is defined as a linear combination of ℓ_∞-norms, we are not aware of the existence of such variational formulations.

Problem (1) has also been addressed with working-set algorithms (Bach, 2009; Jenatton et al., 2009; Schmidt and Murphy, 2010). The main idea of these methods is to solve a sequence of increasingly larger subproblems of (1). Each subproblem consists of an instance of Equation (1) reduced to a specific subset of variables known as the *working set*. As long as some predefined optimality conditions are not satisfied, the working set is augmented with selected inactive variables (for more details, see Bach et al., 2011).

The last approach we would like to mention is that of Chen et al. (2010), who used a smoothing technique introduced by Nesterov (2005). A smooth approximation Ω_μ of Ω is used, when Ω is a sum of ℓ_2-norms, and μ is a parameter controlling the trade-off between smoothness of Ω_μ and quality of the approximation. Then, Equation (1) is solved with accelerated gradient techniques (Beck and Teboulle, 2009; Nesterov, 2007) but Ω_μ is substituted to the regularization Ω. Depending on the required precision for solving the original problem, this method provides a natural choice for the parameter μ, with a known convergence rate. A drawback is that it requires to choose the precision of the optimization beforehand. Moreover, since a ℓ_1-norm is added to the smoothed Ω_μ, the solutions returned by the algorithm might be sparse but possibly without respecting the structure encoded by Ω. This should be contrasted with other smoothing techniques, for example, the reweighted-ℓ_2 scheme we mentioned above, where the solutions are only approximately sparse.

3. Optimization with Proximal Gradient Methods

We address in this section the problem of solving Equation (1) under the following assumptions:

- *f is differentiable with Lipschitz-continuous gradient*. For machine learning problems, this hypothesis holds when f is for example the square, logistic or multi-class logistic loss (see Shawe-Taylor and Cristianini, 2004).
- *Ω is a sum of ℓ_∞-norms*. Even though the ℓ_2-norm is sometimes used in the literature (Jenatton et al., 2009), and is in fact used later in Section 4, the ℓ_∞-norm is piecewise linear, and we take advantage of this property in this work.

To the best of our knowledge, no dedicated optimization method has been developed for this setting. Following Jenatton et al. (2010a, 2011) who tackled the particular case of hierarchical norms, we propose to use proximal gradient methods, which we now introduce.

4. Note that such a scheme is interesting only if the optimization with respect to $\mathbf{w}$ is simple, which is typically the case with the square loss function (Bach et al., 2011). Moreover, for this alternating scheme to be provably convergent, the variables $(z_g)_{g\in\mathcal{G}}$ have to be bounded away from zero, resulting in solutions whose entries may have small values, but not "true" zeros.

3.1 Proximal Gradient Methods

Proximal methods have drawn increasing attention in the signal processing (e.g., Wright et al., 2009b; Combettes and Pesquet, 2010, and numerous references therein) and the machine learning communities (e.g., Bach et al., 2011, and references therein), especially because of their convergence rates (optimal for the class of first-order techniques) and their ability to deal with large nonsmooth convex problems (e.g., Nesterov, 2007; Beck and Teboulle, 2009).

These methods are iterative procedures that can be seen as an extension of gradient-based techniques when the objective function to minimize has a nonsmooth part. The simplest version of this class of methods linearizes at each iteration the function f around the current estimate $\tilde{\mathbf{w}}$, and this estimate is updated as the (unique by strong convexity) solution of the *proximal* problem, defined as:

$$\min_{\mathbf{w}\in\mathbb{R}^p} f(\tilde{\mathbf{w}}) + (\mathbf{w}-\tilde{\mathbf{w}})^\top \nabla f(\tilde{\mathbf{w}}) + \lambda\Omega(\mathbf{w}) + \frac{L}{2}\|\mathbf{w}-\tilde{\mathbf{w}}\|_2^2.$$

The quadratic term keeps the update in a neighborhood where f is close to its linear approximation, and $L>0$ is a parameter which is a upper bound on the Lipschitz constant of ∇f. This problem can be equivalently rewritten as:

$$\min_{\mathbf{w}\in\mathbb{R}^p} \frac{1}{2}\left\|\tilde{\mathbf{w}} - \frac{1}{L}\nabla f(\tilde{\mathbf{w}}) - \mathbf{w}\right\|_2^2 + \frac{\lambda}{L}\Omega(\mathbf{w}),$$

Solving *efficiently* and exactly this problem allows to attain the fast convergence rates of proximal methods, that is, reaching a precision of $O(\frac{L}{k^2})$ in k iterations.[5] In addition, when the nonsmooth term Ω is not present, the previous proximal problem exactly leads to the standard gradient update rule. More generally, we define the *proximal operator*:

Definition 1 (Proximal Operator)
The proximal operator associated with our regularization term $\lambda\Omega$, which we denote by $Prox_{\lambda\Omega}$, is the function that maps a vector $\mathbf{u}\in\mathbb{R}^p$ to the unique solution of

$$\min_{\mathbf{w}\in\mathbb{R}^p} \frac{1}{2}\|\mathbf{u}-\mathbf{w}\|_2^2 + \lambda\Omega(\mathbf{w}). \tag{3}$$

This operator was initially introduced by Moreau (1962) to generalize the projection operator onto a convex set. What makes proximal methods appealing to solve sparse decomposition problems is that this operator can often be computed in closed form. For instance,

- When Ω is the ℓ_1-norm—that is $\Omega(\mathbf{w}) = \|\mathbf{w}\|_1$—the proximal operator is the well-known elementwise soft-thresholding operator,

$$\forall j\in\{1,\ldots,p\},\ \ \mathbf{u}_j \mapsto \operatorname{sign}(\mathbf{u}_j)(|\mathbf{u}_j|-\lambda)_+ \quad = \begin{cases} 0 & \text{if } |\mathbf{u}_j|\le\lambda \\ \operatorname{sign}(\mathbf{u}_j)(|\mathbf{u}_j|-\lambda) & \text{otherwise.}\end{cases}$$

- When Ω is a group-Lasso penalty with ℓ_2-norms—that is, $\Omega(\mathbf{u}) = \sum_{g\in\mathcal{G}}\|\mathbf{u}_g\|_2$, with $\mathcal{G}$ being a partition of $\{1,\ldots,p\}$, the proximal problem is *separable* in every group, and the solution is a generalization of the soft-thresholding operator to groups of variables:

$$\forall g\in\mathcal{G}\ \ ,\mathbf{u}_g \mapsto \mathbf{u}_g - \Pi_{\|.\|_2\le\lambda}[\mathbf{u}_g] = \begin{cases} 0 & \text{if } \|\mathbf{u}_g\|_2\le\lambda \\ \frac{\|\mathbf{u}_g\|_2-\lambda}{\|\mathbf{u}_g\|_2}\mathbf{u}_g & \text{otherwise,}\end{cases}$$

5. Note, however, that fast convergence rates can also be achieved while solving approximately the proximal problem (see Schmidt et al., 2011, for more details).

where $\Pi_{\|.\|_2 \leq \lambda}$ denotes the orthogonal projection onto the ball of the ℓ_2-norm of radius λ.

- When Ω is a group-Lasso penalty with ℓ_∞-norms—that is, $\Omega(\mathbf{u}) = \sum_{g \in \mathcal{G}} \|\mathbf{u}_g\|_\infty$, with $\mathcal{G}$ being a partition of $\{1,\dots,p\}$, the solution is a different group-thresholding operator:

$$\forall g \in \mathcal{G}, \;\; \mathbf{u}_g \mapsto \mathbf{u}_g - \Pi_{\|.\|_1 \leq \lambda}[\mathbf{u}_g],$$

where $\Pi_{\|.\|_1 \leq \lambda}$ denotes the orthogonal projection onto the ℓ_1-ball of radius λ, which can be solved in $O(p)$ operations (Brucker, 1984; Maculan and de Paula, 1989). Note that when $\|\mathbf{u}_g\|_1 \leq \lambda$, we have a group-thresholding effect, with $\mathbf{u}_g - \Pi_{\|.\|_1 \leq \lambda}[\mathbf{u}_g] = 0$.

- When Ω is a tree-structured sum of ℓ_2- or ℓ_∞-norms as introduced by Zhao et al. (2009)—meaning that two groups are either disjoint or one is included in the other, the solution admits a closed form. Let $\preceq$ be a total order on $\mathcal{G}$ such that for g_1, g_2 in $\mathcal{G}$, $g_1 \preceq g_2$ if and only if either $g_1 \subset g_2$ or $g_1 \cap g_2 = \emptyset$.[6] Then, if $g_1 \preceq \ldots \preceq g_{|\mathcal{G}|}$, and if we define Prox^g as (a) the proximal operator $\mathbf{u}_g \mapsto \text{Prox}_{\lambda\eta_g\|.\|}(\mathbf{u}_g)$ on the subspace corresponding to group g and (b) the identity on the orthogonal, Jenatton et al. (2010a, 2011) showed that:

$$\text{Prox}_{\lambda\Omega} = \text{Prox}^{g_m} \circ \ldots \circ \text{Prox}^{g_1},$$

which can be computed in $O(p)$ operations. It also includes the sparse group Lasso (sum of group-Lasso penalty and ℓ_1-norm) of Friedman et al. (2010) and Sprechmann et al. (2010).

The first contribution of our paper is to address the case of general overlapping groups with ℓ_∞-norm.

3.2 Dual of the Proximal Operator

We now show that, for a set $\mathcal{G}$ of general overlapping groups, a convex dual of the proximal problem (3) can be reformulated as a *quadratic min-cost flow problem*. We then propose an efficient algorithm to solve it exactly, as well as a related algorithm to compute the dual norm of Ω. We start by considering the dual formulation to problem (3) introduced by Jenatton et al. (2010a, 2011):

Lemma 2 (Dual of the proximal problem, Jenatton et al., 2010a, 2011)
Given $\mathbf{u}$ *in* $\mathbb{R}^p$*, consider the problem*

$$\min_{\xi \in \mathbb{R}^{p \times |\mathcal{G}|}} \frac{1}{2}\|\mathbf{u} - \sum_{g \in \mathcal{G}} \xi^g\|_2^2 \quad s.t. \quad \forall g \in \mathcal{G}, \; \|\xi^g\|_1 \leq \lambda\eta_g \quad \text{and} \quad \xi^g_j = 0 \text{ if } j \notin g, \tag{4}$$

where $\xi = (\xi^g)_{g \in \mathcal{G}}$ *is in* $\mathbb{R}^{p \times |\mathcal{G}|}$*, and* ξ^g_j *denotes the* j*-th coordinate of the vector* ξ^g*. Then, every solution* $\xi^\star = (\xi^{\star g})_{g \in \mathcal{G}}$ *of Equation (4) satisfies* $\mathbf{w}^\star = \mathbf{u} - \sum_{g \in \mathcal{G}} \xi^{\star g}$*, where* $\mathbf{w}^\star$ *is the solution of Equation (3) when* Ω *is a weighted sum of* ℓ_∞*-norms.*

Without loss of generality,[7] we assume from now on that the scalars $\mathbf{u}_j$ are all non-negative, and we constrain the entries of ξ to be so. Such a formulation introduces $p|\mathcal{G}|$ dual variables which can be much greater than p, the number of primal variables, but it removes the issue of overlapping regularization. We now associate a graph with problem (4), on which the variables ξ^g_j, for g in $\mathcal{G}$ and j in g, can be interpreted as measuring the components of a flow.

6. For a tree-structured set $\mathcal{G}$, such an order exists.

7. Let $\xi^\star$ denote a solution of Equation (4). Optimality conditions of Equation (4) derived in Jenatton et al. (2010a, 2011) show that for all j in $\{1,\dots,p\}$, the signs of the non-zero coefficients $\xi^{\star g}_j$ for g in $\mathcal{G}$ are the same as the signs of the entries $\mathbf{u}_j$. To solve Equation (4), one can therefore flip the signs of the negative variables $\mathbf{u}_j$, then solve the modified dual formulation (with non-negative variables), which gives the magnitude of the entries $\xi^{\star g}_j$ (the signs of these being known).

3.3 Graph Model

Let G be a directed graph $G = (V, E, s, t)$, where V is a set of vertices, $E \subseteq V \times V$ a set of arcs, s a source, and t a sink. For all arcs in E, we define a non-negative capacity constant, and as done classically in the network flow literature (Ahuja et al., 1993; Bertsekas, 1998), we define a *flow* as a non-negative function on arcs that satisfies capacity constraints on all arcs (the value of the flow on an arc is less than or equal to the arc capacity) and conservation constraints on all vertices (the sum of incoming flows at a vertex is equal to the sum of outgoing flows) except for the source and the sink. For every arc e in E, we also define a real-valued cost function, which depends on the value of the flow on e. We now introduce the *canonical* graph G associated with our optimization problem:

Definition 3 (Canonical Graph)
Let $\mathcal{G} \subseteq \{1,\dots,p\}$ be a set of groups, and $(\eta_g)_{g\in\mathcal{G}}$ be positive weights. The canonical graph $G = (V, E, s, t)$ is the unique graph defined as follows:

1. *$V = V_u \cup V_{gr}$, where V_u is a vertex set of size p, one vertex being associated to each index j in $\{1,\dots,p\}$, and V_{gr} is a vertex set of size $|\mathcal{G}|$, one vertex per group g in $\mathcal{G}$. We thus have $|V| = |\mathcal{G}| + p$. For simplicity, we identify groups g in $\mathcal{G}$ and indices j in $\{1,\dots,p\}$ with vertices of the graph, such that one can from now on refer to "vertex j" or "vertex g".*

2. *For every group g in $\mathcal{G}$, E contains an arc (s, g). These arcs have capacity $\lambda\eta_g$ and zero cost.*

3. *For every group g in $\mathcal{G}$, and every index j in g, E contains an arc (g, j) with zero cost and infinite capacity. We denote by ξ^g_j the flow on this arc.*

4. *For every index j in $\{1,\dots,p\}$, E contains an arc (j,t) with infinite capacity and a cost $\frac{1}{2}(\mathbf{u}_j - \overline{\xi}_j)^2$, where $\overline{\xi}_j$ is the flow on (j,t).*

Examples of canonical graphs are given in Figures 1a-c for three simple group structures. The flows ξ^g_j associated with G can now be identified with the variables of problem (4). Since we have assumed the entries of $\mathbf{u}$ to be non-negative, we can now reformulate Equation (4) as

$$\min_{\xi\in\mathbb{R}_+^{p\times|\mathcal{G}|},\,\overline{\xi}\in\mathbb{R}^p} \sum_{j=1}^{p} \frac{1}{2}(\mathbf{u}_j - \overline{\xi}_j)^2 \;\text{ s.t. }\; \overline{\xi} = \sum_{g\in\mathcal{G}} \xi^g \;\text{ and }\; \forall g \in \mathcal{G},\; \left\{ \sum_{j\in g} \xi^g_j \leq \lambda\eta_g \;\text{ and }\; \mathrm{Supp}(\xi^g) \subseteq g \right\}. \tag{5}$$

Indeed,

- the only arcs with a cost are those leading to the sink, which have the form (j,t), where j is the index of a variable in $\{1,\dots,p\}$. The sum of these costs is $\sum_{j=1}^{p} \frac{1}{2}(\mathbf{u}_j - \overline{\xi}_j)^2$, which is the objective function minimized in Equation (5);
- by flow conservation, we necessarily have $\overline{\xi}_j = \sum_{g\in\mathcal{G}} \xi^g_j$ in the canonical graph;
- the only arcs with a capacity constraints are those coming out of the source, which have the form (s, g), where g is a group in $\mathcal{G}$. By flow conservation, the flow on an arc (s, g) is $\sum_{j\in g} \xi^g_j$ which should be less than $\lambda\eta_g$ by capacity constraints;
- all other arcs have the form (g, j), where g is in $\mathcal{G}$ and j is in g. Thus, $\mathrm{Supp}(\xi^g) \subseteq g$.

Therefore we have shown that finding a flow *minimizing the sum of the costs* on such a graph is equivalent to solving problem (4). When some groups are included in others, the canonical graph can be simplified to yield a graph with a smaller number of edges. Specifically, if h and g are groups with $h \subset g$, the edges (g, j) for $j \in h$ carrying a flow ξ_j^g can be removed and replaced by a single edge (g,h) of infinite capacity and zero cost, carrying the flow $\sum_{j \in h} \xi_j^g$. This simplification is illustrated in Figure 1d, with a graph equivalent to the one of Figure 1c. This does not change the optimal value of $\bar{\xi}^\star$, which is the quantity of interest for computing the optimal primal variable $\mathbf{w}^\star$. We present in Appendix A a formal definition of equivalent graphs. These simplifications are useful in practice, since they reduce the number of edges in the graph and improve the speed of our algorithms.

3.4 Computation of the Proximal Operator

Quadratic min-cost flow problems have been well studied in the operations research literature (Hochbaum and Hong, 1995). One of the simplest cases, where $\mathcal{G}$ contains a single group as in Figure 1a, is solved by an orthogonal projection on the ℓ_1-ball of radius $\lambda\eta_g$. It has been shown, both in machine learning (Duchi et al., 2008) and operations research (Hochbaum and Hong, 1995; Brucker, 1984), that such a projection can be computed in $O(p)$ operations. When the group structure is a tree as in Figure 1d, strategies developed in the two communities are also similar (Jenatton et al., 2010a; Hochbaum and Hong, 1995),[8] and solve the problem in $O(pd)$ operations, where d is the depth of the tree.

The general case of overlapping groups is more difficult. Hochbaum and Hong (1995) have shown that *quadratic min-cost flow problems* can be reduced to a specific *parametric max-flow* problem, for which an efficient algorithm exists (Gallo et al., 1989).[9] While this generic approach could be used to solve Equation (4), we propose to use Algorithm 1 that also exploits the fact that our graphs have non-zero costs only on edges leading to the sink. As shown in Appendix D, it it has a significantly better performance in practice. This algorithm clearly shares some similarities with existing approaches in network flow optimization such as the simplified version of Gallo et al. (1989) presented by Babenko and Goldberg (2006) that uses a divide and conquer strategy. Moreover, an equivalent algorithm exists for minimizing convex functions over polymatroid sets (Groenevelt, 1991). This equivalence, a priori non trivial, is uncovered through a representation of structured sparsity-inducing norms via submodular functions, which was recently proposed by Bach (2010).

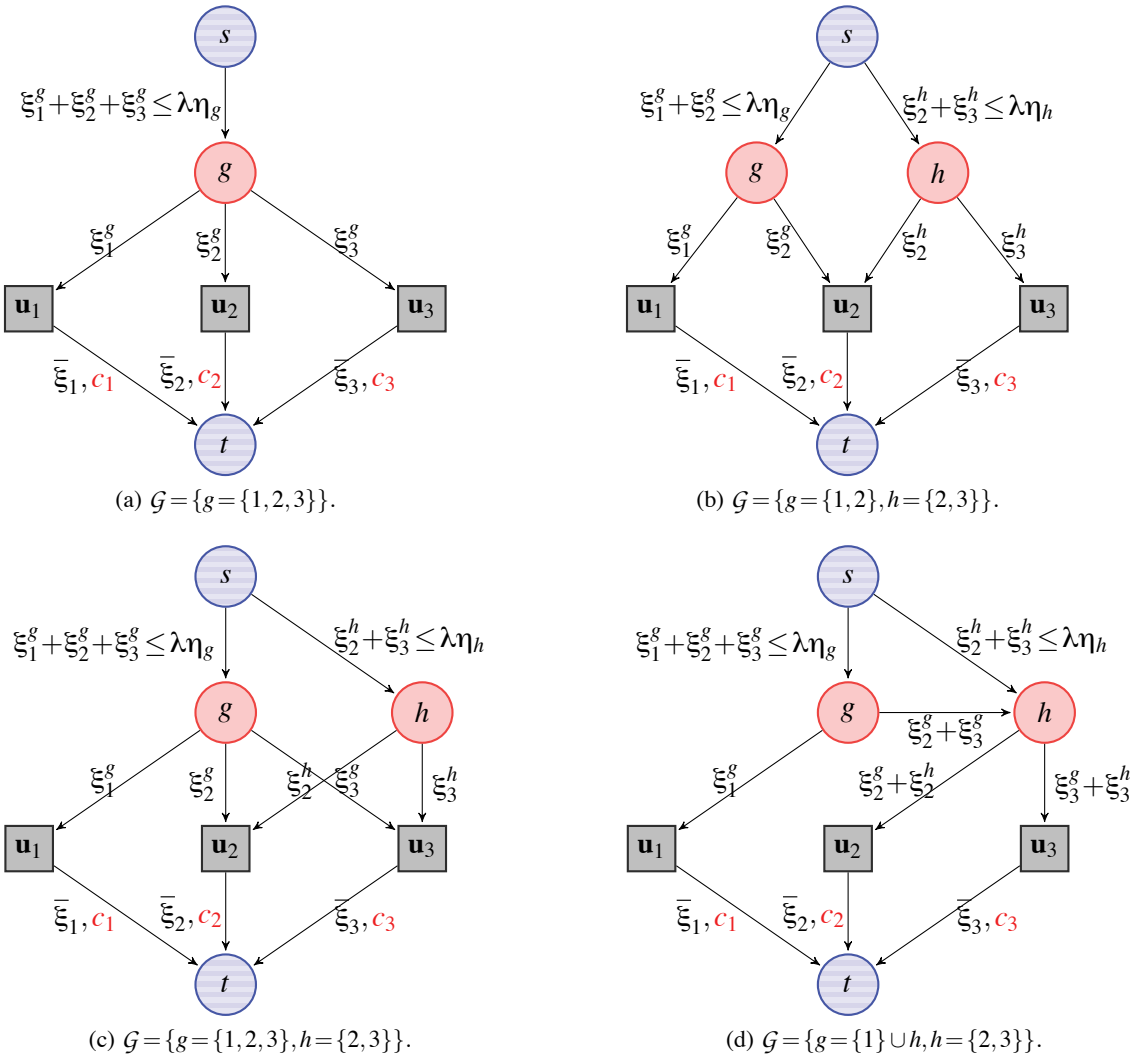
The intuition behind our algorithm, `computeFlow` (see Algorithm 1), is the following: since $\bar{\xi} = \sum_{g \in \mathcal{G}} \xi^g$ is the only value of interest to compute the solution of the proximal operator $\mathbf{w} = \mathbf{u} - \bar{\xi}$, the first step looks for a candidate value γ for $\bar{\xi}$ by solving the following relaxed version of problem (5):

$$\arg\min_{\gamma \in \mathbb{R}^p} \sum_{j \in V_u} \frac{1}{2}(\mathbf{u}_j - \gamma_j)^2 \;\; \text{s.t.} \;\; \sum_{j \in V_u} \gamma_j \leq \lambda \sum_{g \in V_{gr}} \eta_g. \tag{6}$$

The cost function here is the same as in problem (5), but the constraints are weaker: Any feasible point of problem (5) is also feasible for problem (6). This problem can be solved in linear time (Brucker, 1984). Its solution, which we denote γ for simplicity, provides the lower bound $\|\mathbf{u} - \gamma\|_2^2/2$ for the optimal cost of problem (5).

8. Note however that, while Hochbaum and Hong (1995) only consider a tree-structured sum of ℓ_∞-norms, the results from Jenatton et al. (2010a) also apply for a sum of ℓ_2-norms.

9. By definition, a parametric max-flow problem consists in solving, for every value of a parameter, a max-flow problem on a graph whose arc capacities depend on this parameter.



(a) $\mathcal{G}=\{g=\{1,2,3\}\}$.

(b) $\mathcal{G}=\{g=\{1,2\},h=\{2,3\}\}$.

(c) $\mathcal{G}=\{g=\{1,2,3\},h=\{2,3\}\}$.

(d) $\mathcal{G}=\{g=\{1\}\cup h,h=\{2,3\}\}$.

Figure 1: Graph representation of simple proximal problems with different group structures $\mathcal{G}$. The three indices $1,2,3$ are represented as grey squares, and the groups g,h in $\mathcal{G}$ as red (darker) discs. The source is linked to every group g,h with respective maximum capacity $\lambda\eta_g,\lambda\eta_h$ and zero cost. Each variable $\mathbf{u}_j$ is linked to the sink t, with an infinite capacity, and with a cost $c_j \triangleq \frac{1}{2}(\mathbf{u}_j-\overline{\xi}_j)^2$. All other arcs in the graph have zero cost and infinite capacity. They represent inclusion relations in-between groups, and between groups and variables. The graphs (c) and (d) correspond to a special case of tree-structured hierarchy in the sense of Jenatton et al. (2010a). Their min-cost flow problems are equivalent.

Algorithm 1 Computation of the proximal operator for overlapping groups.

input $\mathbf{u} \in \mathbb{R}^p$, a set of groups $\mathcal{G}$, positive weights $(\eta_g)_{g\in\mathcal{G}}$, and λ (regularization parameter).

1: Build the initial graph $G_0 = (V_0, E_0, s, t)$ as explained in Section 3.4.
2: Compute the optimal flow: $\overline{\xi} \leftarrow \texttt{computeFlow}(V_0, E_0)$.
3: **Return:** $\mathbf{w} = \mathbf{u} - \overline{\xi}$ (optimal solution of the proximal problem).

Function `computeFlow`$(V = V_u \cup V_{gr}, E)$

1: Projection step: $\gamma \leftarrow \arg\min_\gamma \sum_{j\in V_u} \frac{1}{2}(\mathbf{u}_j - \gamma_j)^2$ s.t. $\sum_{j\in V_u} \gamma_j \leq \lambda \sum_{g\in V_{gr}} \eta_g$.
2: For all nodes j in V_u, set γ_j to be the capacity of the arc (j,t).
3: Max-flow step: Update $(\overline{\xi}_j)_{j\in V_u}$ by computing a max-flow on the graph (V, E, s, t).
4: **if** $\exists\, j \in V_u$ s.t. $\overline{\xi}_j \neq \gamma_j$ **then**
5: Denote by (s, V^+) and (V^-, t) the two disjoint subsets of (V, s, t) separated by the minimum (s,t)-cut of the graph, and remove the arcs between V^+ and V^-. Call E^+ and E^- the two remaining disjoint subsets of E corresponding to V^+ and V^-.
6: $(\overline{\xi}_j)_{j\in V_u^+} \leftarrow \texttt{computeFlow}(V^+, E^+)$.
7: $(\overline{\xi}_j)_{j\in V_u^-} \leftarrow \texttt{computeFlow}(V^-, E^-)$.
8: **end if**
9: **Return:** $(\overline{\xi}_j)_{j\in V_u}$.

The second step tries to construct a feasible flow $(\xi, \overline{\xi})$, satisfying additional capacity constraints equal to γ_j on arc (j,t), and whose cost matches this lower bound; this latter problem can be cast as a max-flow problem (Goldberg and Tarjan, 1986). If such a flow exists, the algorithm returns $\overline{\xi} = \gamma$, the cost of the flow reaches the lower bound, and is therefore optimal. If such a flow does not exist, we have $\overline{\xi} \neq \gamma$, the lower bound is not achievable, and we build a minimum (s,t)-cut of the graph (Ford and Fulkerson, 1956) defining two disjoints sets of nodes V^+ and V^-; V^+ is the part of the graph which is reachable from the source (for every node j in $\mathbf{V}^+$, there exists a non-saturated path from s to j), whereas all paths going from s to nodes in V^- are saturated. More details about these properties can be found at the beginning of Appendix B. At this point, it is possible to show that the value of the optimal min-cost flow on all arcs between V^+ and V^- is necessary zero. Thus, removing them yields an equivalent optimization problem, which can be decomposed into two independent problems of smaller sizes and solved recursively by the calls to `computeFlow`(V^+, E^+) and `computeFlow`(V^-, E^-). A formal proof of correctness of Algorithm 1 and further details are relegated to Appendix B.

The approach of Hochbaum and Hong (1995); Gallo et al. (1989) which recasts the quadratic min-cost flow problem as a parametric max-flow is guaranteed to have the same worst-case complexity as a single max-flow algorithm. However, we have experimentally observed a significant discrepancy between the worst case and empirical complexities for these flow problems, essentially because the empirical cost of each max-flow is significantly smaller than its theoretical cost. Despite the fact that the worst-case guarantees for our algorithm is weaker than theirs (up to a factor $|V|$), it is more adapted to the structure of our graphs and has proven to be much faster in our experiments (see Appendix D).[10] Some implementation details are also crucial to the efficiency of the algorithm:

10. The best theoretical worst-case complexity of a max-flow is achieved by Goldberg and Tarjan (1986) and is $O\big(|V||E|\log(|V|^2/|E|)\big)$. Our algorithm achieves the same worst-case complexity when the cuts are well balanced—

- **Exploiting maximal connected components**: When there exists no arc between two subsets of V, the solution can be obtained by solving two smaller optimization problems corresponding to the two disjoint subgraphs. It is indeed possible to process them independently to solve the global min-cost flow problem. To that effect, before calling the function `computeFlow`(V,E), we look for maximal connected components $(V_1,E_1),\dots,(V_N,E_N)$ and call sequentially the procedure `computeFlow`(V_i,E_i) for i in $\{1,\dots,N\}$.

- **Efficient max-flow algorithm**: We have implemented the "push-relabel" algorithm of Goldberg and Tarjan (1986) to solve our max-flow problems, using classical heuristics that significantly speed it up in practice; see Goldberg and Tarjan (1986) and Cherkassky and Goldberg (1997). We use the so-called "highest-active vertex selection rule, global and gap heuristics" (Goldberg and Tarjan, 1986; Cherkassky and Goldberg, 1997), which has a worst-case complexity of $O(|V|^2|E|^{1/2})$ for a graph (V,E,s,t). This algorithm leverages the concept of *pre-flow* that relaxes the definition of flow and allows vertices to have a positive excess.

- **Using flow warm-restarts**: The max-flow steps in our algorithm can be initialized with any valid pre-flow, enabling warm-restarts. This is also a key concept in the parametric max-flow algorithm of Gallo et al. (1989).

- **Improved projection step**: The first line of the procedure `computeFlow` can be replaced by $\gamma \leftarrow \arg\min_\gamma \sum_{j\in V_u} \frac{1}{2}(\mathbf{u}_j-\gamma_j)^2$ s.t. $\sum_{j\in V_u}\gamma_j \leq \lambda\sum_{g\in V_{gr}}\eta_g$ and $|\gamma_j| \leq \lambda\sum_{g\ni j}\eta_g$. The idea is to build a relaxation of Equation (5) which is closer to the original problem than the one of Equation (6), but that still can be solved in linear time. The structure of the graph will indeed not allow $\bar{\xi}_j$ to be greater than $\lambda\sum_{g\ni j}\eta_g$ after the max-flow step. This modified projection step can still be computed in linear time (Brucker, 1984), and leads to better performance.

3.5 Computation of the Dual Norm

The dual norm Ω^* of Ω, defined for any vector κ in $\mathbb{R}^p$ by

$$\Omega^*(\kappa) \triangleq \max_{\Omega(\mathbf{z})\leq 1} \mathbf{z}^\top\kappa,$$

is a key quantity to study sparsity-inducing regularizations in many respects. For instance, dual norms are central in working-set algorithms (Jenatton et al., 2009; Bach et al., 2011), and arise as well when proving theoretical estimation or prediction guarantees (Negahban et al., 2009).

In our context, we use it to monitor the convergence of the proximal method through a duality gap, hence defining a proper optimality criterion for problem (1). As a brief reminder, the duality gap of a minimization problem is defined as the difference between the primal and dual objective functions, evaluated for a feasible pair of primal/dual variables (see Section 5.5, Boyd and Vandenberghe, 2004). This gap serves as a certificate of (sub)optimality: if it is equal to zero, then the optimum is reached, and provided that strong duality holds, the converse is true as well (see Section 5.5, Boyd and Vandenberghe, 2004). A description of the algorithm we use in the experiments (Beck and Teboulle, 2009) along with the integration of the computation of the duality gap is given in Appendix C.

that is $|V^+| \approx |V^-| \approx |V|/2$, but we lose a factor $|V|$ when it is not the case. The practical speed of such algorithms is however significantly different than their theoretical worst-case complexities (see Boykov and Kolmogorov, 2004).

We now denote by f^* the Fenchel conjugate of f (Borwein and Lewis, 2006), defined by $f^*(\kappa) \triangleq \sup_{\mathbf{z}}[\mathbf{z}^\top \kappa - f(\mathbf{z})]$. The duality gap for problem (1) can be derived from standard Fenchel duality arguments (Borwein and Lewis, 2006) and it is equal to

$$f(\mathbf{w}) + \lambda\Omega(\mathbf{w}) + f^*(-\kappa) \text{ for } \mathbf{w}, \kappa \text{ in } \mathbb{R}^p \text{ with } \Omega^*(\kappa) \leq \lambda.$$

Therefore, evaluating the duality gap requires to compute efficiently Ω^* in order to find a feasible dual variable κ (the gap is otherwise equal to $+\infty$ and becomes non-informative). This is equivalent to solving another network flow problem, based on the following variational formulation:

$$\Omega^*(\kappa) = \min_{\xi \in \mathbb{R}^{p\times|\mathcal{G}|}} \tau \quad \text{s.t.} \quad \sum_{g\in\mathcal{G}} \xi^g = \kappa, \text{ and } \forall g \in \mathcal{G},\ \|\xi^g\|_1 \leq \tau\eta_g \text{ with } \xi^g_j = 0 \text{ if } j \notin g. \tag{7}$$

In the network problem associated with (7), the capacities on the arcs (s,g), $g \in \mathcal{G}$, are set to $\tau\eta_g$, and the capacities on the arcs (j,t), j in $\{1,\dots,p\}$, are fixed to κ_j. Solving problem (7) amounts to finding the smallest value of τ, such that there exists a flow saturating all the capacities κ_j on the arcs leading to the sink t. Equation (7) and Algorithm 2 are proven to be correct in Appendix B.

Algorithm 2 Computation of the dual norm.

input $\kappa \in \mathbb{R}^p$, a set of groups $\mathcal{G}$, positive weights $(\eta_g)_{g\in\mathcal{G}}$.

1: Build the initial graph $G_0 = (V_0, E_0, s, t)$ as explained in Section 3.5.
2: $\tau \leftarrow \texttt{dualNorm}(V_0, E_0)$.
3: **Return:** τ (value of the dual norm).

Function `dualNorm`$(V = V_u \cup V_{gr}, E)$

1: $\tau \leftarrow (\sum_{j\in V_u} \kappa_j)/(\sum_{g\in V_{gr}} \eta_g)$ and set the capacities of arcs (s,g) to $\tau\eta_g$ for all g in V_{gr}.
2: Max-flow step: Update $(\bar{\xi}_j)_{j\in V_u}$ by computing a max-flow on the graph (V,E,s,t).
3: **if** $\exists\ j \in V_u$ s.t. $\bar{\xi}_j \neq \kappa_j$ **then**
4: Define (V^+,E^+) and (V^-,E^-) as in Algorithm 1, and set $\tau \leftarrow \texttt{dualNorm}(V^-,E^-)$.
5: **end if**
6: **Return:** τ.

4. Optimization with Proximal Splitting Methods

We now present proximal splitting algorithms (see Combettes and Pesquet, 2008, 2010; Tomioka et al., 2011; Boyd et al., 2011, and references therein) for solving Equation (1). Differentiability of f is not required here and the regularization function can either be a sum of ℓ_2- or ℓ_∞-norms. However, we assume that:

(A) either f can be written $f(\mathbf{w}) = \sum_{i=1}^n \tilde{f}_i(\mathbf{w})$, where the functions $\tilde{f}_i$ are such that $\text{prox}_{\gamma\tilde{f}_i}$ can be obtained in closed form for all $\gamma > 0$ and all i—that is, for all $\mathbf{u}$ in $\mathbb{R}^m$, the following problems admit closed form solutions: $\min_{\mathbf{v}\in\mathbb{R}^m} \frac{1}{2}\|\mathbf{u}-\mathbf{v}\|_2^2 + \gamma\tilde{f}_i(\mathbf{v})$.

(B) or f can be written $f(\mathbf{w}) = \tilde{f}(\mathbf{X}\mathbf{w})$ for all $\mathbf{w}$ in $\mathbb{R}^p$, where $\mathbf{X}$ in $\mathbb{R}^{n\times p}$ is a design matrix, and one knows how to efficiently compute $\text{prox}_{\gamma\tilde{f}}$ for all $\gamma > 0$.

It is easy to show that this condition is satisfied for the square and hinge loss functions, making it possible to build linear SVMs with a structured sparse regularization. These assumptions are not the same as the ones of Section 3, and the scope of the problems addressed is therefore slightly different. Proximal splitting methods seem indeed to offer more flexibility regarding the regularization function, since they can deal with sums of ℓ_2-norms.[11] However, proximal gradient methods, as presented in Section 3, enjoy a few advantages over proximal splitting methods, namely: automatic parameter tuning with line-search schemes (Nesterov, 2007), known convergence rates (Nesterov, 2007; Beck and Teboulle, 2009), and ability to provide sparse solutions (approximate solutions obtained with proximal splitting methods often have small values, but not "true" zeros).

4.1 Algorithms

We consider a class of algorithms which leverage the concept of variable splitting (see Combettes and Pesquet, 2010; Bertsekas and Tsitsiklis, 1989; Tomioka et al., 2011). The key is to introduce additional variables $\mathbf{z}^g$ in $\mathbb{R}^{|g|}$, one for every group g in $\mathcal{G}$, and equivalently reformulate Equation (1) as

$$\min_{\substack{\mathbf{w}\in\mathbb{R}^p \\ \mathbf{z}^g\in\mathbb{R}^{|g|} \text{ for } g\in\mathcal{G}}} f(\mathbf{w}) + \lambda \sum_{g\in\mathcal{G}} \eta_g \|\mathbf{z}^g\| \text{ s.t. } \forall g \in \mathcal{G},\ \mathbf{z}^g = \mathbf{w}_g, \tag{8}$$

The issue of overlapping groups is removed, but new constraints are added, and as in Section 3, the method introduces additional variables which induce a memory cost of $O(\sum_{g\in\mathcal{G}} |g|)$.

To solve this problem, it is possible to use the so-called alternating direction method of multipliers (ADMM) (see Combettes and Pesquet, 2010; Bertsekas and Tsitsiklis, 1989; Tomioka et al., 2011; Boyd et al., 2011).[12] It introduces dual variables $\boldsymbol{\nu}^g$ in $\mathbb{R}^{|g|}$ for all g in $\mathcal{G}$, and defines the augmented Lagrangian:

$$\mathcal{L}\big(\mathbf{w}, (\mathbf{z}^g)_{g\in\mathcal{G}}, (\boldsymbol{\nu}^g)_{g\in\mathcal{G}}\big) \triangleq f(\mathbf{w}) + \sum_{g\in\mathcal{G}} \big[\lambda\eta_g\|\mathbf{z}^g\| + \boldsymbol{\nu}^{g\top}(\mathbf{z}^g - \mathbf{w}_g) + \frac{\gamma}{2}\|\mathbf{z}^g - \mathbf{w}_g\|_2^2\big],$$

where $\gamma > 0$ is a parameter. It is easy to show that solving Equation (8) amounts to finding a saddle-point of the augmented Lagrangian.[13] The ADMM algorithm finds such a saddle-point by iterating between the minimization of $\mathcal{L}$ with respect to each primal variable, keeping the other ones fixed, and gradient ascent steps with respect to the dual variables. More precisely, it can be summarized as:

1. Minimize $\mathcal{L}$ with respect to $\mathbf{w}$, keeping the other variables fixed.

11. We are not aware of any efficient algorithm providing the exact solution of the proximal operator associated to a sum of ℓ_2-norms, which would be necessary for using (accelerated) proximal gradient methods. An iterative algorithm could possibly be used to compute it approximately (e.g., see Jenatton et al., 2010a, 2011), but such a procedure would be computationally expensive and would require to be able to deal with approximate computations of the proximal operators (e.g., see Combettes and Pesquet, 2010; Schmidt et al., 2011, and discussions therein). We have chosen not to consider this possibility in this paper.
12. This method is used by Sprechmann et al. (2010) for computing the proximal operator associated to hierarchical norms, and independently in the same context as ours by Boyd et al. (2011) and Qin and Goldfarb (2011).
13. The augmented Lagrangian is in fact the classical Lagrangian (see Boyd and Vandenberghe, 2004) of the following optimization problem which is equivalent to Equation (8):

$$\min_{\mathbf{w}\in\mathbb{R}^p, (\mathbf{z}^g\in\mathbb{R}^{|g|})_{g\in\mathcal{G}}} f(\mathbf{w}) + \lambda \sum_{g\in\mathcal{G}} \eta_g\|\mathbf{z}^g\| + \frac{\gamma}{2}\|\mathbf{z}^g - \mathbf{w}_g\|_2^2 \text{ s.t. } \forall g \in \mathcal{G},\ \mathbf{z}^g = \mathbf{w}_g.$$

2. Minimize $\mathcal{L}$ with respect to the $\mathbf{z}^g$'s, keeping the other variables fixed. The solution can be obtained in closed form: for all g in $\mathcal{G}$, $\mathbf{z}^g \leftarrow \text{prox}_{\frac{\lambda\eta_g}{\gamma}\|.\|}[\mathbf{w}_g - \frac{1}{\gamma}\nu^g]$.

3. Take a gradient ascent step on $\mathcal{L}$ with respect to the ν^g's: $\nu^g \leftarrow \nu^g + \gamma(\mathbf{z}^g - \mathbf{w}_g)$.

4. Go back to step 1.

Such a procedure is guaranteed to converge to the desired solution for all value of $\gamma > 0$ (however, tuning γ can greatly influence the convergence speed), but solving efficiently step 1 can be difficult. To cope with this issue, we propose two variations exploiting assumptions **(A)** and **(B)**.

4.1.1 SPLITTING THE LOSS FUNCTION f

We assume condition **(A)**—that is, we have $f(\mathbf{w}) = \sum_{i=1}^n \tilde{f}_i(\mathbf{w})$. For example, when f is the square loss function $f(\mathbf{w}) = \frac{1}{2}\|\mathbf{y} - \mathbf{X}\mathbf{w}\|_2^2$, where $\mathbf{X}$ in $\mathbb{R}^{n\times p}$ is a design matrix and $\mathbf{y}$ is in $\mathbb{R}^n$, we would define for all i in $\{1,\ldots,n\}$ the functions $\tilde{f}_i : \mathbb{R} \to \mathbb{R}$ such that $\tilde{f}_i(\mathbf{w}) \triangleq \frac{1}{2}(\mathbf{y}_i - \mathbf{x}_i^\top \mathbf{w})^2$, where $\mathbf{x}_i$ is the i-th row of $\mathbf{X}$.

We now introduce new variables $\mathbf{v}^i$ in $\mathbb{R}^p$ for $i = 1,\ldots,n$, and replace $f(\mathbf{w})$ in Equation (8) by $\sum_{i=1}^n \tilde{f}_i(\mathbf{v}^i)$, with the additional constraints that $\mathbf{v}^i = \mathbf{w}$. The resulting equivalent optimization problem can now be tackled using the ADMM algorithm, following the same methodology presented above. It is easy to show that every step can be obtained efficiently, as long as one knows how to compute the proximal operator associated to the functions $\tilde{f}_i$ in closed form. This is in fact the case for the square and hinge loss functions, where n is the number of training points. The main problem of this strategy is the possible high memory usage it requires when n is large.

4.1.2 DEALING WITH THE DESIGN MATRIX

If we assume condition **(B)**, another possibility consists of introducing a new variable $\mathbf{v}$ in $\mathbb{R}^n$, such that one can replace the function $f(\mathbf{w}) = \tilde{f}(\mathbf{X}\mathbf{w})$ by $\tilde{f}(\mathbf{v})$ in Equation (8) with the additional constraint $\mathbf{v} = \mathbf{X}\mathbf{w}$. Using directly the ADMM algorithm to solve the corresponding problem implies adding a term $\kappa^\top(\mathbf{v} - \mathbf{X}\mathbf{w}) + \frac{\gamma}{2}\|\mathbf{v} - \mathbf{X}\mathbf{w}\|_2^2$ to the augmented Lagrangian $\mathcal{L}$, where κ is a new dual variable. The minimization of $\mathcal{L}$ with respect to $\mathbf{v}$ is now obtained by $\mathbf{v} \leftarrow \text{prox}_{\frac{1}{\gamma}\tilde{f}}[\mathbf{X}\mathbf{w} - \kappa]$, which is easy to compute according to **(B)**. However, the design matrix $\mathbf{X}$ in the quadratic term makes the minimization of $\mathcal{L}$ with respect to $\mathbf{w}$ more difficult. To overcome this issue, we adopt a strategy presented by Zhang et al. (2011), which replaces at iteration k the quadratic term $\frac{\gamma}{2}\|\mathbf{v} - \mathbf{X}\mathbf{w}\|_2^2$ in the augmented Lagrangian by an additional proximity term: $\frac{\gamma}{2}\|\mathbf{v} - \mathbf{X}\mathbf{w}\|_2^2 + \frac{\gamma}{2}\|\mathbf{w} - \mathbf{w}^k\|_\mathbf{Q}^2$, where $\mathbf{w}^k$ is the current estimate of $\mathbf{w}$, and $\|\mathbf{w} - \mathbf{w}^k\|_\mathbf{Q}^2 = (\mathbf{w} - \mathbf{w}^k)^\top \mathbf{Q}(\mathbf{w} - \mathbf{w}^k)$, where $\mathbf{Q}$ is a symmetric positive definite matrix. By choosing $\mathbf{Q} \triangleq \delta\mathbf{I} - \mathbf{X}^\top\mathbf{X}$, with δ large enough, minimizing $\mathcal{L}$ with respect to $\mathbf{w}$ becomes simple, while convergence to the solution is still ensured. More details can be found in Zhang et al. (2011).

5. Applications and Experiments

In this section, we present various experiments demonstrating the applicability and the benefits of our methods for solving large-scale sparse and structured regularized problems.

5.1 Speed Benchmark

We consider a structured sparse decomposition problem with overlapping groups of ℓ_∞-norms, and compare the proximal gradient algorithm FISTA (Beck and Teboulle, 2009) with our proximal operator presented in Section 3 (referred to as ProxFlow), two variants of proximal splitting methods, (ADMM) and (Lin-ADMM) respectively presented in Section 4.1.1 and 4.1.2, and two generic optimization techniques, namely a subgradient descent (SG) and an interior point method,[14] on a regularized linear regression problem. SG, ProxFlow, ADMM and Lin-ADMM are implemented in `C++`.[15] Experiments are run on a single-core 2.8 GHz CPU. We consider a design matrix $\mathbf{X}$ in $\mathbb{R}^{n\times p}$ built from overcomplete dictionaries of discrete cosine transforms (DCT), which are naturally organized on one- or two-dimensional grids and display local correlations. The following families of groups $\mathcal{G}$ using this spatial information are thus considered: (1) every contiguous sequence of length 3 for the one-dimensional case, and (2) every 3×3-square in the two-dimensional setting. We generate vectors $\mathbf{y}$ in $\mathbb{R}^n$ according to the linear model $\mathbf{y} = \mathbf{X}\mathbf{w}_0 + \varepsilon$, where $\varepsilon \sim \mathcal{N}(0, 0.01\|\mathbf{X}\mathbf{w}_0\|_2^2)$. The vector $\mathbf{w}_0$ has about 20% percent nonzero components, randomly selected, while respecting the structure of $\mathcal{G}$, and uniformly generated in $[-1,1]$.

In our experiments, the regularization parameter λ is chosen to achieve the same level of sparsity (20%). For SG, ADMM and Lin-ADMM, some parameters are optimized to provide the lowest value of the objective function after 1000 iterations of the respective algorithms. For SG, we take the step size to be equal to $a/(k+b)$, where k is the iteration number, and (a,b) are the pair of parameters selected in $\{10^{-3},\dots,10\}\times\{10^2,10^3,10^4\}$. Note that a step size of the form $a/(\sqrt{t}+b)$ is also commonly used in subgradient descent algorithms. In the context of hierarchical norms, both choices have led to similar results (Jenatton et al., 2011). The parameter γ for ADMM is selected in $\{10^{-2},\dots,10^2\}$. The parameters (γ,δ) for Lin-ADMM are selected in $\{10^{-2},\dots,10^2\}\times\{10^{-1},\dots,10^8\}$. For interior point methods, since problem (1) can be cast either as a quadratic (QP) or as a conic program (CP), we show in Figure 2 the results for both formulations. On three problems of different sizes, with $(n,p)\in\{(100,10^3),(1024,10^4),(1024,10^5)\}$, our algorithms ProxFlow, ADMM and Lin-ADMM compare favorably with the other methods, (see Figure 2), except for ADMM in the large-scale setting which yields an objective function value similar to that of SG after 10^4 seconds. Among ProxFlow, ADMM and Lin-ADMM, ProxFlow is consistently better than Lin-ADMM, which is itself better than ADMM. Note that for the small scale problem, the performance of ProxFlow and Lin-ADMM is similar. In addition, note that QP, CP, SG, ADMM and Lin-ADMM do not obtain sparse solutions, whereas ProxFlow does.[16]

5.2 Wavelet Denoising with Structured Sparsity

We now illustrate the results of Section 3, where a single large-scale proximal operator ($p\approx 250\,000$) associated to a sum of ℓ_∞-norms has to be computed. We choose an image denoising task with an orthonormal wavelet basis, following an experiment similar to one proposed in Jenatton et al. (2011). Specifically, we consider the following formulation

$$\min_{\mathbf{w}\in\mathbb{R}^p} \frac{1}{2}\|\mathbf{y}-\mathbf{X}\mathbf{w}\|_2^2 + \lambda\Omega(\mathbf{w}),$$

14. In our simulations, we use the commercial software `Mosek`, `http://www.mosek.com/`
15. Our implementation of ProxFlow is available at `http://www.di.ens.fr/willow/SPAMS/`.
16. To reduce the computational cost of this experiment, the curves reported are the results of one single run. Similar types of experiments with several runs have shown very small variability (Bach et al., 2011).

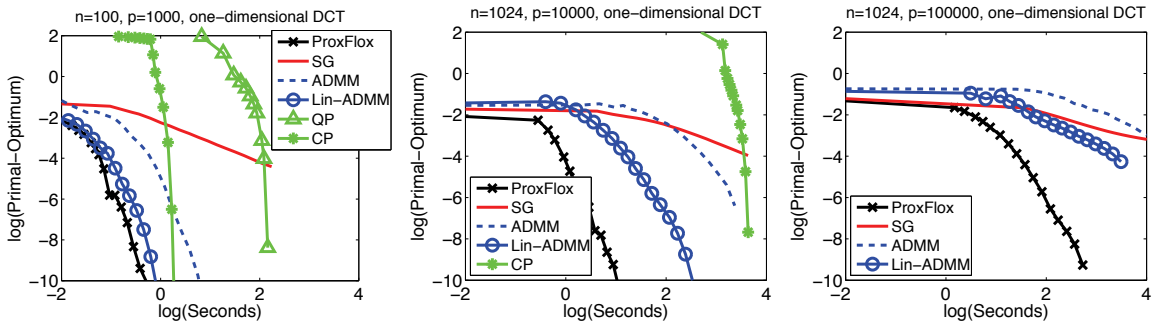


Figure 2: Speed comparisons: distance to the optimal primal value versus CPU time (log-log scale). Due to the computational burden, QP and CP could not be run on every problem.

where $\mathbf{y}$ in $\mathbb{R}^p$ is a noisy input image, $\mathbf{w}$ represents wavelets coefficients, $\mathbf{X}$ in $\mathbb{R}^{p\times p}$ is an orthonormal wavelet basis, $\mathbf{Xw}$ is the estimate of the denoised image, and Ω is a sparsity-inducing norm. Since here the basis is orthonormal, solving the decomposition problem boils down to computing $\mathbf{w}^\star = \text{prox}_{\lambda\Omega}[\mathbf{X}^\top\mathbf{y}]$. This makes of Algorithm 1 a good candidate to solve it when Ω is a sum of ℓ_∞-norms. We compare the following candidates for the sparsity-inducing norms Ω:

- the ℓ_1-norm, leading to the wavelet soft-thresholding of Donoho and Johnstone (1995).
- a sum of ℓ_∞-norms with a hierarchical group structure adapted to the wavelet coefficients, as proposed in Jenatton et al. (2011). Considering a natural quad-tree for wavelet coefficients (see Mallat, 1999), this norm takes the form of Equation (2) with one group per wavelet coefficient that contains the coefficient and all its descendants in the tree. We call this norm Ω_{tree}.
- a sum of ℓ_∞-norms with overlapping groups representing 2×2 spatial neighborhoods in the wavelet domain. This regularization encourages neighboring wavelet coefficients to be set to zero together, which was also exploited in the past in block-thresholding approaches for wavelet denoising (Cai, 1999). We call this norm Ω_{grid}.

We consider Daubechies3 wavelets (see Mallat, 1999) for the matrix $\mathbf{X}$, use 12 classical standard test images,[17] and generate noisy versions of them corrupted by a white Gaussian noise of variance σ^2. For each image, we test several values of $\lambda = 2^{\frac{i}{4}}\sigma\sqrt{\log p}$, with i taken in the range $\{-15,-14,\ldots,15\}$. We then keep the parameter λ giving the best reconstruction error on average on the 12 images. The factor $\sigma\sqrt{\log p}$ is a classical heuristic for choosing a reasonable regularization parameter (see Mallat, 1999). We provide reconstruction results in terms of PSNR in Table 1.[18] Unlike Jenatton et al. (2011), who set all the weights η_g in Ω equal to one, we tried exponential weights of the form $\eta_g = \rho^k$, with k being the depth of the group in the wavelet tree, and ρ is taken in $\{0.25, 0.5, 1, 2, 4\}$. As for λ, the value providing the best reconstruction is kept. The wavelet transforms in our experiments are computed with the matlabPyrTools software.[19] Interestingly, we observe in Table 1 that the results obtained with Ω_{grid} are significantly better than those obtained

17. These images are used in classical image denoising benchmarks. See Mairal et al. (2009).
18. Denoting by MSE the mean-squared-error for images whose intensities are between 0 and 255, the PSNR is defined as $\text{PSNR} = 10\log_{10}(255^2/\text{MSE})$ and is measured in dB. A gain of 1dB reduces the MSE by approximately 20%.
19. The matlabPyrTools can be found at `http://www.cns.nyu.edu/$\sim$eero/steerpyr/`.

	PSNR			IPSNR vs. ℓ_1		
σ	ℓ_1	Ω_{tree}	Ω_{grid}	ℓ_1	Ω_{tree}	Ω_{grid}
5	35.67	35.98	**36.15**	$0.00 \pm .0$	$0.31 \pm .18$	$\mathbf{0.48 \pm .25}$
10	31.00	31.60	**31.88**	$0.00 \pm .0$	$0.61 \pm .28$	$\mathbf{0.88 \pm .28}$
25	25.68	26.77	**27.07**	$0.00 \pm .0$	$1.09 \pm .32$	$\mathbf{1.38 \pm .26}$
50	22.37	23.84	**24.06**	$0.00 \pm .0$	$1.47 \pm .34$	$\mathbf{1.68 \pm .41}$
100	19.64	21.49	**21.56**	$0.00 \pm .0$	$1.85 \pm .28$	$\mathbf{1.92 \pm .29}$

Table 1: PSNR measured for the denoising of 12 standard images when the regularization function is the ℓ_1-norm, the tree-structured norm Ω_{tree}, and the structured norm Ω_{grid}, and improvement in PSNR compared to the ℓ_1-norm (IPSNR). Best results for each level of noise and each wavelet type are in bold. The reported values are averaged over 5 runs with different noise realizations.

with Ω_{tree}, meaning that encouraging spatial consistency in wavelet coefficients is more effective than using a hierarchical coding. We also note that our approach is relatively fast, despite the high dimension of the problem. Solving exactly the proximal problem with Ω_{grid} for an image with $p = 512 \times 512 = 262\,144$ pixels (and therefore approximately the same number of groups) takes approximately $\approx 4-6$ seconds on a single core of a 3.07GHz CPU.

5.3 CUR-like Matrix Factorization

In this experiment, we show how our tools can be used to perform the so-called CUR matrix decomposition (Mahoney and Drineas, 2009). It consists of a low-rank approximation of a data matrix $\mathbf{X}$ in $\mathbb{R}^{n\times p}$ in the form of a product of three matrices—that is, $\mathbf{X} \approx \mathbf{CUR}$. The particularity of the CUR decomposition lies in the fact that the matrices $\mathbf{C} \in \mathbb{R}^{n\times c}$ and $\mathbf{R} \in \mathbb{R}^{r\times p}$ are constrained to be respectively a subset of c columns and r rows of the original matrix $\mathbf{X}$. The third matrix $\mathbf{U} \in \mathbb{R}^{c\times r}$ is then given by $\mathbf{C}^+\mathbf{X}\mathbf{R}^+$, where $\mathbf{A}^+$ denotes a Moore-Penrose generalized inverse of the matrix $\mathbf{A}$ (Horn and Johnson, 1990). Such a matrix factorization is particularly appealing when the interpretability of the results matters (Mahoney and Drineas, 2009). For instance, when studying gene-expression data sets, it is easier to gain insight from the selection of actual patients and genes, rather than from linear combinations of them.

In Mahoney and Drineas (2009), CUR decompositions are computed by a sampling procedure based on the singular value decomposition of $\mathbf{X}$. In a recent work, Bien et al. (2010) have shown that *partial* CUR decompositions, that is, the selection of either rows or columns of $\mathbf{X}$, can be obtained by solving a convex program with a group-Lasso penalty. We propose to extend this approach to the simultaneous selection of both rows and columns of $\mathbf{X}$, with the following convex problem:

$$\min_{\mathbf{W}\in\mathbb{R}^{p\times n}} \frac{1}{2}\|\mathbf{X} - \mathbf{X}\mathbf{W}\mathbf{X}\|_{\text{F}}^2 + \lambda_{\text{row}} \sum_{i=1}^{n} \|\mathbf{W}^i\|_\infty + \lambda_{\text{col}} \sum_{j=1}^{p} \|\mathbf{W}_j\|_\infty. \tag{9}$$

In this formulation, the two sparsity-inducing penalties controlled by the parameters λ_{row} and λ_{col} set to zero some entire rows and columns of the solutions of problem (9). Now, let us denote by $\mathbf{W}_{\text{IJ}}$ in $\mathbb{R}^{|\text{I}|\times|\text{J}|}$ the submatrix of $\mathbf{W}$ reduced to its nonzero rows and columns, respectively indexed by $\text{I} \subseteq \{1,\dots,p\}$ and $\text{J} \subseteq \{1,\dots,n\}$. We can then readily identify the three components of the CUR decomposition of $\mathbf{X}$, namely

$$\mathbf{X}\mathbf{W}\mathbf{X} = \mathbf{C}\mathbf{W}_{\text{IJ}}\mathbf{R} \approx \mathbf{X}.$$

Problem (9) has a smooth convex data-fitting term and brings into play a sparsity-inducing norm with overlapping groups of variables (the rows and the columns of $\mathbf{W}$). As a result, it is a particular instance of problem (1) that can therefore be handled with the optimization tools introduced in this paper. We now compare the performance of the sampling procedure from Mahoney and Drineas (2009) with our proposed sparsity-based approach. To this end, we consider the four gene-expression data sets `9_Tumors`, `Brain_Tumors1`, `Leukemia1` and `SRBCT`, with respective dimensions $(n,p) \in \{(60,5727),(90,5921),(72,5328),(83,2309)\}$.[20] In the sequel, the matrix $\mathbf{X}$ is normalized to have unit Frobenius-norm while each of its columns is centered. To begin with, we run our approach[21] over a grid of values for λ_{row} and λ_{col} in order to obtain solutions with different sparsity levels, that is, ranging from $|\mathrm{I}| = p$ and $|\mathrm{J}| = n$ down to $|\mathrm{I}| = |\mathrm{J}| = 0$. For each pair of values $[|\mathrm{I}|,|\mathrm{J}|]$, we then apply the sampling procedure from Mahoney and Drineas (2009). Finally, the variance explained by the CUR decompositions is reported in Figure 3 for both methods. Since the sampling approach involves some randomness, we show the average and standard deviation of the results based on five initializations. The conclusions we can draw from the experiments match the ones already reported in Bien et al. (2010) for the partial CUR decomposition. We can indeed see that both schemes perform similarly. However, our approach has the advantage not to be randomized, which can be less disconcerting in the practical perspective of analyzing a single run of the algorithm. It is finally worth being mentioned that the convex approach we develop here is flexible and can be extended in different ways. For instance, we can imagine to add further low-rank/sparsity constraints on $\mathbf{W}$ thanks to sparsity-promoting convex regularizations.

5.4 Background Subtraction

Following Cehver et al. (2008); Huang et al. (2009), we consider a background subtraction task. Given a sequence of frames from a fixed camera, we try to segment out foreground objects in a new image. If we denote by $\mathbf{y} \in \mathbb{R}^n$ this image composed of n pixels, we model $\mathbf{y}$ as a sparse linear combination of p other images $\mathbf{X} \in \mathbb{R}^{n \times p}$, plus an error term $\mathbf{e}$ in $\mathbb{R}^n$, that is, $\mathbf{y} \approx \mathbf{X}\mathbf{w} + \mathbf{e}$ for some sparse vector $\mathbf{w}$ in $\mathbb{R}^p$. This approach is reminiscent of Wright et al. (2009a) in the context of face recognition, where $\mathbf{e}$ is further made sparse to deal with small occlusions. The term $\mathbf{X}\mathbf{w}$ accounts for *background* parts present in both $\mathbf{y}$ and $\mathbf{X}$, while $\mathbf{e}$ contains specific, or *foreground*, objects in $\mathbf{y}$. The resulting optimization problem is given by

$$\min_{\mathbf{w}\in\mathbb{R}^p, \mathbf{e}\in\mathbb{R}^n} \frac{1}{2}\|\mathbf{y}-\mathbf{X}\mathbf{w}-\mathbf{e}\|_2^2 + \lambda_1\|\mathbf{w}\|_1 + \lambda_2\{\|\mathbf{e}\|_1 + \Omega(\mathbf{e})\}, \text{ with } \lambda_1,\lambda_2 \geq 0. \tag{10}$$

In this formulation, the only ℓ_1-norm penalty does not take into account the fact that neighboring pixels in $\mathbf{y}$ are likely to share the same label (background or foreground), which may lead to scattered pieces of foreground and background regions (Figure 4). We therefore put an additional structured regularization term Ω on $\mathbf{e}$, where the groups in $\mathcal{G}$ are all the overlapping 3×3-squares on the image. For the sake of comparison, we also consider the regularization $\tilde{\Omega}$ where the groups are *non-overlapping* 3×3-squares.

This optimization problem can be viewed as an instance of problem (1), with the particular design matrix $[\mathbf{X},\ \mathbf{I}]$ in $\mathbb{R}^{n\times(p+n)}$, defined as the columnwise concatenation of $\mathbf{X}$ and the identity

20. The data sets are freely available at `http://www.gems-system.org/`.

21. More precisely, since the penalties in problem (9) shrink the coefficients of $\mathbf{W}$, we follow a two-step procedure: We first run our approach to determine the sets of nonzero rows and columns, and then compute $\mathbf{W}_{\mathrm{IJ}} = \mathbf{C}^{+}\mathbf{X}\mathbf{R}^{+}$.

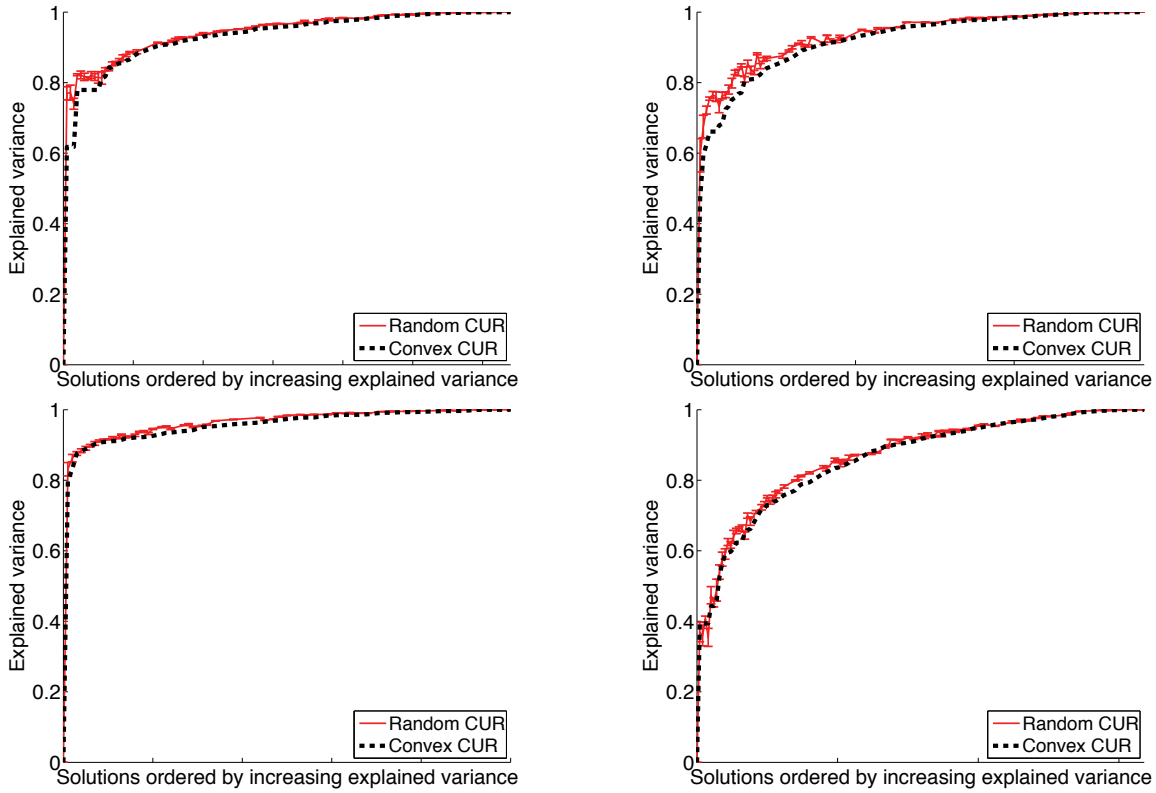


Figure 3: Explained variance of the CUR decompositions obtained for our sparsity-based approach and the sampling scheme from Mahoney and Drineas (2009). For the latter, we report the average and standard deviation of the results based on five initializations. From left to right and top to bottom, the curves correspond to the data sets `9_Tumors`, `Brain_Tumors1`, `Leukemia1` and `SRBCT`.

matrix. As a result, we could directly apply the same procedure as the one used in the other experiments. Instead, we further exploit the specific structure of problem (10): Notice that for a fixed vector $\mathbf{e}$, the optimization with respect to $\mathbf{w}$ is a standard Lasso problem (with the vector of observations $\mathbf{y} - \mathbf{e}$),[22] while for $\mathbf{w}$ fixed, we simply have a proximal problem associated to the sum of Ω and the ℓ_1-norm. Alternating between these two simple and computationally inexpensive steps, that is, optimizing with respect to one variable while keeping the other one fixed, is guaranteed to converge to a solution of (10).[23] In our simulations, this alternating scheme has led to a significant speed-up compared to the general procedure.

A data set with hand-segmented images is used to illustrate the effect of Ω.[24] For simplicity, we use a single regularization parameter, that is, $\lambda_1 = \lambda_2$, chosen to maximize the number of pixels

22. Since successive frames might not change much, the columns of $\mathbf{X}$ exhibit strong correlations. Consequently, we use the LARS algorithm (Efron et al., 2004) whose complexity is independent of the level of correlation in $\mathbf{X}$.
23. More precisely, the convergence is guaranteed since the non-smooth part in (10) is *separable* with respect to $\mathbf{w}$ and $\mathbf{e}$ (Tseng, 2001). The result from Bertsekas (1999) may also be applied here, after reformulating (10) as a smooth convex problem under separable conic constraints.
24. Data set can be found at `http://research.microsoft.com/en-us/um/people/jckrumm/wallflower/testimages.htm`.

matching the ground truth. We consider $p = 200$ images with $n = 57600$ pixels (i.e., a resolution of 120×160, times 3 for the RGB channels). As shown in Figure 4, adding Ω improves the background subtraction results for the two tested images, by removing the scattered artifacts due to the lack of structural constraints of the ℓ_1-norm, which encodes neither spatial nor color consistency. The group sparsity regularization $\tilde{\Omega}$ also improves upon the ℓ_1-norm but introduces block-artefacts corresponding to the non-overlapping group structure.

5.5 Topographic Dictionary Learning

Let us consider a set $\mathbf{Y} = [\mathbf{y}^1, \ldots, \mathbf{y}^n]$ in $\mathbb{R}^{m \times n}$ of n signals of dimension m. The problem of dictionary learning, originally introduced by Olshausen and Field (1996), is a matrix factorization problem which aims at representing these signals as linear combinations of *dictionary elements* that are the columns of a matrix $\mathbf{X} = [\mathbf{x}^1, \ldots, \mathbf{x}^p]$ in $\mathbb{R}^{m \times p}$. More precisely, the dictionary $\mathbf{X}$ is *learned* along with a matrix of *decomposition coefficients* $\mathbf{W} = [\mathbf{w}^1, \ldots, \mathbf{w}^n]$ in $\mathbb{R}^{p \times n}$, so that $\mathbf{y}^i \approx \mathbf{X}\mathbf{w}^i$ for every signal $\mathbf{y}^i$. Typically, n is large compared to m and p. In this experiment, we consider for instance a database of $n = 100\,000$ natural image patches of size $m = 12 \times 12$ pixels, for dictionaries of size $p = 400$. Adapting the dictionary to specific data has proven to be useful in many applications, including image restoration (Elad and Aharon, 2006; Mairal et al., 2009), learning image features in computer vision (Kavukcuoglu et al., 2009). The resulting optimization problem can be written

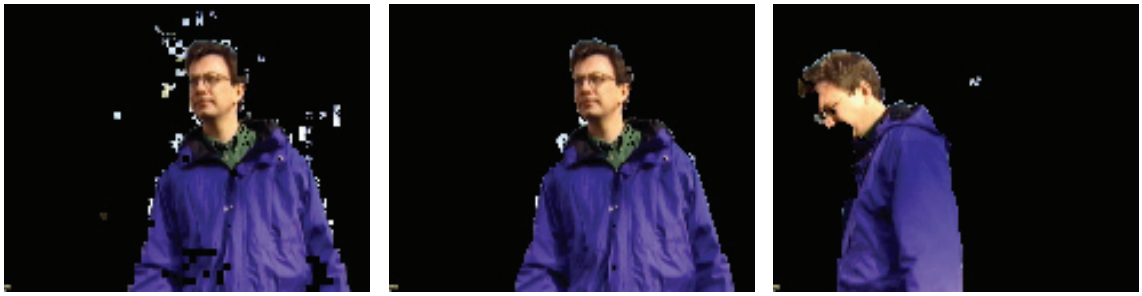
$$\min_{\mathbf{X} \in \mathcal{C}, \mathbf{W} \in \mathbb{R}^{p \times n}} \sum_{i=1}^{n} \frac{1}{2} \|\mathbf{y}^i - \mathbf{X}\mathbf{w}^i\|_2^2 + \lambda \Omega(\mathbf{w}^i), \tag{11}$$

where $\mathcal{C}$ is a convex set of matrices in $\mathbb{R}^{m \times p}$ whose columns have ℓ_2-norms less than or equal to one,[25] λ is a regularization parameter and Ω is a sparsity-inducing norm. When Ω is the ℓ_1-norm, we obtain a classical formulation, which is known to produce dictionary elements that are reminiscent of Gabor-like functions, when the columns of $\mathbf{Y}$ are whitened natural image patches (Olshausen and Field, 1996).

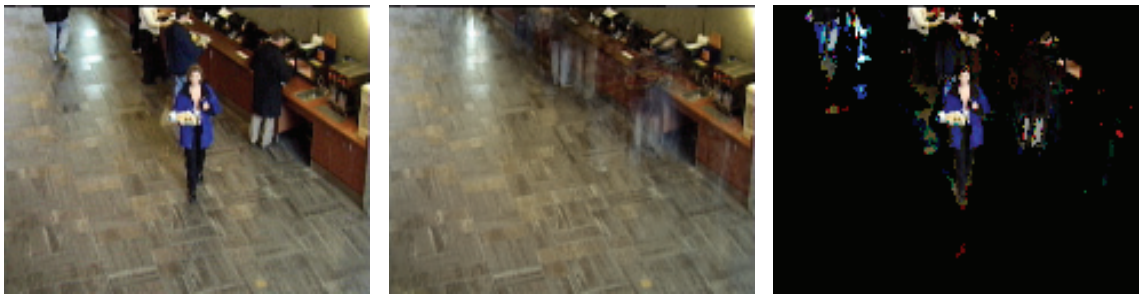
Another line of research tries to put a structure on decomposition coefficients instead of considering them as independent. Jenatton et al. (2010a, 2011) have for instance embedded dictionary elements into a tree, by using a hierarchical norm (Zhao et al., 2009) for Ω. This model encodes a rule saying that a dictionary element can be used in the decomposition of a signal only if its ancestors in the tree are used as well. In the related context of independent component analysis (ICA), Hyvärinen et al. (2001) have arranged independent components (corresponding to dictionary elements) on a two-dimensional grid, and have modelled spatial dependencies between them. When learned on whitened natural image patches, this model exhibits "Gabor-like" functions which are smoothly organized on the grid, which the authors call a topographic map. As shown by Kavukcuoglu et al. (2009), such a result can be reproduced with a dictionary learning formulation, using a structured norm for Ω. Following their formulation, we organize the p dictionary elements on a $\sqrt{p} \times \sqrt{p}$ grid, and consider p overlapping groups that are 3×3 or 4×4 spatial neighborhoods on the grid (to avoid boundary effects, we assume the grid to be cyclic). We define Ω as a sum of ℓ_2-norms over these groups, since the ℓ_∞-norm has proven to be less adapted for this task. Another formulation achieving a similar effect was also proposed by Garrigues and Olshausen (2010) in the context of sparse coding with a probabilistic model.

25. Since the quadratic term in Equation (11) is invariant by multiplying $\mathbf{X}$ by a scalar and $\mathbf{W}$ by its inverse, constraining the norm of $\mathbf{X}$ has proven to be necessary in practice to prevent it from being arbitrarily large.

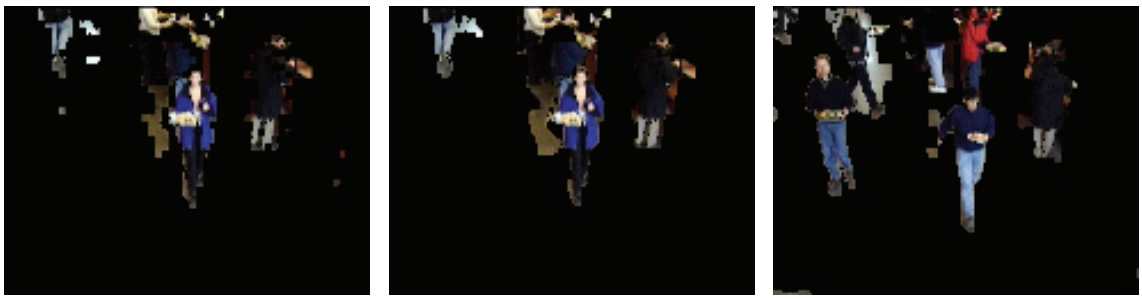
(a) Original frame. (b) Estimated background with Ω. (c) ℓ_1, 87.1%.



(d) $\ell_1 + \tilde{\Omega}$ (non-overlapping), 96.3%. (e) $\ell_1 + \Omega$ (overlapping), 98.9%. (f) Ω, another frame.



(g) Original frame. (h) Estimated background with Ω. (i) ℓ_1, 90.5%.



(j) $\ell_1 + \tilde{\Omega}$ (non-overlapping), 92.6%. (k) $\ell_1 + \Omega$ (overlapping), 93.8%. (l) Ω, another frame.

Figure 4: Background subtraction results. For two videos, we present the original image $\mathbf{y}$, the estimated background (i.e., $\mathbf{Xw}$) reconstructed by our method, and the foreground (i.e., the sparsity pattern of $\mathbf{e}$ as a mask on the original image) detected with ℓ_1, $\ell_1 + \tilde{\Omega}$ (non-overlapping groups) and with $\ell_1 + \Omega$. Figures (f) and (l) present another foreground found with Ω, on a different image, with the same values of λ_1, λ_2 as for the previous image. Best seen in color.

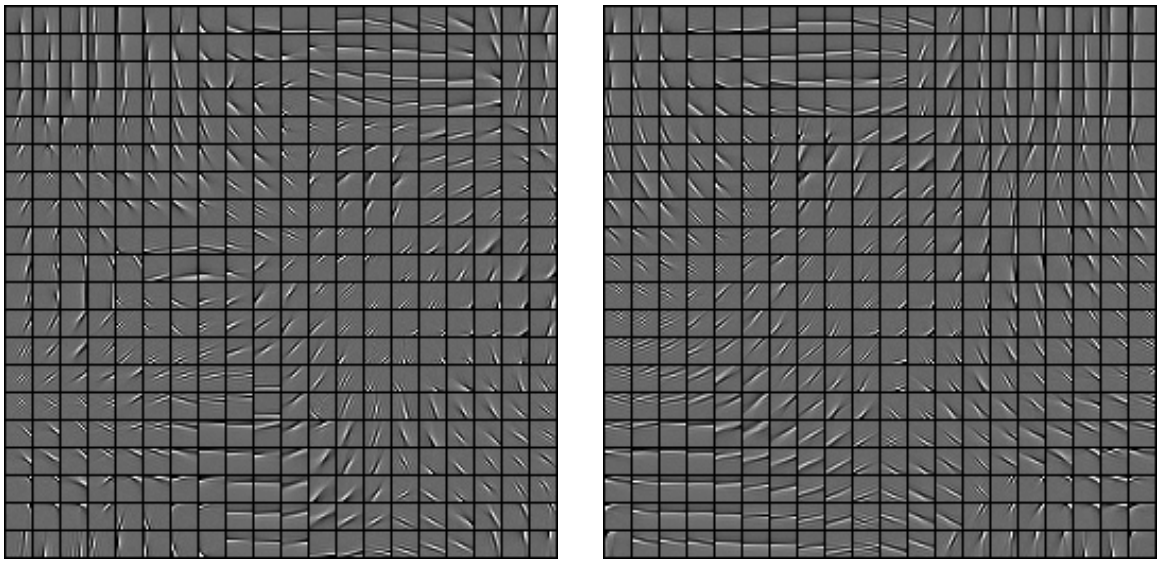


Figure 5: Topographic dictionaries with 400 elements, learned on a database of 12×12 whitened natural image patches with 3×3 (left) and 4×4 (right) cyclic overlapping groups.

As Kavukcuoglu et al. (2009) and Olshausen and Field (1996), we consider a projected stochastic gradient descent algorithm for learning $\mathbf{X}$—that is, at iteration t, we randomly draw one signal $\mathbf{y}^t$ from the database $\mathbf{Y}$, compute a sparse code $\mathbf{w}^t = \arg\min_{\mathbf{w}\in\mathbb{R}^p} \frac{1}{2}\|\mathbf{y}^t - \mathbf{X}\mathbf{w}^t\|_2^2 + \lambda\Omega(\mathbf{w})$, and update $\mathbf{X}$ as follows: $\mathbf{X} \leftarrow \Pi_{\mathcal{C}}[\mathbf{X} - \rho(\mathbf{X}\mathbf{w}^t - \mathbf{y}^t)\mathbf{w}^{t\top}]$, where ρ is a fixed learning rate, and $\Pi_{\mathcal{C}}$ denotes the operator performing orthogonal projections onto the set $\mathcal{C}$. In practice, to further improve the performance, we use a mini-batch, drawing 500 signals at eatch iteration instead of one (see Mairal et al., 2010a). Our approach mainly differs from Kavukcuoglu et al. (2009) in the way the sparse codes $\mathbf{w}^t$ are obtained. Whereas Kavukcuoglu et al. (2009) uses a subgradient descent algorithm to solve them, we use the proximal splitting methods presented in Section 4. The natural image patches we use are also preprocessed: They are first centered by removing their mean value (often called DC component), and whitened, as often done in the literature (Hyvärinen et al., 2001; Garrigues and Olshausen, 2010). The parameter λ is chosen such that in average $\|\mathbf{y}^i - \mathbf{X}\mathbf{w}^i\|_2 \approx 0.4\|\mathbf{y}^i\|_2$ for all new patch considered by the algorithm. Examples of obtained results are shown on Figure 5, and exhibit similarities with the topographic maps of Hyvärinen et al. (2001). Note that even though Equation (11) is convex with respect to each variable $\mathbf{X}$ and $\mathbf{W}$ when one fixes the other, it is not jointly convex, and one can not guarantee our method to find a global optimum. Despite its intrinsic non-convex nature, local minima obtained with various optimization procedures have been shown to be good enough for many tasks (Elad and Aharon, 2006; Mairal et al., 2009; Kavukcuoglu et al., 2009).

5.6 Multi-Task Learning of Hierarchical Structures

As mentioned in the previous section, Jenatton et al. (2010a) have recently proposed to use a hierarchical structured norm to learn dictionaries of natural image patches. In Jenatton et al. (2010a), the dictionary elements are embedded in a *predefined* tree $\mathcal{T}$, via a particular instance of the structured norm Ω, which we refer to it as Ω_{tree}, and call $\mathcal{G}$ the underlying set of groups. In this case, using

the same notation as in Section 5.5, each signal $\mathbf{y}^i$ admits a sparse decomposition in the form of a subtree of dictionary elements.

Inspired by ideas from multi-task learning (Obozinski et al., 2010), we propose to learn the tree structure $\mathcal{T}$ by pruning irrelevant parts of a larger initial tree $\mathcal{T}_0$. We achieve this by using an additional regularization term Ω_{joint} across the different decompositions, so that subtrees of $\mathcal{T}_0$ will *simultaneously* be removed for all signals $\mathbf{y}^i$. With the notation from Section 5.5, the approach of Jenatton et al. (2010a) is then extended by the following formulation:

$$\min_{\mathbf{X}\in C,\mathbf{W}\in\mathbb{R}^{p\times n}} \frac{1}{n}\sum_{i=1}^{n}\left[\frac{1}{2}\|\mathbf{y}^i-\mathbf{X}\mathbf{w}^i\|_2^2+\lambda_1\Omega_{\text{tree}}(\mathbf{w}^i)\right]+\lambda_2\Omega_{\text{joint}}(\mathbf{W}), \tag{12}$$

where $\mathbf{W}\triangleq[\mathbf{w}^1,\ldots,\mathbf{w}^n]$ is the matrix of decomposition coefficients in $\mathbb{R}^{p\times n}$. The new regularization term operates on the rows of $\mathbf{W}$ and is defined as $\Omega_{\text{joint}}(\mathbf{W})\triangleq\sum_{g\in\mathcal{G}}\max_{i\in\{1,\ldots,n\}}|\mathbf{w}_g^i|$.[26] The overall penalty on $\mathbf{W}$, which results from the combination of Ω_{tree} and Ω_{joint}, is itself an instance of Ω with general overlapping groups, as defined in Equation (2).

To address problem (12), we use the same optimization scheme as Jenatton et al. (2010a), that is, alternating between $\mathbf{X}$ and $\mathbf{W}$, fixing one variable while optimizing with respect to the other. The task we consider is the denoising of natural image patches, with the same data set and protocol as Jenatton et al. (2010a). We study whether learning the hierarchy of the dictionary elements improves the denoising performance, compared to standard sparse coding (i.e., when Ω_{tree} is the ℓ_1-norm and $\lambda_2=0$) and the hierarchical dictionary learning of Jenatton et al. (2010a) based on predefined trees (i.e., $\lambda_2=0$). The dimensions of the training set—50000 patches of size 8×8 for dictionaries with up to $p=400$ elements—impose to handle extremely large graphs, with $|E|\approx|V|\approx 4.10^7$. Since problem (12) is too large to be solved exactly sufficiently many times to select the regularization parameters (λ_1,λ_2) rigorously, we use the following heuristics: we optimize mostly with the currently pruned tree held fixed (i.e., $\lambda_2=0$), and only prune the tree (i.e., $\lambda_2>0$) every few steps on a random subset of 10000 patches. We consider the same hierarchies as in Jenatton et al. (2010a), involving between 30 and 400 dictionary elements. The regularization parameter λ_1 is selected on the validation set of 25000 patches, for both sparse coding (Flat) and hierarchical dictionary learning (Tree). Starting from the tree giving the best performance (in this case the largest one, see Figure 6), we solve problem (12) following our heuristics, for increasing values of λ_2. As shown in Figure 6, there is a regime where our approach performs significantly better than the two other compared methods. The standard deviation of the noise is 0.2 (the pixels have values in $[0,1]$); no significant improvements were observed for lower levels of noise. Our experiments use the algorithm of Beck and Teboulle (2009) based on our proximal operator, with weights η_g set to 1. We present this algorithm in more details in Appendix C.

6. Conclusion

We have presented new optimization methods for solving sparse structured problems involving sums of ℓ_2- or ℓ_∞-norms of any (overlapping) groups of variables. Interestingly, this sheds new light on connections between sparse methods and the literature of network flow optimization. In particular, the proximal operator for the sum of ℓ_∞-norms can be cast as a specific form of quadratic min-cost flow problem, for which we proposed an efficient and simple algorithm.

26. The simplified case where Ω_{tree} and Ω_{joint} are the ℓ_1- and mixed ℓ_1/ℓ_2-norms (Yuan and Lin, 2006) corresponds to Sprechmann et al. (2010).

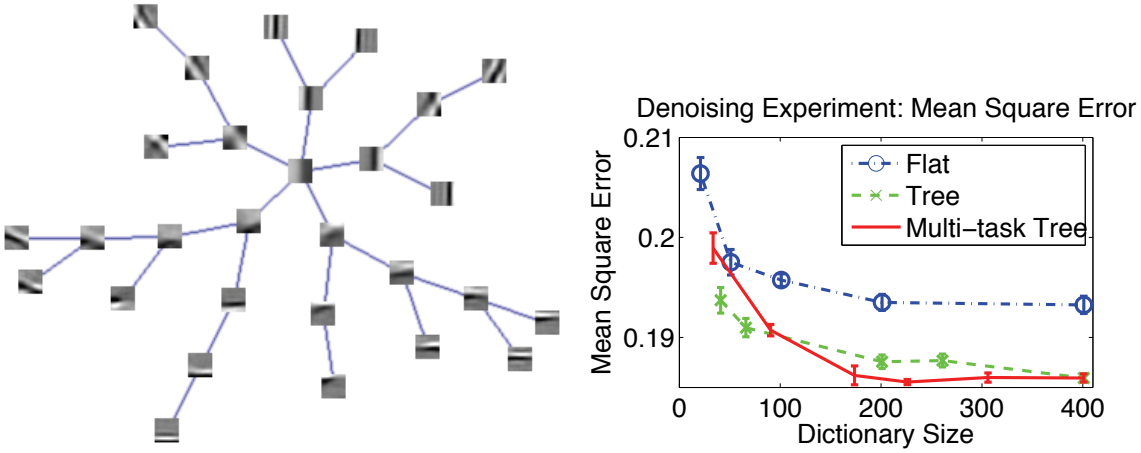


Figure 6: Left: Hierarchy obtained by pruning a larger tree of 76 elements. Right: Mean square error versus dictionary size. The error bars represent two standard deviations, based on three runs.

In addition to making it possible to resort to accelerated gradient methods, an efficient computation of the proximal operator offers more generally a certain modularity, in that it can be used as a building-block for other optimization problems. A case in point is dictionary learning where proximal problems come up and have to be solved repeatedly in an inner-loop. Interesting future work includes the computation of other structured norms such as the one introduced by Jacob et al. (2009), or total-variation based penalties, whose proximal operators are also based on minimum cost flow problems (Chambolle and Darbon, 2009). Several experiments demonstrate that our algorithm can be applied to a wide class of learning problems, which have not been addressed before with convex sparse methods.

Acknowledgments

This paper was partially supported by grants from the Agence Nationale de la Recherche (MGA Project) and from the European Research Council (SIERRA Project). In addition, Julien Mairal was supported by the NSF grant SES-0835531 and NSF award CCF-0939370. The authors would like to thank Jean Ponce for interesting discussions and suggestions for improving this manuscript, Jean-Christophe Pesquet and Patrick-Louis Combettes for pointing us to the literature of proximal splitting methods, Ryota Tomioka for his advice on using augmented Lagrangian methods.

Appendix A. Equivalence to Canonical Graphs

Formally, the notion of equivalence between graphs can be summarized by the following lemma:

Lemma 4 (Equivalence to canonical graphs.)
Let $G=(V,E,s,t)$ be the canonical graph corresponding to a group structure $\mathcal{G}$. Let $G'=(V,E',s,t)$ be a graph sharing the same set of vertices, source and sink as G, but with a different arc set E'. We say that G' is equivalent to G if and only if the following conditions hold:

- *Arcs of E' outgoing from the source are the same as in E, with the same costs and capacities.*

- *Arcs of E' going to the sink are the same as in E, with the same costs and capacities.*
- *For every arc (g,j) in E, with (g,j) in $V_{gr} \times V_u$, there exists a unique path in E' from g to j with zero costs and infinite capacities on every arc of the path.*
- *Conversely, if there exists a path in E' between a vertex g in V_{gr} and a vertex j in V_u, then there exists an arc (g,j) in E.*

Then, the cost of the optimal min-cost flow on G and G' are the same. Moreover, the values of the optimal flow on the arcs (j,t), j in V_u, are the same on G and G'.

Proof We first notice that on both G and G', the cost of a flow on the graph only depends on the flow on the arcs (j,t), j in V_u, which we have denoted by $\overline{\xi}$ in E.

We will prove that finding a feasible flow π on G with a cost $c(\pi)$ is equivalent to finding a feasible flow π' on G' with the same cost $c(\pi) = c(\pi')$. We now use the concept of *path flow*, which is a flow vector in G carrying the same positive value on every arc of a directed path between two nodes of G. It intuitively corresponds to sending a positive amount of flow along a path of the graph.

According to the definition of graph equivalence introduced in the Lemma, it is easy to show that there is a bijection between the arcs in E, and the paths in E' with positive capacities on every arc. Given now a feasible flow π in G, we build a feasible flow π' on G' which is a *sum* of path flows. More precisely, for every arc a in E, we consider its equivalent path in E', with a path flow carrying the same amount of flow as a. Therefore, each arc a' in E' has a total amount of flow that is equal to the sum of the flows carried by the path flows going over a'. It is also easy to show that this construction builds a flow on G' (capacity and conservation constraints are satisfied) and that this flow π' has the same cost as π, that is, $c(\pi) = c(\pi')$.

Conversely, given a flow π' on G', we use a classical path flow decomposition (see Bertsekas, 1998, Proposition 1.1), saying that there exists a decomposition of π' as a sum of path flows in E'. Using the bijection described above, we know that each path in the previous sums corresponds to a unique arc in E. We now build a flow π in G, by associating to each path flow in the decomposition of π', an arc in E carrying the same amount of flow. The flow of every other arc in E is set to zero. It is also easy to show that this builds a valid flow in G that has the same cost as π'. ■

Appendix B. Convergence Analysis

We show in this section the correctness of Algorithm 1 for computing the proximal operator, and of Algorithm 2 for computing the dual norm $\Omega^\star$.

B.1 Computation of the Proximal Operator

We first prove that our algorithm converges and that it finds the optimal solution of the proximal problem. This requires that we introduce the optimality conditions for problem (4) derived from Jenatton et al. (2010a, 2011) since our convergence proof essentially checks that these conditions are satisfied upon termination of the algorithm.

Lemma 5 (Optimality conditions of the problem (4) from Jenatton et al. 2010a, 2011)
The primal-dual variables $(\mathbf{w}, \xi)$ are respectively solutions of the primal (3) and dual problems (4)

if and only if the dual variable ξ is feasible for the problem (4) and

$$\mathbf{w} = \mathbf{u} - \sum_{g \in \mathcal{G}} \xi^g,$$
$$\forall g \in \mathcal{G}, \quad \begin{cases} \mathbf{w}_g^\top \xi_g^g = \|\mathbf{w}_g\|_\infty \|\xi^g\|_1 \ \textit{and} \ \|\xi^g\|_1 = \lambda \eta_g, \\ \textit{or} \ \mathbf{w}_g = 0. \end{cases}$$

Note that these optimality conditions provide an intuitive view of our min-cost flow problem. Solving the min-cost flow problem is equivalent to sending the maximum amount of flow in the graph under the capacity constraints, while respecting the rule that *the flow coming from a group g should always be directed to the variables $\mathbf{u}_j$ with maximum residual $\mathbf{u}_j - \sum_{g \in \mathcal{G}} \xi_j^g$*. This point can be more formaly seen by noticing that one of the optimality conditions above corresponds to the case of equality in the ℓ_1/ℓ_∞ Hölder inequality.

Before proving the convergence and correctness of our algorithm, we also recall classical properties of the min capacity cuts, which we intensively use in the proofs of this paper. The procedure `computeFlow` of our algorithm finds a minimum (s,t)-cut of a graph $G = (V,E,s,t)$, dividing the set V into two disjoint parts V^+ and V^-. V^+ is by construction the sets of nodes in V such that there exists a non-saturating path from s to V, while all the paths from s to V^- are saturated. Conversely, arcs from V^+ to t are all saturated, whereas there can be non-saturated arcs from V^- to t. Moreover, the following properties, which are illustrated on Figure 7, hold

- There is no arc going from V^+ to V^-. Otherwise the value of the cut would be infinite (arcs inside V have infinite capacity by construction of our graph).
- There is no flow going from V^- to V^+ (see Bertsekas, 1998).
- The cut goes through all arcs going from V^+ to t, and all arcs going from s to V^-.

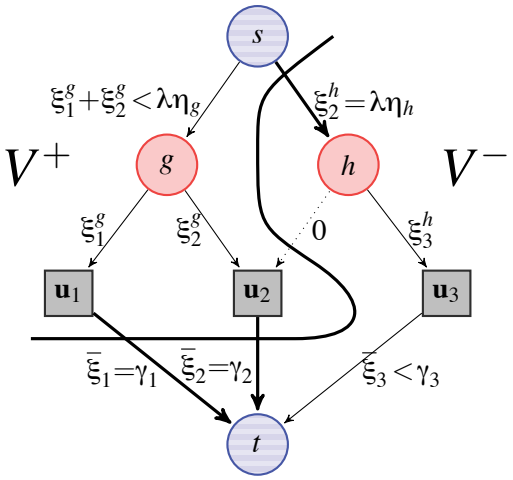


Figure 7: Cut computed by our algorithm. $V^+ = V_u^+ \cup V_{gr}^+$, with $V_{gr}^+ = \{g\}$, $V_u^+ = \{1,2\}$, and $V^- = V_u^- \cup V_{gr}^-$, with $V_{gr}^- = \{h\}$, $V_u^- = \{3\}$. Arcs going from s to V^- are saturated, as well as arcs going from V^+ to t. Saturated arcs are in bold. Arcs with zero flow are dotted.

Recall that we assume (cf. Section 3.3) that the scalars $\mathbf{u}_j$ are all non negative, and that we add non-negativity constraints on ξ. With the optimality conditions of Lemma 5 in hand, we can show our first convergence result.

Proposition 6 (Convergence of Algorithm 1)
Algorithm 1 converges in a finite and polynomial number of operations.

Proof Our algorithm splits recursively the graph into disjoints parts and processes each part recursively. The processing of one part requires an orthogonal projection onto an ℓ_1-ball and a max-flow algorithm, which can both be computed in polynomial time. To prove that the procedure converges, it is sufficient to show that when the procedure `computeFlow` is called for a graph (V,E,s,t) and computes a cut (V^+,V^-), then the components V^+ and V^- are both non-empty.

Suppose for instance that $V^-=\emptyset$. In this case, the capacity of the min-cut is equal to $\sum_{j\in V_u}\gamma_j$, and the value of the max-flow is $\sum_{j\in V_u}\overline{\xi}_j$. Using the classical max-flow/min-cut theorem (Ford and Fulkerson, 1956), we have equality between these two terms. Since, by definition of both γ and $\overline{\xi}$, we have for all j in V_u, $\overline{\xi}_j \leq \gamma_j$, we obtain a contradiction with the existence of j in V_u such that $\overline{\xi}_j \neq \gamma_j$.

Conversely, suppose now that $V^+=\emptyset$. Then, the value of the max-flow is still $\sum_{j\in V_u}\overline{\xi}_j$, and the value of the min-cut is $\lambda\sum_{g\in V_{gr}}\eta_g$. Using again the max-flow/min-cut theorem, we have that $\sum_{j\in V_u}\overline{\xi}_j = \lambda\sum_{g\in V_{gr}}\eta_g$. Moreover, by definition of γ, we also have $\sum_{j\in V_u}\overline{\xi}_j \leq \sum_{j\in V_u}\gamma_j \leq \lambda\sum_{g\in V_{gr}}\eta_g$, leading to a contradiction with the existence of j in V_u satisfying $\overline{\xi}_j \neq \gamma_j$. We remind the reader of the fact that such a $j\in V_u$ exists since the cut is only computed when the current estimate $\overline{\xi}$ is not optimal yet. This proof holds for any graph that is equivalent to the canonical one. ■

After proving the convergence, we prove that the algorithm is correct with the next proposition.

Proposition 7 (Correctness of Algorithm 1)
Algorithm 1 solves the proximal problem of Equation (3).

Proof For a group structure $\mathcal{G}$, we first prove the correctness of our algorithm if the graph used is its associated canonical graph that we denote $G_0=(V_0,E_0,s,t)$. We proceed by induction on the number of nodes of the graph. The induction hypothesis $\mathcal{H}(k)$ is the following:

For all canonical graphs $G=(V=V_u\cup V_{gr},E,s,t)$ associated with a group structure $\mathcal{G}_V$ with weights $(\eta_g)_{g\in\mathcal{G}_V}$ such that $|V|\leq k$, `computeFlow`(V,E) solves the following optimization problem:

$$\min_{(\xi_j^g)_{j\in V_u,g\in V_{gr}}} \sum_{j\in V_u}\frac{1}{2}(\mathbf{u}_j-\sum_{g\in V_{gr}}\xi_j^g)^2 \text{ s.t. } \forall g\in V_{gr},\ \sum_{j\in V_u}\xi_j^g\leq\lambda\eta_g \text{ and } \xi_j^g=0,\ \forall j\notin g. \qquad (13)$$

Since $\mathcal{G}_{V_0}=\mathcal{G}$, it is sufficient to show that $\mathcal{H}(|V_0|)$ to prove the proposition.

We initialize the induction by $\mathcal{H}(2)$, corresponding to the simplest canonical graph, for which $|V_{gr}|=|V_u|=1$). Simple algebra shows that $\mathcal{H}(2)$ is indeed correct.

We now suppose that $\mathcal{H}(k')$ is true for all $k'<k$ and consider a graph $G=(V,E,s,t)$, $|V|=k$. The first step of the algorithm computes the variable $(\gamma_j)_{j\in V_u}$ by a projection on the ℓ_1-ball. This is

itself an instance of the dual formulation of Equation (4) in a simple case, with one group containing all variables. We can therefore use Lemma 5 to characterize the optimality of $(\gamma_j)_{j\in V_u}$, which yields

$$\left\{\begin{array}{l} \sum_{j\in V_u}(\mathbf{u}_j-\gamma_j)\gamma_j = \left(\max_{j\in V_u}|\mathbf{u}_j-\gamma_j|\right)\sum_{j\in V_u}\gamma_j \text{ and } \sum_{j\in V_u}\gamma_j = \lambda\sum_{g\in V_{gr}}\eta_g, \\ \text{or } \mathbf{u}_j-\gamma_j = 0,\ \forall j\in V_u. \end{array}\right. \tag{14}$$

The algorithm then computes a max-flow, using the scalars γ_j as capacities, and we now have two possible situations:

1. If $\overline{\xi}_j = \gamma_j$ for all j in V_u, the algorithm stops; we write $\mathbf{w}_j = \mathbf{u}_j - \overline{\xi}_j$ for j in V_u, and using Equation (14), we obtain

   $$\left\{\begin{array}{l} \sum_{j\in V_u}\mathbf{w}_j\overline{\xi}_j = (\max_{j\in V_u}|\mathbf{w}_j|)\sum_{j\in V_u}\overline{\xi}_j \text{ and } \sum_{j\in V_u}\overline{\xi}_j = \lambda\sum_{g\in V_{gr}}\eta_g, \\ \text{or } \mathbf{w}_j = 0,\ \forall j\in V_u. \end{array}\right.$$

   We can rewrite the condition above as

   $$\sum_{g\in V_{gr}}\sum_{j\in g}\mathbf{w}_j\xi_j^g = \sum_{g\in V_{gr}}(\max_{j\in V_u}|\mathbf{w}_j|)\sum_{j\in V_u}\xi_j^g.$$

   Since all the quantities in the previous sum are positive, this can only hold if for all $g\in V_{gr}$,

   $$\sum_{j\in V_u}\mathbf{w}_j\xi_j^g = (\max_{j\in V_u}|\mathbf{w}_j|)\sum_{j\in V_u}\xi_j^g.$$

   Moreover, by definition of the max flow and the optimality conditions, we have

   $$\forall g\in V_{gr},\ \sum_{j\in V_u}\xi_j^g \le \lambda\eta_g, \text{ and } \sum_{j\in V_u}\overline{\xi}_j = \lambda\sum_{g\in V_{gr}}\eta_g,$$

   which leads to

   $$\forall g\in V_{gr},\ \sum_{j\in V_u}\xi_j^g = \lambda\eta_g.$$

   By Lemma 5, we have shown that the problem (13) is solved.

2. Let us now consider the case where there exists j in V_u such that $\overline{\xi}_j \neq \gamma_j$. The algorithm splits the vertex set V into two parts V^+ and V^-, which we have proven to be non-empty in the proof of Proposition 6. The next step of the algorithm removes all edges between V^+ and V^- (see Figure 7). Processing (V^+,E^+) and (V^-,E^-) independently, it updates the value of the flow matrix $\xi_j^g,\ j\in V_u,\ g\in V_{gr}$, and the corresponding flow vector $\overline{\xi}_j,\ j\in V_u$. As for V, we denote by $V_u^+ \triangleq V^+\cap V_u$, $V_u^- \triangleq V^-\cap V_u$ and $V_{gr}^+ \triangleq V^+\cap V_{gr}$, $V_{gr}^- \triangleq V^-\cap V_{gr}$.

   Then, we notice that (V^+,E^+,s,t) and (V^-,E^-,s,t) are respective canonical graphs for the group structures $\mathcal{G}_{V^+} \triangleq \{g\cap V_u^+ \mid g\in V_{gr}\}$, and $\mathcal{G}_{V^-} \triangleq \{g\cap V_u^- \mid g\in V_{gr}\}$.

   Writing $\mathbf{w}_j = \mathbf{u}_j - \overline{\xi}_j$ for j in V_u, and using the induction hypotheses $\mathcal{H}(|V^+|)$ and $\mathcal{H}(|V^-|)$, we now have the following optimality conditions deriving from Lemma 5 applied on Equation (13) respectively for the graphs (V^+,E^+) and (V^-,E^-):

   $$\forall g\in V_{gr}^+, g'\triangleq g\cap V_u^+,\ \left\{\begin{array}{l} \mathbf{w}_{g'}^\top\xi_{g'}^g = \|\mathbf{w}_{g'}\|_\infty\sum_{j\in g'}\xi_j^g \text{ and } \sum_{j\in g'}\xi_j^g = \lambda\eta_g, \\ \text{or } \mathbf{w}_{g'} = 0, \end{array}\right. \tag{15}$$

and

$$\forall g \in V_{gr}^-, g' \triangleq g \cap V_u^-, \left\{ \begin{array}{l} \mathbf{w}_{g'}^\top \xi_{g'}^g = \|\mathbf{w}_{g'}\|_\infty \sum_{j \in g'} \xi_j^g \text{ and } \sum_{j \in g'} \xi_j^g = \lambda \eta_g, \\ \text{or } \mathbf{w}_{g'} = 0. \end{array} \right. \quad (16)$$

We will now combine Equation (15) and Equation (16) into optimality conditions for Equation (13). We first notice that $g \cap V_u^+ = g$ since there are no arcs between V^+ and V^- in E (see the properties of the cuts discussed before this proposition). It is therefore possible to replace g' by g in Equation (15). We will show that it is possible to do the same in Equation (16), so that combining these two equations yield the optimality conditions of Equation (13).

More precisely, we will show that for all $g \in V_{gr}^-$ and $j \in g \cap V_u^+$, $|\mathbf{w}_j| \leq \max_{l \in g \cap V_u^-} |\mathbf{w}_l|$, in which case g' can be replaced by g in Equation (16). This result is relatively intuitive: (s, V^+) and (V^-, t) being an (s,t)-cut, all arcs between s and V^- are saturated, while there are unsaturated arcs between s and V^+; one therefore expects the residuals $\mathbf{u}_j - \overline{\xi}_j$ to decrease on the V^+ side, while increasing on the V^- side. The proof is nonetheless a bit technical.

Let us show first that for all g in V_{gr}^+, $\|\mathbf{w}_g\|_\infty \leq \max_{j \in V_u} |\mathbf{u}_j - \gamma_j|$. We split the set V^+ into disjoint parts:

$$\begin{aligned} V_{gr}^{++} &\triangleq \{g \in V_{gr}^+ \text{ s.t. } \|\mathbf{w}_g\|_\infty \leq \max_{j \in V_u} |\mathbf{u}_j - \gamma_j|\}, \\ V_u^{++} &\triangleq \{j \in V_u^+ \text{ s.t. } \exists g \in V_{gr}^{++},\ j \in g\}, \\ V_{gr}^{+-} &\triangleq V_{gr}^+ \setminus V_{gr}^{++} = \{g \in V_{gr}^+ \text{ s.t. } \|\mathbf{w}_g\|_\infty > \max_{j \in V_u} |\mathbf{u}_j - \gamma_j|\}, \\ V_u^{+-} &\triangleq V_u^+ \setminus V_u^{++}. \end{aligned}$$

As previously, we denote $V^{+-} \triangleq V_u^{+-} \cup V_{gr}^{+-}$ and $V^{++} \triangleq V_u^{++} \cup V_{gr}^{++}$. We want to show that V_{gr}^{+-} is necessarily empty. We reason by contradiction and assume that $V_{gr}^{+-} \neq \varnothing$.

According to the definition of the different sets above, we observe that no arcs are going from V^{++} to V^{+-}, that is, for all g in V_{gr}^{++}, $g \cap V_u^{+-} = \varnothing$. We observe as well that the flow from V_{gr}^{+-} to V_u^{++} is the null flow, because optimality conditions (15) imply that for a group g only nodes $j \in g$ such that $\mathbf{w}_j = \|\mathbf{w}_g\|_\infty$ receive some flow, which excludes nodes in V_u^{++} provided $V_{gr}^{+-} \neq \varnothing$; Combining this fact and the inequality $\sum_{g \in V_{gr}^+} \lambda \eta_g \geq \sum_{j \in V_u^+} \gamma_j$ (which is a direct consequence of the minimum (s,t)-cut), we have as well

$$\sum_{g \in V_{gr}^{+-}} \lambda \eta_g \geq \sum_{j \in V_u^{+-}} \gamma_j.$$

Let $j \in V_u^{+-}$, if $\overline{\xi}_j \neq 0$ then for some $g \in V_{gr}^{+-}$ such that j receives some flow from g, which from the optimality conditions (15) implies $\mathbf{w}_j = \|\mathbf{w}_g\|_\infty$; by definition of V_{gr}^{+-}, $\|\mathbf{w}_g\|_\infty > \mathbf{u}_j - \gamma_j$. But since at the optimum, $\mathbf{w}_j = \mathbf{u}_j - \overline{\xi}_j$, this implies that $\overline{\xi}_j < \gamma_j$, and in turn that $\sum_{j \in V_u^{+-}} \overline{\xi}_j = \lambda \sum_{g \in V_{gr}^{+-}} \eta_g$. Finally,

$$\lambda \sum_{g \in V_{gr}^{+-}} \eta_g = \sum_{j \in V_u^{+-}, \overline{\xi}_j \neq 0} \overline{\xi}_j < \sum_{j \in V_u^{+-}} \gamma_j$$

and this is a contradiction.

We now have that for all g in V_{gr}^+, $\|\mathbf{w}_g\|_\infty \leq \max_{j\in V_u} |\mathbf{u}_j - \gamma_j|$. The proof showing that for all g in V_{gr}^-, $\|\mathbf{w}_g\|_\infty \geq \max_{j\in V_u} |\mathbf{u}_j - \gamma_j|$, uses the same kind of decomposition for V^-, and follows along similar arguments. We will therefore not detail it.

To summarize, we have shown that for all $g \in V_{gr}^-$ and $j \in g \cap V_u^+$, $|\mathbf{w}_j| \leq \max_{l\in g\cap V_u^-} |\mathbf{w}_l|$. Since there is no flow from V^- to V^+, that is, $\xi_j^g = 0$ for g in V_{gr}^- and j in V_u^+, we can now replace the definition of g' in Equation (16) by $g' \triangleq g \cap V_u$, the combination of Equation (15) and Equation (16) gives us optimality conditions for Equation (13).

The proposition being proved for the canonical graph, we extend it now for an equivalent graph in the sense of Lemma 4. First, we observe that the algorithm gives the same values of γ for two equivalent graphs. Then, it is easy to see that the value $\bar{\xi}$ given by the max-flow, and the chosen (s,t)-cut is the same, which is enough to conclude that the algorithm performs the same steps for two equivalent graphs. ■

B.2 Computation of the Dual Norm $\Omega^\star$

As for the proximal operator, the computation of dual norm Ω^* can itself be shown to solve another network flow problem, based on the following variational formulation, which extends a previous result from Jenatton et al. (2009):

Lemma 8 (Dual formulation of the dual-norm $\Omega^\star$.)
Let $\kappa \in \mathbb{R}^p$. We have

$$\Omega^*(\kappa) = \min_{\xi\in\mathbb{R}^{p\times|\mathcal{G}|},\tau\in\mathbb{R}} \tau \quad s.t. \quad \sum_{g\in\mathcal{G}} \xi^g = \kappa, \text{ and } \forall g \in \mathcal{G},\ \|\xi^g\|_1 \leq \tau\eta_g \text{ with } \xi_j^g = 0 \text{ if } j \notin g.$$

Proof By definition of $\Omega^*(\kappa)$, we have

$$\Omega^*(\kappa) \triangleq \max_{\Omega(\mathbf{z})\leq 1} \mathbf{z}^\top\kappa.$$

By introducing the primal variables $(\alpha_g)_{g\in\mathcal{G}} \in \mathbb{R}^{|\mathcal{G}|}$, we can rewrite the previous maximization problem as

$$\Omega^*(\kappa) = \max_{\sum_{g\in\mathcal{G}}\eta_g\alpha_g\leq 1} \kappa^\top\mathbf{z}, \quad \text{s.t.} \quad \forall\, g \in \mathcal{G},\ \|\mathbf{z}_g\|_\infty \leq \alpha_g,$$

with the additional $|\mathcal{G}|$ conic constraints $\|\mathbf{z}_g\|_\infty \leq \alpha_g$. This primal problem is convex and satisfies Slater's conditions for generalized conic inequalities, which implies that strong duality holds (Boyd and Vandenberghe, 2004). We now consider the Lagrangian $\mathcal{L}$ defined as

$$\mathcal{L}(\mathbf{z},\alpha_g,\tau,\gamma_g,\xi) = \kappa^\top\mathbf{z} + \tau(1 - \sum_{g\in\mathcal{G}} \eta_g\alpha_g) + \sum_{g\in\mathcal{G}} \begin{pmatrix}\alpha_g\\ \mathbf{z}_g\end{pmatrix}^\top \begin{pmatrix}\gamma_g\\ \xi_g^g\end{pmatrix},$$

with the dual variables $\{\tau, (\gamma_g)_{g\in\mathcal{G}}, \xi\} \in \mathbb{R}_+ \times \mathbb{R}^{|\mathcal{G}|} \times \mathbb{R}^{p\times|\mathcal{G}|}$ such that for all $g \in \mathcal{G}$, $\xi_j^g = 0$ if $j \notin g$ and $\|\xi^g\|_1 \leq \gamma_g$. The dual function is obtained by taking the derivatives of $\mathcal{L}$ with respect to the primal variables $\mathbf{z}$ and $(\alpha_g)_{g\in\mathcal{G}}$ and equating them to zero, which leads to

$$\begin{aligned} \forall j \in \{1,\dots,p\}, \quad \kappa_j + \textstyle\sum_{g\in\mathcal{G}} \xi_j^g &= 0 \\ \forall g \in \mathcal{G}, \quad \tau\eta_g - \gamma_g &= 0. \end{aligned}$$

After simplifying the Lagrangian and flipping the sign of ξ, the dual problem then reduces to

$$\min_{\xi\in\mathbb{R}^{p\times|\mathcal{G}|},\tau\in\mathbb{R}} \tau \quad \text{s.t.} \begin{cases} \forall j\in\{1,\dots,p\}, \kappa_j = \sum_{g\in\mathcal{G}} \xi_j^g \text{ and } \xi_j^g = 0 \text{ if } j\notin g, \\ \forall g\in\mathcal{G}, \|\xi^g\|_1 \leq \tau\eta_g, \end{cases}$$

which is the desired result. ■

We now prove that Algorithm 2 is correct.

Proposition 9 (Convergence and correctness of Algorithm 2)
Algorithm 2 computes the value of Ω^* in a finite and polynomial number of operations.

Proof The convergence of the algorithm only requires to show that the cardinality of V in the different calls of the function `computeFlow` strictly decreases. Similar arguments to those used in the proof of Proposition 6 can show that each part of the cuts (V^+,V^-) are both non-empty. The algorithm thus requires a finite number of calls to a max-flow algorithm and converges in a finite and polynomial number of operations.

Let us now prove that the algorithm is correct for a canonical graph. We proceed again by induction on the number of nodes of the graph. More precisely, we consider the induction hypothesis $\mathcal{H}'(k)$ defined as:

for all canonical graphs $G=(V,E,s,t)$ associated with a group structure $\mathcal{G}_V$ and *such that* $|V|\leq k$, `dualNormAux`$(V=V_u\cup V_{gr},E)$ *solves the following optimization problem:*

$$\min_{\xi,\tau} \tau \quad \text{s.t.} \quad \forall j\in V_u, \kappa_j = \sum_{g\in V_{gr}} \xi_j^g, \text{ and } \forall g\in V_{gr}, \sum_{j\in V_u} \xi_j^g \leq \tau\eta_g \text{ with } \xi_j^g = 0 \text{ if } j\notin g. \qquad (17)$$

We first initialize the induction by $\mathcal{H}(2)$ (i.e., with the simplest canonical graph, such that $|V_{gr}| = |V_u| = 1$). Simple algebra shows that $\mathcal{H}(2)$ is indeed correct.

We next consider a canonical graph $G=(V,E,s,t)$ such that $|V|=k$, and suppose that $\mathcal{H}'(k-1)$ is true. After the max-flow step, we have two possible cases to discuss:

1. If $\bar{\xi}_j = \gamma_j$ for all j in V_u, the algorithm stops. We know that any scalar τ such that the constraints of Equation (17) are all satisfied necessarily verifies $\sum_{g\in V_{gr}} \tau\eta_g \geq \sum_{j\in V_u} \kappa_j$. We have indeed that $\sum_{g\in V_{gr}} \tau\eta_g$ is the value of an (s,t)-cut in the graph, and $\sum_{j\in V_u} \kappa_j$ is the value of the max-flow, and the inequality follows from the max-flow/min-cut theorem (Ford and Fulkerson, 1956). This gives a lower-bound on τ. Since this bound is reached, τ is optimal.

2. We now consider the case where there exists j in V_u such that $\bar{\xi}_j \neq \kappa_j$, meaning that for the given value of τ, the constraint set of Equation (17) is not feasible for ξ, and that the value of τ should necessarily increase. The algorithm splits the vertex set V into two non-empty parts V^+ and V^- and we remark that there are no arcs going from V^+ to V^-, and no flow going from V^- to V^+. Since the arcs going from s to V^- are saturated, we have that $\sum_{g\in V_{gr}^-} \tau\eta_g \leq \sum_{j\in V_u^-} \kappa_j$. Let us now consider $\tau^\star$ the solution of Equation (17). Using the induction hypothesis $\mathcal{H}'(|V^-|)$, the algorithm computes a new value τ' that solves Equation (17) when replacing V by V^- and this new value satisfies the following inequality $\sum_{g\in V_{gr}^-} \tau'\eta_g \geq \sum_{j\in V_u^-} \kappa_j$. The value of τ' has therefore increased and the updated flow ξ now satisfies the constraints of Equation (17) and therefore $\tau'\geq\tau^\star$. Since there are no arcs going from V^+ to V^-, $\tau^\star$ is feasible for Equation (17) when replacing V by V^- and we have that $\tau^\star\geq\tau'$ and then $\tau'=\tau^\star$.

To prove that the result holds for any equivalent graph, similar arguments to those used in the proof of Proposition 6 can be exploited, showing that the algorithm computes the same values of τ and same (s,t)-cuts at each step. ■

Appendix C. Algorithm FISTA with Duality Gap

In this section, we describe in details the algorithm FISTA (Beck and Teboulle, 2009) when applied to solve problem (1), with a duality gap as the stopping criterion. The algorithm, as implemented in the experiments, is summarized in Algorithm 3.

Without loss of generality, let us assume we are looking for models of the form $\mathbf{Xw}$, for some matrix $\mathbf{X} \in \mathbb{R}^{n \times p}$ (typically, a linear model where $\mathbf{X}$ is the design matrix composed of n observations in $\mathbb{R}^p$). Thus, we can consider the following primal problem

$$\min_{\mathbf{w} \in \mathbb{R}^p} f(\mathbf{Xw}) + \lambda \Omega(\mathbf{w}), \tag{18}$$

in place of problem (1). Based on Fenchel duality arguments (Borwein and Lewis, 2006),

$$f(\mathbf{Xw}) + \lambda \Omega(\mathbf{w}) + f^*(-\kappa), \text{ for } \mathbf{w} \in \mathbb{R}^p, \kappa \in \mathbb{R}^n \text{ and } \Omega^*(\mathbf{X}^\top \kappa) \leq \lambda,$$

is a duality gap for problem (18), where $f^*(\kappa) \triangleq \sup_{\mathbf{z}}[\mathbf{z}^\top \kappa - f(\mathbf{z})]$ is the Fenchel conjugate of f (Borwein and Lewis, 2006). Given a primal variable $\mathbf{w}$, a good dual candidate κ can be obtained by looking at the conditions that have to be satisfied by the pair $(\mathbf{w}, \kappa)$ at optimality (Borwein and Lewis, 2006). In particular, the dual variable κ is chosen to be

$$\kappa = -\rho^{-1} \nabla f(\mathbf{Xw}), \text{ with } \rho \triangleq \max\left\{\lambda^{-1} \Omega^*(\mathbf{X}^\top \nabla f(\mathbf{Xw})), 1\right\}.$$

Consequently, computing the duality gap requires evaluating the dual norm Ω^*, as explained in Algorithm 2. We sum up the computation of the duality gap in Algorithm 3. Moreover, we refer to the proximal operator associated with $\lambda\Omega$ as $\text{prox}_{\lambda\Omega}$.[27] In our experiments, we choose the line-search parameter ν to be equal to 1.5.

Appendix D. Speed Comparison of Algorithm 1 with Parametric Max-Flow Solvers

As shown by Hochbaum and Hong (1995), min-cost flow problems, and in particular, the dual problem of (3), can be reduced to a specific *parametric max-flow* problem. We thus compare our approach (ProxFlow) with the efficient parametric max-flow algorithm proposed by Gallo et al. (1989) and a simplified version of the latter proposed by Babenko and Goldberg (2006). We refer to these two algorithms as GGT and SIMP respectively. The benchmark is established on the same data sets as those already used in the experimental section of the paper, namely: (1) three data sets built from overcomplete bases of discrete cosine transforms (DCT), with respectively 10^4, 10^5 and 10^6 variables, and (2) images used for the background subtraction task, composed of 57600 pixels. For GGT and SIMP, we use the `paraF` software which is a `C++` parametric max-flow implementation available at `http://www.avglab.com/andrew/soft.html`. Experiments were conducted on

27. As a brief reminder, it is defined as the function that maps the vector $\mathbf{u}$ in $\mathbb{R}^p$ to the (unique, by strong convexity) solution of Equation (3).

Algorithm 3 FISTA procedure to solve problem (18).

Inputs: initial $\mathbf{w}_{(0)} \in \mathbb{R}^p$, $\Omega, \lambda > 0, \varepsilon_{\text{gap}} > 0$ (precision for the duality gap).
Parameters: $\nu > 1, L_0 > 0$.
Outputs: solution $\mathbf{w}$.
Initialization: $\mathbf{y}_{(1)} = \mathbf{w}_{(0)}, t_1 = 1, k = 1$.
while $\left\{ \texttt{computeDualityGap}\big(\mathbf{w}_{(k-1)}\big) > \varepsilon_{\text{gap}} \right\}$ **do**
  Find the smallest integer $s_k \geq 0$ such that
    $f(\text{prox}_{[\lambda\Omega]}(\mathbf{y}_{(k)})) \leq f(\mathbf{y}_{(k)}) + \Delta_{(k)}^\top \nabla f(\mathbf{y}_{(k)}) + \frac{\tilde{L}}{2}\|\Delta_{(k)}\|_2^2$,
    with $\tilde{L} \triangleq L_k \nu^{s_k}$ and $\Delta_{(k)} \triangleq \mathbf{y}_{(k)} - \text{prox}_{[\lambda\Omega]}(\mathbf{y}_{(k)})$.
  $L_k \leftarrow L_{k-1}\nu^{s_k}$.
  $\mathbf{w}_{(k)} \leftarrow \text{prox}_{[\lambda\Omega]}(\mathbf{y}_{(k)})$.
  $t_{k+1} \leftarrow (1 + \sqrt{1 + t_k^2})/2$.
  $\mathbf{y}_{(k+1)} \leftarrow \mathbf{w}_{(k)} + \frac{t_k - 1}{t_{k+1}}(\mathbf{w}_{(k)} - \mathbf{w}_{(k-1)})$.
  $k \leftarrow k + 1$.
end while
Return: $\mathbf{w} \leftarrow \mathbf{w}_{(k-1)}$.

Procedure `computeDualityGap`($\mathbf{w}$)

$\kappa \leftarrow -\rho^{-1}\nabla f(\mathbf{X}\mathbf{w})$, with $\rho \triangleq \max\left\{\lambda^{-1}\Omega^*(\mathbf{X}^\top \nabla f(\mathbf{X}\mathbf{w})), 1\right\}$.
Return: $f(\mathbf{X}\mathbf{w}) + \lambda\Omega(\mathbf{w}) + f^*(-\kappa)$.

a single-core 2.33 Ghz. We report in the following table the average execution time in seconds of each algorithm for 5 runs, as well as the statistics of the corresponding problems:

Number of variables p	10000	100000	1000000	57600
$\lvert V\rvert$	20000	200000	2000000	57600
$\lvert E\rvert$	110000	500000	11000000	579632
ProxFlow (in sec.)	**0.4**	**3.1**	**113.0**	**1.7**
GGT (in sec.)	2.4	26.0	525.0	16.7
SIMP (in sec.)	1.2	13.1	284.0	8.31

Although we provide the speed comparison for a single value of λ (the one used in the corresponding experiments of the paper), we observed that our approach consistently outperforms GGT and SIMP for values of λ corresponding to different regularization regimes.

References

R. K. Ahuja, T. L. Magnanti, and J. Orlin. *Network Flows*. Prentice Hall, 1993.

A. Argyriou, T. Evgeniou, and M. Pontil. Convex multi-task feature learning. *Machine Learning*, 73(3):243–272, 2008.

M. Babenko and A.V. Goldberg. Experimental evaluation of a parametric flow algorithm. Technical report, Microsoft Research, 2006. MSR-TR-2006-77.

F. Bach. High-dimensional non-linear variable selection through hierarchical kernel learning. Technical report, arXiv:0909.0844, 2009.

F. Bach. Structured sparsity-inducing norms through submodular functions. In *Advances in Neural Information Processing Systems*, 2010.

F. Bach, R. Jenatton, J. Mairal, and G. Obozinski. Convex optimization with sparsity-inducing norms. In S. Sra, S. Nowozin, and S. J. Wright, editors, *Optimization for Machine Learning*. MIT Press, 2011. To appear.

R. G. Baraniuk, V. Cevher, M. Duarte, and C. Hegde. Model-based compressive sensing. *IEEE Transactions on Information Theory*, 56(4):1982–2001, 2010.

A. Barron, J. Rissanen, and B. Yu. The minimum description length principle in coding and modeling. *IEEE Transactions on Information Theory*, 44(6):2743–2760, 1998.

A. Beck and M. Teboulle. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM Journal on Imaging Sciences*, 2(1):183–202, 2009.

D. P. Bertsekas. *Network Optimization: Continuous and Discrete Models*. Athena Scientific, 1998.

D. P. Bertsekas. *Nonlinear Programming*. Athena Scientific Belmont, 1999.

D. P. Bertsekas and J. N. Tsitsiklis. *Parallel and Distributed Computation: Numerical Methods*. Prentice Hall Inc., 1989.

P. Bickel, Y. Ritov, and A. Tsybakov. Simultaneous analysis of Lasso and Dantzig selector. *Annals of Statistics*, 37(4):1705–1732, 2009.

J. Bien, Y. Xu, and M. W. Mahoney. CUR from a sparse optimization viewpoint. In *Advances in Neural Information Processing Systems*, 2010.

J. M. Borwein and A. S. Lewis. *Convex Analysis and Nonlinear Optimization: Theory and Examples*. Springer, 2006.

S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends in Machine Learning*, 3(1):1–122, 2011.

S. P. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004.

Y. Boykov and V. Kolmogorov. An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(9):1124–1137, 2004.

P. Brucker. An O(n) algorithm for quadratic knapsack problems. *Operations Research Letters*, 3: 163–166, 1984.

T. T. Cai. Adaptive wavelet estimation: a block thresholding and oracle inequality approach. *Annals of Statistics*, pages 898–924, 1999.

V. Cehver, M. F. Duarte, C. Hedge, and R. G. Baraniuk. Sparse signal recovery using Markov random fields. In *Advances in Neural Information Processing Systems*, 2008.

A. Chambolle and J. Darbon. On total variation minimization and surface evolution using parametric maximal flows. *International Journal of Computer Vision*, 84(3), 2009.

S. S. Chen, D. L. Donoho, and M. A. Saunders. Atomic decomposition by basis pursuit. *SIAM Journal on Scientific Computing*, 20:33–61, 1999.

X. Chen, Q. Lin, S. Kim, J. Pena, J. G. Carbonell, and E. P. Xing. An efficient proximal-gradient method for single and multi-task regression with structured sparsity. Technical report, 2010. ArXiv:1005.4717v1.

B. V. Cherkassky and A. V. Goldberg. On implementing the push-relabel method for the maximum flow problem. *Algorithmica*, 19(4):390–410, 1997.

P. L. Combettes and J.-C. Pesquet. A proximal decomposition method for solving convex variational inverse problems. *Inverse Problems*, 24(27), 2008. Art. 065014.

P. L. Combettes and J.-C. Pesquet. Proximal splitting methods in signal processing. In *Fixed-Point Algorithms for Inverse Problems in Science and Engineering*. Springer, 2010.

D. L. Donoho and I. M. Johnstone. Adapting to unknown smoothness via wavelet shrinkage. *Journal of the American Statistical Association*, 90(432):1200–1224, 1995.

J. Duchi, S. Shalev-Shwartz, Y. Singer, and T. Chandra. Efficient projections onto the ℓ_1-ball for learning in high dimensions. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2008.

B. Efron, T. Hastie, I. Johnstone, and R. Tibshirani. Least angle regression. *Annals of Statistics*, 32 (2):407–499, 2004.

M. Elad and M. Aharon. Image denoising via sparse and redundant representations over learned dictionaries. *IEEE Transactions on Image Processing*, 54(12):3736–3745, December 2006.

L. R. Ford and D. R. Fulkerson. Maximal flow through a network. *Canadian Journal of Mathematics*, 8(3):399–404, 1956.

J. Friedman, T. Hastie, and R. Tibshirani. A note on the group Lasso and a sparse group Lasso. Technical report, Preprint arXiv:1001.0736, 2010.

S. Fujishige. *Submodular Functions and Optimization*. Elsevier, 2005.

G. Gallo, M. E. Grigoriadis, and R. E. Tarjan. A fast parametric maximum flow algorithm and applications. *SIAM J. Comput.*, 18:30–55, 1989.

P. Garrigues and B. Olshausen. Group sparse coding with a Laplacian scale mixture prior. In *Advances in Neural Information Processing Systems*, 2010.

A. V. Goldberg and R. E. Tarjan. A new approach to the maximum flow problem. In *Proc. of ACM Symposium on Theory of Computing*, pages 136–146, 1986.

D. Goldfarg and S. Ma. Fast multiple splitting algorithms for convex optimization. Technical report, 2009. Preprint arXiv:0912.4570v1.

H. Groenevelt. Two algorithms for maximizing a separable concave function over a polymatroid feasible region. *Europeran Journal of Operations Research*, pages 227–236, 1991.

L. He and L. Carin. Exploiting structure in wavelet-based Bayesian compressive sensing. *IEEE Transactions on Signal Processing*, 57(9):3488–3497, 2009.

D. S. Hochbaum and S. P. Hong. About strongly polynomial time algorithms for quadratic optimization over submodular constraints. *Mathematical Programming*, 69(1):269–309, 1995.

H. Hoefling. A path algorithm for the fused lasso signal approximator. *Journal of Computational and Graphical Statistics*, 19(4):984–1006, 2010.

R. A. Horn and C. R. Johnson. *Matrix analysis*. Cambridge University Press, 1990.

J. Huang and T. Zhang. The benefit of group sparsity. *Annals of Statistics*, 38(4):1978–2004, 2010.

J. Huang, Z. Zhang, and D. Metaxas. Learning with structured sparsity. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2009.

A. Hyvärinen, P. Hoyer, and M. Inki. Topographic independent component analysis. *Neural Computation*, 13(7):1527–1558, 2001.

L. Jacob, G. Obozinski, and J.-P. Vert. Group Lasso with overlap and graph Lasso. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2009.

R. Jenatton, J-Y. Audibert, and F. Bach. Structured variable selection with sparsity-inducing norms. Technical report, 2009. Preprint arXiv:0904.3523v3.

R. Jenatton, J. Mairal, G. Obozinski, and F. Bach. Proximal methods for sparse hierarchical dictionary learning. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2010a.

R. Jenatton, G. Obozinski, and F. Bach. Structured sparse principal component analysis. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2010b.

R. Jenatton, J. Mairal, G. Obozinski, and F. Bach. Proximal methods for hierarchical sparse coding. *Journal of Machine Learning Research*, 12:2297–2334, 2011.

K. Kavukcuoglu, M. Ranzato, R. Fergus, and Y. LeCun. Learning invariant features through topographic filter maps. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2009.

S. Kim and E. P. Xing. Tree-guided group Lasso for multi-task regression with structured sparsity. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2010.

N. Maculan and J. R. G. Galdino de Paula. A linear-time median-finding algorithm for projecting a vector on the simplex of R^n. *Operations Research Letters*, 8(4):219–222, 1989.

M. W. Mahoney and P. Drineas. CUR matrix decompositions for improved data analysis. *Proceedings of the National Academy of Sciences*, 106(3):697, 2009.

J. Mairal, F. Bach, J. Ponce, G. Sapiro, and A. Zisserman. Non-local sparse models for image restoration. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2009.

J. Mairal, F. Bach, J. Ponce, and G. Sapiro. Online learning for matrix factorization and sparse coding. *Journal of Machine Learning Research*, 11:19–60, 2010a.

J. Mairal, R. Jenatton, G. Obozinski, and F. Bach. Network flow algorithms for structured sparsity. In *Advances in Neural Information Processing Systems*, 2010b.

S. Mallat. *A Wavelet Tour of Signal Processing, Second Edition*. Academic Press, New York, September 1999.

C. A. Micchelli, J. M. Morales, and M. Pontil. A family of penalty functions for structured sparsity. In *Advances in Neural Information Processing Systems*, 2010.

J. J. Moreau. Fonctions convexes duales et points proximaux dans un espace Hilbertien. *Compte Rendus de l'Académie des Sciencs, Paris, Série A, Mathématiques*, 255:2897–2899, 1962.

D. Needell and J. A. Tropp. CoSaMP: Iterative signal recovery from incomplete and inaccurate samples. *Applied and Computational Harmonic Analysis*, 26(3):301–321, 2009.

S. Negahban, P. Ravikumar, M. J. Wainwright, and B. Yu. A unified framework for high-dimensional analysis of M-estimators with decomposable regularizers. In *Advances in Neural Information Processing Systems*, 2009.

Y. Nesterov. Smooth minimization of non-smooth functions. *Mathematical Programming*, 103(1): 127–152, 2005.

Y. Nesterov. Gradient methods for minimizing composite objective function. Technical report, Center for Operations Research and Econometrics (CORE), Catholic University of Louvain, 2007.

G. Obozinski, B. Taskar, and M. I. Jordan. Joint covariate selection and joint subspace selection for multiple classification problems. *Statistics and Computing*, 20(2):231–252, 2010.

B. A. Olshausen and D. J. Field. Emergence of simple-cell receptive field properties by learning a sparse code for natural images. *Nature*, 381:607–609, 1996.

Z. Qin and D. Goldfarb. Structured sparsity via alternating directions methods. Technical report, 2011. preprint ArXiv:1105.0728.

A. Rakotomamonjy, F. Bach, S. Canu, and Y. Grandvalet. SimpleMKL. *Journal of Machine Learning Research*, 9:2491–2521, 2008.

V. Roth and B. Fischer. The group-Lasso for generalized linear models: uniqueness of solutions and efficient algorithms. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2008.

M. Schmidt and K. Murphy. Convex structure learning in log-linear models: Beyond pairwise potentials. In *Proceedings of the International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2010.

M. Schmidt, N. Le Roux, and F. Bach. Convergence rates of inexact proximal-gradient methods for convex optimization. In *Advances in Neural Information Processing Systems*, 2011. to appear, preprint ArXiv:1109.2415v1.

J. Shawe-Taylor and N. Cristianini. *Kernel Methods for Pattern Analysis*. Cambridge University Press, 2004.

P. Sprechmann, I. Ramirez, G. Sapiro, and Y. C. Eldar. Collaborative hierarchical sparse modeling. Technical report, 2010. Preprint arXiv:1003.0400v1.

M. Stojnic, F. Parvaresh, and B. Hassibi. On the reconstruction of block-sparse signals with an optimal number of measurements. *IEEE Transactions on Signal Processing*, 57(8):3075–3085, 2009.

R. Tibshirani. Regression shrinkage and selection via the Lasso. *Journal of the Royal Statistical Society: Series B*, 58(1):267–288, 1996.

R. Tomioka, T. Suzuki, and M. Sugiyama. Augmented Lagrangian methods for learning, selecting and combining features. In S. Sra, S. Nowozin, and S. J. Wright, editors, *Optimization for Machine Learning*. MIT Press, 2011. To appear.

P. Tseng. Convergence of a block coordinate descent method for nondifferentiable minimization. *Journal of optimization theory and applications*, 109(3):475–494, 2001.

B. A. Turlach, W. N. Venables, and S. J. Wright. Simultaneous variable selection. *Technometrics*, 47(3):349–363, 2005.

M. J. Wainwright. Sharp thresholds for noisy and high-dimensional recovery of sparsity using ℓ_1-constrained quadratic programming (Lasso). *IEEE Transactions on Information Theory*, 55(5): 2183–2202, May 2009.

J. Wright, A. Y. Yang, A. Ganesh, S. S. Sastry, and Y. Ma. Robust face recognition via sparse representation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(2):210–227, 2009a.

S. Wright, R. Nowak, and M. Figueiredo. Sparse reconstruction by separable approximation. *IEEE Transactions on Signal Processing*, 57(7):2479–2493, 2009b.

M. Yuan and Y. Lin. Model selection and estimation in regression with grouped variables. *Journal of the Royal Statistical Society: Series B*, 68:49–67, 2006.

X. Zhang, M. Burger, and S. Osher. A unified primal-dual algorithm framework based on Bregman iteration. *Journal of Scientific Computing*, 46(1):20–46, 2011.

P. Zhao, G. Rocha, and B. Yu. The composite absolute penalties family for grouped and hierarchical variable selection. *Annals of Statistics*, 37(6A):3468–3497, 2009.

Journal of Machine Learning Research 12 (2011) 2721-2748 Submitted 8/09; Revised 4/11; Published 9/11

Large Margin Hierarchical Classification with Mutually Exclusive Class Membership

Huixin Wang HXWANG@STAT.UMN.EDU
Xiaotong Shen XSHEN@STAT.UMN.EDU
School of Statistics
University of Minnesota
Minneapolis, MN 55455

Wei Pan WEIP@BIOSTAT.UMN.EDU
Division of Biostatistics
University of Minnesota
Minneapolis, MN 55455

Editor: Nicolo-Cesa Bianchi

Abstract

In hierarchical classification, class labels are structured, that is each label value corresponds to one non-root node in a tree, where the inter-class relationship for classification is specified by directed paths of the tree. In such a situation, the focus has been on how to leverage the inter-class relationship to enhance the performance of flat classification, which ignores such dependency. This is critical when the number of classes becomes large relative to the sample size. This paper considers single-path or partial-path hierarchical classification, where only one path is permitted from the root to a leaf node. A large margin method is introduced based on a new concept of generalized margins with respect to hierarchy. For implementation, we consider support vector machines and ψ-learning. Numerical and theoretical analyses suggest that the proposed method achieves the desired objective and compares favorably against strong competitors in the literature, including its flat counterparts. Finally, an application to gene function prediction is discussed.

Keywords: difference convex programming, gene function annotation, margins, multi-class classification, structured learning

1. Introduction

In many applications, knowledge is organized and explored in a hierarchical fashion. For instance, in one of the central problems in modern biomedical research—gene function prediction, biological functions of genes are often organized by a hierarchical annotation system such as MIPS (the Munich Information Center for Protein Sequences, Mewes et al., 2002)for yeast S. *cerevisiae*. MIPS is structured hierarchically, with upper-level functional categories describing more general information concerning biological functions of genes, while low-level ones refer to more specific and detailed functional categories. A hierarchy of this sort presents the current available knowledge. To predict unknown gene functions, a gene is classified, through some predictors, into one or more gene functional categories in the hierarchy of MIPS, forming novel hypotheses for confirmatory biological experiments (Hughes et al., 2000). Classification like this is called hierarchical classification, which has been widely used in webpage classification and document categorization. Hierarchical classification involves inter-class dependencies specified by a prespecified hierarchy, which is unlike

multiclass classification where class membership is mutually exclusive for all classes. The primary objective of hierarchical classification is leveraging inter-class relationships to enhance multiclass classification ignoring such dependencies, known as flat classification. This is particularly critical in high-dimensional problems with a large number of classes in classification. To achieve the desired objective, this paper develops a large margin approach for single-path or partial-path hierarchical classification with hierarchy defined by a tree.

Hierarchical classification, an important subject which has not yet received much attention, can be thought of as nested classification within the framework of multiclass classification. One major challenge is how to formulate a loosely defined hierarchical structure into classification to achieve higher generalization performance, which, otherwise, is impossible for flat classification, especially in a high-dimensional situation. Three major approaches have been proposed in the literature. The first is the so called "flat approach", which ignores the hierarchical structure. Recent studies suggest that higher classification accuracy results can be realized by incorporating the hierarchical structure (Dekel et al., 2004). Relevant references can be found in Yang and Liu (1999) for nearest neighbor, Lewis (1998) for naive Bayes, Joachims (1998) for support vector machines (SVM, Boser et al., 1992; Vapnik, 1998), among others. The second is the sequential approach, where a multiclass classifier is trained locally at each parent node of the hierarchy. As a result, the classifier may be not well trained due to a small training sample locally and lack of global comparisons. Further investigations are necessary with regard to how to use the given hierarchy in classification to improve the predictive performance, as noted in Dekel et al. (2004) and Cesa-Bianchi et al. (2006). The third is the promising structured approach, which recognizes the importance of a hierarchical structure in classification. Shahbaba and Neal (2007) proposed a Bayesian method through a constrained hierarchical prior and a Markov Chain Monte Carlo implementation. Cai and Hofmann (2004) and Rousu et al. (2006) employed structured linear and kernel representations and loss functions defined by a tree, together with loss-weighted multiclass SVM, whereas Dekel et al. (2004) developed a batch and on-line version of loss-weighted hierarchical SVM, and Cesa-Bianchi et al. (2006) developed sequential training based SVM with certain hierarchical loss functions. The structured approach uses a weighted loss defined by a hierarchy, such as the symmetric difference loss and a sub-tree H-loss, see, for instance, Cesa-Bianchi et al. (2006), as opposed to the conventional 0-1 loss, then maximizes the loss-weighted margins for a multiclass SVM, as described in Lin et al. (2002). Ensembles of nested dichotomies in Dong et al. (2005) and Zimek et al. (2008) have achieved good performance. Despite progress, issues remain with respect to how to fully take into account a hierarchical structure and to what role the hierarchy plays.

To meet the challenge, this article develops a large margin method for hierarchical classification, based on a new concept of structured functional and geometric margins defined for each node of the hierarchy, which differs from the concept of the loss-weighted margins in structured prediction. This concept of margins with respect to hierarchy is designed to account for inter-class dependencies in classification. As a result, the complexity of the classification problem reduces, translating into higher generalization accuracy of classification. Our theory describes when this will occur, depending on the structure of a tree hierarchy. In contrast to existing approaches, the proposed method trains a classifier globally while making sequential nested partitions of classification regions. The proposed method is implemented for support vector machines (SVM, Boser et al., 1992) and ψ-learning (Shen et al., 2003) through quadratic and difference convex (DC) programming.

To examine the proposed method's generalization performance, we perform simulation studies. They indicate that the proposed method achieves higher performance than three strong competitors.

A theoretical investigation confirms that the empirical performance is indeed attributed to a reduced size of the function space for classification, as measured by the metric entropy, through effective use of a hierarchical structure. In fact, stronger inter-class relations tend to lead to better performance over its flat counterpart. In conclusion, both the numerical and theoretical results suggest that a tree hierarchical structure has been incorporated into classification for generalization.

This article is organized as follows. Section 2 formulates the problem of hierarchical classification. Section 3 introduces the proposed method and develops computational tools. Section 4 performs simulation studies and presents an application of the proposed method in gene function prediction. Section 5 is devoted to theoretical investigation of the proposed method and to the study of the role of a hierarchical structure in classification. Section 6 discusses the method, followed by technical details in the Appendix.

2. Single-path and Partial-path Hierarchical Classification

In single-path or partial-path hierarchical classification, input $\boldsymbol{X} = (X_1, \cdots, X_q) \in S \subset \mathbb{R}^q$ is a vector of q covariates, and we code output $Y \in \{1, \cdots, K\}$, corresponding to non-root nodes $\{1, \cdots, K\}$ in a rooted tree $\mathcal{H}$, a graph with nodes connected by directed paths from the root 0, where directed edge $i \to j$ specifies a parent-child relationship from i to j. Here Y is structured in that $i \to j$ in $\mathcal{H}$ induces a subset relation between the corresponding classes i and j in classification, that is, the classification region of class j is a subset of that of class i. As a result, direct and indirect relations among nodes over $\mathcal{H}$ impose an inter-class relationship among K classes in classification.

Before proceeding, we introduce some notations for a tree $\mathcal{H}$ with k leaves and $(K-k)$ non-leaf-nodes, where a non-leaf node is an ancestor of a leaf one. Denote by $|\mathcal{H}|$ the size of $\mathcal{H}$. For each $t \in \{1, \cdots, K\}$, define $par(t)$, $chi(t)$ $sib(t)$, $anc(t)$ and $sub(t)$ to be sets of its parent(s) (immediate ancestor), its children (immediate offsprings), its siblings (nodes sharing the same parent with node t), its ancestors (immediate or remote), and the subtree rooted from t, respectively. Throughout this paper, $par(t)$, $chi(t)$ and $sib(t)$ are allowed to be empty. Assume, without loss of generality, that $|par(t)| = 1$ for non-root node t because multiple parents are not permitted for a tree. Also we define $\mathcal{L}$ to be the set of leaves of $\mathcal{H}$.

To classify $\boldsymbol{x}$, a decision function vector $\boldsymbol{f} = (f_1, \cdots, f_K) \in \mathcal{F} = \prod_{j=1}^{K} \mathcal{F}_j$ is introduced, where $f_j(\boldsymbol{x})$; $j = 1, \cdots, K$, mapping from $\mathbb{R}^q$ onto $\mathbb{R}^1$, represents class j and mimics $P(Y = j | \boldsymbol{X} = \boldsymbol{x})$. Then $\boldsymbol{f}$ is estimated through a training sample $Z_i = (\boldsymbol{X}_i, Y_i)_{i=1}^n$, independent and identically distributed according to an unknown probability $P(\boldsymbol{x}, y)$. To assign $\boldsymbol{x}$, we introduce a top-down decision rule $d^H(\boldsymbol{f}(\boldsymbol{x}))$ with respect to $\mathcal{H}$ through $\boldsymbol{f}$. From the top to the bottom, we go through each node j and assign $\boldsymbol{x}$ to one of its children $l = \text{argmax}_{t \in chi(j)} f_t(\boldsymbol{x})$ having the highest value among f_t's for $t \in chi(j)$ when $j \notin \mathcal{L}$, and assign $\boldsymbol{x}$ to j otherwise.

This top-down rule is sequential, and yields mutually exclusive membership for sibling classes. In particular, for each parent j, $chi(j)$ gives a partition of the classification region of parent class j. This permits an observation staying at a parent when one child of the parent is defined as itself, see, for example, the node labeled 03.01 in Figure 3, which is a case of partial-path hierarchical classification.

Finally, a classifier is constructed through $d^H(\cdot)$ to have small generalization error $El_{0-1}(Y, d^H(\boldsymbol{f}(\boldsymbol{X}))$, with $l_{0-1}(Y, d^H(\boldsymbol{f}(\boldsymbol{X})) = I(Y \neq d^H(\boldsymbol{f}(\boldsymbol{X}))$ the 0-1 hierarchical loss.

3. Proposed Method

In the existing literature on hierarchical classification, the margins are defined by the conventional unstructured margins for multiclass classification, for instance, the loss-weighted hierarchical SVM of Cai and Hofmann (2004), denoted as HSVM$_c$. For unstructured margins in classification, a certain number of pairwise comparisons is required, which is the same as conventional multiclass classification. In what follows, we propose a new framework using a given hierarchy to define margins, leading to a reduced number of pairwise comparisons for hierarchical classification.

3.1 Margins with Respect to $\mathcal{H}$

We first explore a connection between classification and function comparisons, based on the concept of generalized functional margins with respect to a hierarchy is introduced. Over a hierarchy $\mathcal{H}$, the top-down rule $d^H(\boldsymbol{f}(\boldsymbol{x}))$ is employed for classification. To classify, comparing some components of $\boldsymbol{f}$ at certain relevant nodes in $\mathcal{H}$ is necessary, which is in a parallel fashion as in multiclass classification. Consider leaf node 4 in the tree $\mathcal{H}$ described in Figure 2 (c). There $f_4 - f_3$ and $f_6 - f_5$ need to be compared against 0 to classify at node 4 through the top-down rule, that is, $\min(f_4 - f_3, f_6 - f_5)$ is less than 0 or not, which leads to our margin definition for $(\boldsymbol{x}, y = 4)$ $U(\boldsymbol{f}(\boldsymbol{x}), y = 4) = \min(f_4 - f_3, f_6 - f_5)$. More generally, we define set $U(\boldsymbol{f}(\boldsymbol{x}), y)$, for $y \in \{1, \cdots, K\}$ to be $\{f_t - f_j : j \in sib(t), t \in anc(y) \cup \{y\}\} = \{u_{y,1}, u_{y,2}, \cdots, u_{y,k_y}\}$ with k_y elements. This set compares any class t against sibling classes defined by $sib(t)$ for y and any of its ancestors t, permitting hierarchical classification at any location of $\mathcal{H}$ and generating a single-path or partial-path from the root to the node corresponding to class y.

For classification evaluation, we define the generalized functional margin with respect to $\mathcal{H}$ for $(\boldsymbol{x}, y)$ as $u_{min}(\boldsymbol{f}(\boldsymbol{x}), y) = \min\{u_{y,j} : u_{y,j} \in U(\boldsymbol{f}(\boldsymbol{x}), y)\}$. In light of the result of Lemma 1, this quantity is directly related to the generalization error, which summarizes the overall error in hierarchical classification as the 0-1 loss in binary classification. That is, a classification error occurs if and only if $u_{min}(\boldsymbol{f}(\boldsymbol{x}), y) < 0$. Moreover, *this definition reduces to that of multiclass margin classification of Liu and Shen (2006) when no hierarchical structure is imposed*. In contrast to the definition of multiclass classification, the number of comparisons required for classification over a tree $\mathcal{H}$ is usually smaller, owing to the fact that only siblings need to be compared through the top-down rule, as opposed to comparisons of all pairs of classes in multiclass classification.

Lemma 1 establishes a key connection between the generalization error and our definition of $u_{min}(f(X), Y)$.

Lemma 1 *With $I(\cdot)$ denoting the indicator function,*

$$GE(d) = El_{0-1}(Y, d(\boldsymbol{X})) \equiv EI(Y \neq d(\boldsymbol{X})) = EI(u_{min}(f(X), Y) < 0),$$

where l_{0-1} is the 0-1 loss in hierarchical classification, and $I(\cdot)$ is the indicator function.

This lemma says that a classification error occurs for decision function $\boldsymbol{f}$ and an observation $(\boldsymbol{x}, y)$, if and only if the functional margin $u_{min}(\boldsymbol{f}(\boldsymbol{x}), y)$ is negative.

3.2 Cost Function and Geometric Margin

To achieve our objective of constructing classifier $d^H(\hat{\boldsymbol{f}}(\boldsymbol{x}))$ having small generalization error, we construct a cost function to yield an estimate $\hat{\boldsymbol{f}}$ for $d^H(\hat{\boldsymbol{f}}(\boldsymbol{x}))$. Ideally, one may minimize the

empirical generalization error $n^{-1}\sum_{i=1}^{n} I(u_{min}(f(X_i),Y_i)<0)$ based on $(\boldsymbol{X}_i,Y_i)_{i=1}^{n}$. However, it is computationally infeasible because of discontinuity of $I(\cdot)$. For this reason, we replace $I(\cdot)$ by a surrogate loss $v(\cdot)$ to use the existing two-class surrogate losses in hierarchical classification. In addition to computational benefits, certain loss functions $v(\cdot)$ may also lead to desirable large margin properties (Zhu and Hastie, 2005). Given functional margin $u = u_{min}(\boldsymbol{f}(\boldsymbol{x}),y)$, we say that a loss $v(\cdot)$ is a margin loss if it can be written as a function of u. Moreover, it is a large margin if $v(u)$ is nonincreasing in u. Most importantly, $v(u_{min}(\boldsymbol{f}(\boldsymbol{x}),y))$ yields Fisher-consistency in hierarchical classification, which constitutes a basis of studying the generalization error in Section 5. Note that in the two-class case a number of margin losses have been proposed. Convex margin losses are the hinge loss $v(u)=(1-u)_+$ for SVM and the logistic loss $v(u)=\log(1+e^{-u})$ for logistic regression (Zhu and Hastie, 2005). Nonconvex large margin losses include, for example ψ-loss $v(u)=\psi(u)$ for ψ-learning, with $\psi(u)=1-\text{sign}(u)$ and $\text{sign}(u)=I(u>0)$, if $u\geq 1$ or $u<0$, and $1-u$ otherwise (Shen et al., 2003).

Placing a margin loss $v(\cdot)$ in the framework of penalization, we propose our cost function for hierarchical classification:

$$s(\boldsymbol{f}) = C\sum_{i=1}^{n} v(u_{min}(\boldsymbol{f}(\boldsymbol{x}_i),y_i)) + J(\boldsymbol{f}), \tag{1}$$

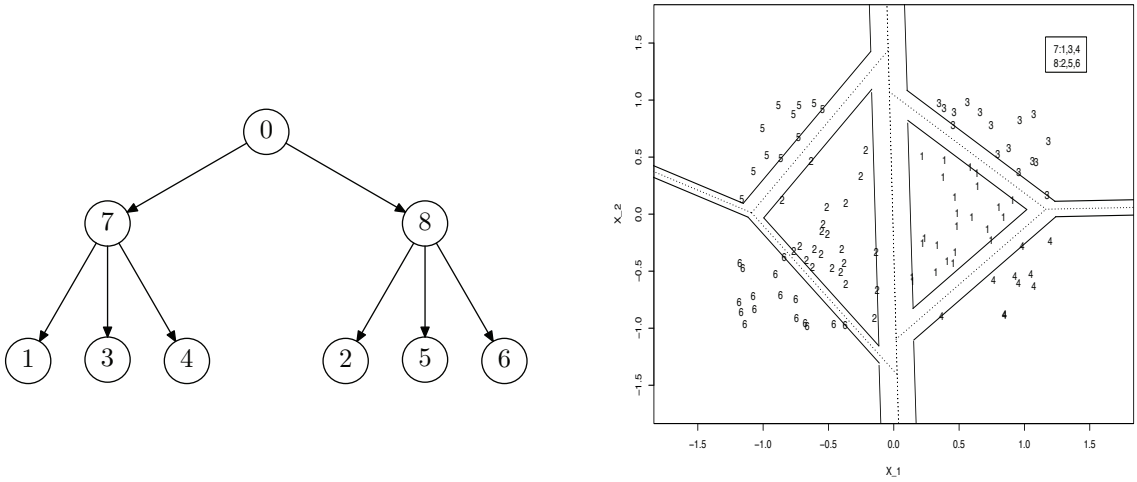
subject to sum to zero constraints $\sum_{\{t\in sib(j)\cup\{j\}\}} f_t(\boldsymbol{x})=0;\ \forall j=1\cdots,K, sib(j)\neq\varnothing, \boldsymbol{x}\in S$, the domain of X_1, for removing redundancy among the components of $\boldsymbol{f}$. For example, for the tree $\mathcal{H}$ in Figure 2 (c), three constraints are imposed: $f_1+f_2=0$, $f_3+f_4=0$ and $f_5+f_6=0$, for three pairs of siblings. In (1), penalty $J(\boldsymbol{f})$ is the inverse geometric margin to be introduced, and $C>0$ is a tuning parameter regularizing the trade-off between minimizing $J(\boldsymbol{f})$ and minimizing training error. Minimizing (1) with respect to $\boldsymbol{f}\in\mathcal{F}$, a candidate function space, yields an estimate $\hat{\boldsymbol{f}}$, thus classifier $d^H(\hat{\boldsymbol{f}}(\boldsymbol{x}))$. Note that *(1) reduces to that of multiclass margin classification of Liu and Shen (2006) when no hierarchical structure is specified*.

To introduce the geometric margin with respect to $\mathcal{H}$ in the L_2-norm, (with other norms applied similarly), consider a generic vector of functions $\boldsymbol{f}$: $f_j(\boldsymbol{x})=\boldsymbol{w}_j^T\tilde{\boldsymbol{x}}+b_j;\ j=1,\cdots,K$, with $\tilde{\boldsymbol{x}}=\boldsymbol{x}$ and $\tilde{\boldsymbol{x}}=(\mathcal{K}(\boldsymbol{x}_1,\cdot),\cdots,\mathcal{K}(\boldsymbol{x}_n,\cdot))^T$ for linear and kernel learning. The geometric margin is defined as $\min_{\{(t,j):t\in sib(j)\}}\gamma_{j,t}$, where $\gamma_{j,t}=\frac{2}{\|\boldsymbol{w}_j-\boldsymbol{w}_t\|^2_{\mathcal{K}}}$ is the usual separation margin defined for classes j versus $t\in sib(j)$, representing the vertical distance between two parallel hyperplanes $f_j-f_t=\pm 1$ (Shen and Wang, 2007). Here $\|\boldsymbol{w}_j\|^2_{\mathcal{K}}$ is $\|\boldsymbol{w}_j\|^2$ in the linear case and is $\boldsymbol{w}_j^T\mathcal{K}\boldsymbol{w}_j$ in the kernel case with $\mathcal{K}$ being an $n\times n$ kernel matrix. Note that the other form of the margin in the L_p-norm (with $1\leq p\leq\infty$) can be defined similarly. Ideally, $J(\boldsymbol{f})$ is $\max_{\{(t,j):t\in sib(j)\}}\gamma_{j,t}^{-1}=\max_{\{(t,j):t\in sib(j)\}}\frac{\|\boldsymbol{w}_j-\boldsymbol{w}_t\|^2}{2}$, the inverse of the geometric margin. However, it is less tractable numerically. Practically, we work with its upper bound $J(\boldsymbol{f})=\frac{1}{2}\sum_{j=1}^{K}\|\boldsymbol{w}_j\|^2_{\mathcal{K}}$ instead.

For hierarchical classification, (1) yields different classifiers with different choices of margin loss $v(\cdot)$. Specifically, (1) covers multiclass SVM and ψ-learning of Liu and Shen (2006), with equal cost when all the leaf nodes share the same parent—the root, which are called SVM and ψ-learning in what follows.

3.3 Classification and Hierarchy $\mathcal{H}$

The hierarchical structure specified by $\mathcal{H}$ is summarized as the direct parent-child relation and the associated indirect relations, for classification. They are integrated into our framework. Whereas



(a) Hierarchical structure

(b) Geometric margin

Figure 1: Plot of generalized geometric margin with respect to $\mathcal{H}$ in (b), defined by a tree in (a). Classification processes sequentially with a partition of classes 7 and 8 at the top level, and a further partition of class 7 into classes 1,3 and 4, and that of class 8 into classes 3,5 and 6, where classification boundaries are displayed by dotted lines. Geometric margin is defined as the minimal vertical distances between seven pairs of solid parallel lines, representing separations between classes 7 and 8, 2 and 5, 2 and 6, 5 and 6, 1 and 3, 1 and 4, and 3 and 4.

the top-down rule is specified by $\mathcal{H}$, $u_{min}(\boldsymbol{f}(\boldsymbol{x}), y)$ captures the relations through (1). As a result, a problem's complexity is reduced when classification is restricted to $\mathcal{H}$, leading to higher generalization accuracy. This aspect will be confirmed by the numerical results in Section 4, and by a comparison of the generalization errors between hierarchical SVM (HSVM) and hierarchical ψ-learning (HPSI) against their flat counterparts—SVM and ψ-learning in Section 5.

3.4 Minimization

We implement (1) in a generic form: $f_j(\boldsymbol{x}) = \boldsymbol{w}_j^T \tilde{\boldsymbol{x}} + b_j;\ j = 1, \cdots, K$. Note that the sum-to-zero constraints may be infinite, which occurs when the domain of x has infinitely many values. To overcome this difficulty, we derive Theorem 1, which says that reinforcement of the sum-to-zero constraints for (1) suffices at the observed data instead of all possible x-values.

Theorem 1 *Assume that* $\{\tilde{\boldsymbol{x}}_1, \tilde{\boldsymbol{x}}_2, \cdots, \tilde{\boldsymbol{x}}_n\}$ *spans* $\mathbb{R}^q$. *Then, for* $j = 1, \cdots, K$, *minimizing (1) subject to* $\sum_{\{t: t \in sib(j) \cup \{j\}\}} f_j(\boldsymbol{x}) = 0; \forall j = 1 \cdots, K, sib(j) \neq \varnothing, \boldsymbol{x} \in S$, *is equivalent to minimizing (1) subject to* $\sum_{\{t: t \in sib(j) \cup \{j\}\}} f_j(\boldsymbol{x}_i) = 0; \forall j = 1 \cdots, K, sib(j) \neq \varnothing, i = 1, \cdots, n$.

Based on Theorem 1, minimizing (1) is equivalent to

$$\text{minimizing} s(\boldsymbol{f}) = \frac{1}{2}\sum_{j=1}^{K}\|\boldsymbol{w}_j\|^2 + C\sum_{i=1}^{n} v(u_{min}(\boldsymbol{f}(\boldsymbol{x}_i), y_i)), \tag{2}$$

subject to $\sum_{\{t:t\in sib(j)\cup\{j\}\}} f_t(\boldsymbol{x}_i) = 0$; $i = 1,\cdots,n$, $j = 1,\cdots,K$, $sib(j) \neq \varnothing$.

Subsequently, we work with (2), where the proposed classifiers are denoted by HSVM and HPSI when $v(u) = (1-u)_+$ and $v(u) = \psi(u)$, respectively. In the first case, HSVM is solved by quadratic programming (QP), see Appendix B. In the second case, (2) for HPSI is solved by DC programming, to be described next.

For HPSI, we decompose $s(\boldsymbol{f})$ in (2) with $v(u) = \psi(u)$ into a difference of two convex functions: $s(\boldsymbol{f}) = s_1(\boldsymbol{f}) - s_2(\boldsymbol{f})$, where $s_1(\boldsymbol{f}) = \frac{1}{2}\sum_{j=1}^{K}\|\boldsymbol{w}_j\|^2 + C\sum_{i=1}^{n}\psi_1(u_{min}(\boldsymbol{f}(\boldsymbol{x}_i), y_i))$ and $s_2(\boldsymbol{f}) = C\sum_{i=1}^{n}\psi_2(u_{min}(\boldsymbol{f}(\boldsymbol{x}_i), y_i))$, derived from a DC decomposition of $\psi = \psi_1 - \psi_2$, with $\psi_1(u) = (1-u)_+$ and $\psi_2(u) = (-u)_+$. Through our DC decomposition, a sequence of upper approximations of $s(\boldsymbol{f})$ $s_1(\boldsymbol{f}) - \langle \boldsymbol{f} - \hat{\boldsymbol{f}}^{(m-1)}, \nabla s_2(\hat{\boldsymbol{f}}^{(m-1)})\rangle_{\mathcal{K}}$ is constructed iteratively, where $\langle\cdot,\cdot\rangle_{\mathcal{K}}$ is the inner product with respect to kernel $\mathcal{K}$ and $\nabla s_2(\hat{\boldsymbol{f}}^{(m-1)})$ is a gradient vector of $s_2(\boldsymbol{f})$ at the solution $\hat{\boldsymbol{f}}^{(m-1)}$ at iteration $m-1$, defined as a sum of partial derivatives of s_2 over each observation, with $\nabla\psi_2(u) = 0$ when $u > 0$ and $\nabla\psi_2(u) = -1$ otherwise. Note that $s_1(\boldsymbol{f}) - \langle \boldsymbol{f} - \hat{\boldsymbol{f}}^{(m)}, \nabla s_2(\hat{\boldsymbol{f}}^{(m)})\rangle_{\mathcal{K}}$ is a convex upper bound of $s(\boldsymbol{f})$ by convexity of s_2. Then the upper approximation $s_1(\boldsymbol{f}) - \langle \boldsymbol{f} - \hat{f}^{(m-1)}, \nabla s_2(\hat{\boldsymbol{f}}^{(m-1)})\rangle_{\mathcal{K}}$ is minimized to yield $\hat{\boldsymbol{f}}^{(m)}$. This is called a DC method for non-convex minimization in the global optimization literature (An and Tao, 1997).

To design our DC algorithm, starting from an initial value $\hat{\boldsymbol{f}}^{(0)}$, the solution of HSVM, we solve primal problems iteratively. At the mth iteration, we compute

$$\hat{\boldsymbol{f}}^{(m)} = \underset{\boldsymbol{f}}{\text{argmin}}(s_1(\boldsymbol{f}) - \langle \boldsymbol{f}, \nabla s_2(\hat{\boldsymbol{f}}^{(m-1)})\rangle_{\mathcal{K}}), \tag{3}$$

subject to $\sum_{\{t:t\in sib(j)\cup\{j\}\}} f_t(\boldsymbol{x}_i) = 0$; $i = 1,\cdots,n$, $j = 1,\cdots,K$, $sib(j) \neq \varnothing$, through QP and its dual form in Appendix B. The above iterative process continues until a termination criterion is met: $|s(\hat{\boldsymbol{f}}^{(m)}) - s(\hat{\boldsymbol{f}}^{(m-1)})| \leq \varepsilon$, where is $\varepsilon > 0$ is a prespecified tolerance precision. The final estimate $\hat{\boldsymbol{f}}$ is the best solution among $\hat{\boldsymbol{f}}^{(m)}$ over m.

The above algorithm terminates, and its speed of convergence is superlinear, by Theorem 3 of Liu et al. (2005) for ψ-learning. A DC algorithm usually leads to a good local solution even when it is not global (An and Tao, 1997). In our DC decomposition, s_2 can be thought of correcting the bias due to convexity imposed by s_1 that is the cost function of HSVM, which assures that a good local solution or a global solution can be realized. More importantly, an ε-global minimizer can be obtained when the algorithm is combined with the branch-and-bound method, as in Liu et al. (2005). Due to computational consideration, we shall not seek the exact global minimizer.

3.5 Evaluation Losses and Test Errors with Respect to Hierarchy

In hierarchical classification, three types of losses have been proposed for measuring a classifier's performance with respect to $\mathcal{H}$, as a generalization of the 0-1 loss in two-class classification. In addition to $l_{0-1}(Y, d(\boldsymbol{X}))$, there are the symmetric difference loss $l_\Delta(Y, d(\boldsymbol{X}))$ (Tsochantaridis et al., 2004) and the H-loss $l_H(Y, d(\boldsymbol{X}))$ (Rousu et al., 2006; Cesa-Bianchi et al., 2004). As in Tsochantaridis et al. (2004); Rousu et al. (2006); Cesa-Bianchi et al. (2004, 2006), we use the 0-1 loss, symmetric difference loss and H-losses as performance measurements for our examples. Given a classifier $d(\boldsymbol{x})$, $l_\Delta(Y, d(\boldsymbol{X}))$ is $|anc(Y) \triangle anc(d(\boldsymbol{X}))|$, where $\triangle$ denotes the symmetric difference

of two sets. Here $l_H(Y, d(\boldsymbol{X})) = c_j$, with j the highest node yielding the disagreement between Y and $d(\boldsymbol{X})$ in a tree, ignoring any errors occurring at lower levels. In other words, it penalizes the disagreement at a parent while tolerating subsequent errors at offsprings. Two common choices of c_j's have been suggested, leading to the subtree based H-loss l_{sub} and the siblings based H-loss l_{sib}:

$$c_j = |sub(j)|/K;\ j = 1, \ldots, K, \tag{4}$$

$$c_0 = 1,\ c_j = c_{par(j)}/|sib(j) \cup \{j\}|;\ j = 1, \ldots, K. \tag{5}$$

A classifier's generalization performance is measured by the test error, defined as

$$TE(\boldsymbol{f}) = n_{test}^{-1} \sum_{i=1}^{n_{test}} l(Y_i, d^H(\boldsymbol{f}(\boldsymbol{X}_i))), \tag{6}$$

where n_{test} is the size of a test sample, and l is one of the four evaluation losses: l_{0-1}, l_Δ, l_{sib} with c_j's defined by (4) and l_{sub} with c_j's defined by (5). The corresponding test errors are denoted as $TE_{0-1}, TE_\Delta, TE_{sib}$ and TE_{sub}.

4. Numerical Examples

The following discusses the numerical results from three simulated examples together with an application to gene functions classification.

4.1 Simulated Examples

This section applies HSVM and HPSI to three simulated examples, where they are compared against their flat counterparts—k-class SVM and k-class ψ-learning of Liu and Shen (2006), and two strong competitors—HSVM$_c$ and the sequential hierarchical SVM (SHSVM). For SHSVM, we train SVMs separately for each parent node, and use the top-down scheme to label the estimated classes. See Davies et al. (2007) for more details.

All numerical analyses are conducted in R version 2.1.1 for SVM, ψ-learning, HSVM, HPSI, HSVM$_c$ and SHSVM. In linear learning, $\mathcal{K}(x, y) = \langle x, y \rangle$. In Gaussian kernel learning, $\mathcal{K}(\boldsymbol{x}_1, \boldsymbol{x}_2) = \exp(-\|x_1 - x_2\|^2/\sigma^2)$ is used, where σ is the median of the inter-class distances between any two classes, see Jaakkola et al. (1999) for the binary case.

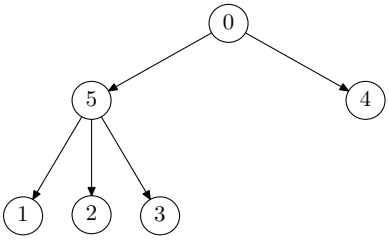
For comparison, we define the amount of improvement based on the test error. In simulated examples, the amount of improvement of a classifier is the percentage of improvement over SVM, in terms of the Bayesian regret:

$$\frac{(TE(\text{SVM}) - Bayes) - (TE(\cdot) - Bayes)}{(TE(\text{SVM}) - Bayes)},$$

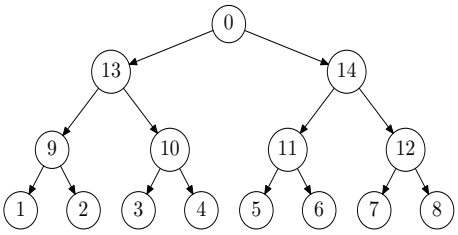
where $TE(\cdot)$ denotes the test error of a classifier, and *Bayes* denotes the Bayes error, which is the ideal optimal performance and serves as a benchmark for comparison. In a real data example where the Bayes rule is unavailable, the amount of improvement is

$$\frac{TE(\text{SVM}) - TE(\cdot)}{TE(\text{SVM})},$$

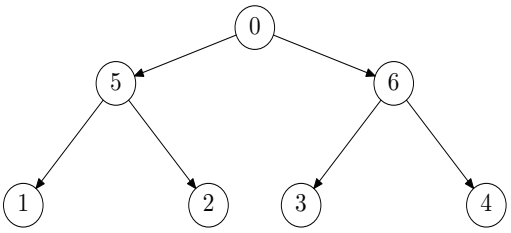
which may underestimate the actual percentage of improvement over SVM.



(a) Example 1



(b) Example 2



(c) Example 3

Figure 2: Hierarchies used in Examples 1, 2 and 3 of Section 4.1, described by four leave-nodes asymmetric tree in (a), and complete binary trees with depth $p = 3$ and $k = 2^p = 8$ leaf nodes in (b) and with depth $p = 2$ and $k = 4$ leaf nodes in (c), respectively.

In addition to test errors, F1-scores are computed for each classifier, which are between 0 and 1 and measure a classification (test)'s accuracy. A F1-score is defined as $2\frac{\rho \cdot r}{\rho + r}$, where the precision ρ is the number of correct results over the number of all results classified to a class by the trained classifier, and the recall r is the number of correct results divided by the number of instances with true label of a class. Specifically, for a given classifier, a F1-score is defined as a weighted average of F1-scores over all classes, weighted by the sample distribution.

For each classifier, we use one independent tuning sample of size n and one independent testing sample of 5×10^4, for tuning and testing. For tuning, the optimal C is obtained by minimizing the tuning error defined in (6) on 61 grid points: $C = 10^{l/10}; l = -30, -29, \cdots, 30$. Given the estimated optimal C, the test error in (6) is computed over the test sample.

Example 1. A random sample $(Y_i, \boldsymbol{X}_i = \{X_{i1}, X_{i2}\})_{i=1}^n$ is generated as follows. First, $\boldsymbol{X}_i \sim U^2(0,1)$ is sampled from the two-dimensional uniform distribution. Second, $Y_i \in \{1,2,3,4\}$ is sampled through conditional distributions: $P(Y_i = 1|X) = 0.17, P(Y_i = 2|X) = 0.17, P(Y_i = 3|X) = 0.17, P(Y_i = 4|X) = 0.49$. This generates a simple asymmetric distribution over a tree hierarchy with a four leaf-nodes as displayed in Figure 2(a).

Clearly, HSVM and HPSI outperform their competitors - HSVM$_c$, SHSVM and SVM under each the four evaluation losses in both linear and Gaussian kernel situations. Specifically, the improvement amount of HSVM over SVM varies from 1.5% to 3.1% in the linear case and 1.6% to 1.9% in the Gaussian kernel case, whereas that of HPSI ranges from 94.5% to 94.7% and 100.0%, respectively. By comparison, the amount of improvement of HSVM$_c$ is from 0.7% to 1.0% in

Linear					
Training Method	Test error				
	l_{0-1}	l_Δ	l_{sib}	l_{sub}	F1-score
SVM	0.545 (0.048)	0.527 (0.023)	0.521 (0.014)	0.521 (0.014)	0.328 (0.012)
ψ-learning	0.515 (0.032)	0.512 (0.015)	0.511 (0.009)	0.511 (0.009)	0.321 (0.011)
% of impro.	86.9%	86.9%	86.9%	86.8%	
HSVM$_c$	0.545 (0.044)	0.527 (0.022)	0.521 (0.012)	0.520 (0.012)	0.328 (0.015)
% of impro.	1.0%	1.0%	0.7%	0.9%	
SHSVM	0.659 (0.130)	0.580 (0.061)	0.554 (0.038)	0.554 (0.038)	0.248 (0.013)
% of impro.	-321.8%	-315.8%	-311.9%	-312.7%	
HSVM	0.545 (0.043)	0.526 (0.021)	0.520 (0.013)	0.520 (0.013)	0.327 (0.005)
% of impro.	1.5%	2.6%	3.0%	3.1%	
HPSI	**0.512(0.019)**	**0.511(0.009)**	**0.511(0.006)**	**0.511(0.006)**	0.322 (0.102)
% of impro.	94.7%	94.6%	94.5%	94.5%	
Bayes Rule	0.51	0.51	0.51	0.51	0.322
Gaussian					
Training Method	Test error				
	l_{0-1}	l_Δ	l_{sib}	l_{sub}	F1-score
SVM	0.547 (0.055)	0.528 (0.026)	0.521 (0.017)	0.521 (0.017)	0.326 (0.012)
ψ-learning	**0.510(0.000)**	**0.510(0.000)**	**0.510(0.000)**	**0.510(0.000)**	0.322 (0.000)
% of impro.	100%	100%	100%	100%	
HSVM$_c$	0.547 (0.054)	0.527 (0.022)	0.521 (0.015)	0.521 (0.015)	0.325 (0.011)
% of impro.	1.0%	1.0%	1.1%	1.1%	
SHSVM	0.626 (0.115)	0.565 (0.054)	0.544 (0.034)	0.544 (0.034)	0.280 (0.078)
% of impro.	-214.6%	-212.3%	-209.8%	-209.8%	
HSVM	0.546 (0.050)	0.527 (0.024)	0.521 (0.015)	0.521 (0.015)	0.324 (0.010)
% of impro.	1.6%	1.6%	1.9%	1.9%	
HPSI	**0.510(0.000)**	**0.510(0.000)**	**0.510(0.000)**	**0.510(0.000)**	0.322 (0.000)
% of impro.	100%	100%	100%	100%	
Bayes Rule	0.51	0.51	0.51	0.51	0.322

Table 1: Averaged test errors as well as estimated standard deviations (in parenthesis) of SVM, ψ-learning, SHSVM, HSVM, HPSI and HSVM$_c$ over 100 simulation replications in Example 1 of Section 4.1. The testing errors are computed under the l_{0-1}, l_Δ, l_{sib} and l_{sub}. The bold face represents the best performance among four competitors for any given loss. For reference, F1-scores, as defined in Section 4.1, for these classifiers are given as well.

the linear case and from 1.0% to 1.1% in the Gaussian kernel case, and that of SHSVM is from -321.8% to -311.9% and -214.6% to -209.8%, which means it is actually much worse than SVM. From hypothesis testing view, the differences of the means for HPSI and SVM are more than three times of the standard error of the differenced means, indicating that these means are statistically different at level of $\alpha = 5\%$. Moreover, HPSI get F1-scores very close to that of the Bayes rule.

In summary, HSVM, especially HPSI indeed yield significant improvements over its competitors.

Example 2. A complete binary tree of depth 3 is considered, which is displayed in Figure 2 (b). There are eight leaf and six non-leaf nodes, coded as $\{1,\cdots,8\}$ and $\{9,\cdots,14\}$, respectively. A random sample of 100 instances $(Y_i, \boldsymbol{X}_i = (X_{i1}, X_{i2}))_{i=1}^{100}$ is generated as follows: $\boldsymbol{X}_i \sim U^2(-1,1)$, where $U^2(-1,1)$ is the uniform distribution on unit square, $Y_i|\boldsymbol{X}_i = \lceil 8 \times X_{i1} \rceil \in \{1,\cdots,8\}$. Then 5% of the samples are randomly chosen with the label values redefined as $Y_i = Y_i + 1$ if $Y_i \neq 8$ and $Y_i = Y_i$ if $Y_i = 8$. Another 5% of the samples are randomly chosen with the label values redefined as $Y_i = Y_i - 1$ if $Y_i \neq 1$ and $Y_i = Y_i$ if $Y_i = 1$. For non-leaf node j, $P(Y_i = j|\boldsymbol{X}_i) = \sum_{\{t \in sub(j) \cap \mathcal{L}\}} P(Y_i = t|\boldsymbol{X}_i)$. This generates a non-separable case.

Linear					
Training Method	Test error				
	l_{0-1}	l_Δ	l_{sib}	l_{sub}	F1-score
SVM	0.326(0.004)	0.179(0.003)	0.148(0.002)	0.122(0.002)	0.671(0.004)
ψ-learning	0.21(0.004)	0.107(0.003)	0.091(0.002)	0.072(0.002)	0.787(0.004)
% of impro.	47.7%	55.4%	50.9%	53.8%	
$HSVM_c$	0.323(0.006)	0.169(0.002)	0.148(0.003)	0.120(0.002)	0.677(0.006)
% of impro.	1.2%	7.7%	0%	2.2%	
SHSVM	0.201(0.003)	0.106(0.002)	0.086(0.001)	0.070(0.001)	0.798(0.003)
% of impro.	51.4%	56.1%	55.4%	55.9%	
HSVM	0.199(0.003)	0.105(0.002)	0.086(0.001)	0.070(0.001)	0.800(0.003)
% of impro.	52.3%	56.9%	55.4%	55.9%	
HPSI	**0.195(0.003)**	**0.102(0.001)**	**0.086(0.002)**	**0.068(0.002)**	0.804(0.003)
% of impro.	53.9%	59.2%	55.4%	58.1%	
Bayes Rule	0.083	0.049	0.036	0.029	0.916
Gaussian					
Training Methods	Test error				
	l_{0-1}	l_Δ	l_{sib}	l_{sub}	F1-score
SVM	0.305(0.015)	0.209(0.001)	0.135(0.008)	0.110(0.007)	0.696(0.015)
ψ-learning	0.206(0.005)	0.113(0.003)	0.087(0.004)	0.069(0.003)	0.798(0.005)
% of impro.	44.6%	60.0%	48.5%	50.6%	
$HSVM_c$	0.313(0.005)	0.166(0.003)	0.128(0.006)	0.109(0.005)	0.685(0.005)
% of impro.	−3.6%	26.9%	7.1%	1.2%	
SHSVM	0.202(0.003)	0.110(0.002)	0.086(0.001)	0.068(0.001)	0.792(0.003)
% of impro.	46.4%	61.9%	49.5%	51.9%	
HSVM	0.205(0.003)	0.112(0.002)	0.087(0.001)	0.069(0.001)	0.795(0.003)
% of impro.	45.0%	60.6%	48.5%	50.6%	
HPSI	**0.190(0.002)**	**0.102(0.002)**	**0.085(0.002)**	**0.063(0.002)**	0.815(0.002)
% of impro.	51.8%	66.9%	50.5%	58.0%	
Bayes Rule	0.083	0.049	0.036	0.029	0.916

Table 2: Averaged test errors as well as estimated standard deviations (in parenthesis) of SVM, ψ-learning, SHSVM, HSVM, HPSI and $HSVM_c$ over 100 simulation replications in Example 2 of Section 4.1. The testing errors are computed under the l_{0-1}, l_Δ, l_{sib} and l_{sub}. The bold face represents the best performance among four competitors for any given loss. For reference, F1-scores, as defined in Section 4.1, for these classifiers are given as well.

As suggested in Table 2, HSVM and HPSI outperform the three competitors under l_{0-1}, l_Δ, l_{sib} and l_{sub} in the linear case, whereas HSVM performs slightly worse than SHSVM in the Gaussian

case. In both cases, the amount of improvement of HSVM and HPSI over their flat counterpart varies. Clearly, HPSI is the winner and outperforms its competitors in all the situations.

With regard to the test errors in Table 2, we also observe the following aspects. First, the five classifiers perform similarly under l_Δ, l_{sib} and l_{sub}. This is because all the eight leaf node classes are at level 3 of the hierarchy, resulting a similar structure under these evaluation losses. Second, the classifiers perform similarly for linear learning and Gaussian kernel learning. This is mainly due to the fact that the ideal optimal decision rule—Bayes rule is linear in this case. Moreover, HPSI and HSVM always have better F1-scores, which are the two most close to that of the Bayes rule.

In summary, HSVM and HPSI indeed yield improvements over their flat counterparts because of the built-in hierarchical structure, and HPSI outperforms its competitors. Here, the hierarchy—a tree of depth 3 is useful in reducing a classification problem's complexity which can be explained by the concept of the margins with respect to hierarchy, as discussed in Section 3.1.

Example 3. A random sample $(Y_i, \boldsymbol{X}_i = \{X_{i1}, X_{i2}\})_{i=1}^n$ is generated as follows. First, $\boldsymbol{X}_i \sim U^2(-1,1)$ is sampled. Second, $Y_i = 1$ if $X_{i1} < 0$ and $X_{i2} < 0$; $Y_i = 2$ if $X_{i1} < 0$ and $X_{i2} \geq 0$; $Y_i = 3$ if $X_{i1} \geq 0$ and $X_{i2} < 0$; $Y_i = 4$ if $X_{i1} \geq 0$ and $X_{i2} \geq 0$. Third, 20% of the sample are chosen at random and their labels are randomly assigned to the other three classes. For non-leaf nodes 5 and 6, $P(Y_i = 5|\boldsymbol{X}_i) = P(Y_i = 1|\boldsymbol{X}_i) + P(Y_i = 2|\boldsymbol{X}_i)$, and $P(Y_i = 6|\boldsymbol{X}_i) = P(Y_i = 3|\boldsymbol{X}_i) + P(Y_i = 4|\boldsymbol{X}_i)$. This generates a complete binary tree of depth 2, where nodes 1 and 2 are siblings of node 5, and nodes 3 and 4 are siblings of node 6, see Figure 2 (c). Experiments are performed with different training sample sizes of 50, 150, 500 and 1500.

Again, HSVM and HPSI outperform their competitors- HSVM$_c$, SHSVM and SVM under the four evaluation losses in all the situations. The amount improvement of HSVM over SVM varies from 22.4% to 52.6% in the linear case and 8.9% to 42.5% in the Gaussian kernel case, whereas that of HPSI ranges from 39.5% to 89.5% and 20.6% to 80.6%, respectively. By comparison, the amount of improvement of HSVM$_c$ is from 6.4% to 23.8% in the linear case and from 2.4% to 18.8% in the Gaussian kernel case, and that of SHSVM is from 21.1% to 47.4% and 9.5% to 45.2%. With regard to F1-scores, HPSI and HSVM remain to be the best, and are much more close to that of the Bayes rule.

In summary, the improvement of HPSI over HSVM becomes more significant when the training size increases. As expected, HPSI is the winner and nearly achieves the optimal performance of the Bayes rule when the sample size gets large.

4.2 Classification of Gene Functions

Biological functions of many known genes remain largely unknown. For yeast *S. cerevisiae*, only 68.5% of the genes were annotated in MIPS, as of May, 2005, for which many of them have only general functions annotated in some top-level categories. Discovery of biological functions therefore becomes very important in biomedical research. As effective means, gene function prediction is performed through known gene functions and gene expression profiles of both annotated and unannotated genes. Biologically, it is generally believed that genes having the same or similar functions tend to be coexpressed (Hughes et al., 2000). By learning the patterns of expression profiles, a gene with unknown functions can be classified into existing functional categories, as well as newly created functional categories. In the process of prediction, classification is essential, as to be discussed next.

Linear							
l &	Sample	TE and % of impro.					
Bayes Rule	Size	SVM	ψ-learning	HSVM$_c$	SHSVM	HSVM	HPSI
l_{0-1}	n=50	0.347(0.070)	0.315(0.058)	0.337(0.047)	0.316(0.047)	0.314(0.058)	**0.289(0.045)**
			21.8%	6.8%	21.1%	22.4%	39.5%
	n=150	0.284(0.043)	0.261(0.030)	0.275(0.023)	0.263(0.023)	0.260(0.030)	**0.237(0.016)**
0.200			27.4%	10.7%	25.0%	28.6%	56.0%
	n=500	0.247(0.014)	0.234(0.013)	0.241(0.014)	0.235(0.014)	0.233(0.013)	**0.213(0.007)**
			27.7%	12.8%	25.5%	29.8%	72.3%
	n=1500	0.230(0.010)	0.217(0.005)	0.223(0.009)	0.218(0.009)	0.217(0.005)	**0.205(0.003)**
			43.3%	23.3%	40.0%	43.3%	83.3%
l_{sib}	n=50	0.276(0.056)	0.249(0.046)	0.269(0.037)	0.250(0.037)	0.248(0.046)	**0.229(0.035)**
			24.8%	6.4%	23.9%	25.7%	43.1%
	n=150	0.230(0.032)	0.211(0.022)	0.222(0.018)	0.213(0.018)	0.210(0.022)	**0.191(0.012)**
0.167			30.2%	12.7%	27.0%	31.7%	61.9%
	n=500	0.203(0.012)	0.193(0.011)	0.198(0.012)	0.194(0.012)	0.192(0.011)	**0.175(0.005)**
			27.8%	13.9%	25.0%	30.6%	77.8%
	n=1500	0.188(0.007)	0.178(0.004)	0.183(0.007)	0.179(0.007)	0.178(0.004)	**0.170(0.002)**
			47.6%	23.8%	42.9%	47.6%	85.7%
l_{sub}	n=50	0.252(0.051)	0.227(0.042)	0.244(0.041)	0.228(0.041)	0.226(0.042)	**0.210(0.033)**
			26.0%	8.3%	25.0%	27.1%	43.8%
	n=150	0.212(0.029)	0.194(0.020)	0.203(0.020)	0.196(0.020)	0.193(0.020)	**0.176(0.010)**
0.156			32.1%	16.1%	28.6%	33.9%	64.3%
	n=500	0.188(0.011)	0.179(0.010)	0.184(0.011)	0.180(0.011)	0.178(0.010)	**0.162(0.005)**
			28.1%	12.5%	25.0%	31.3%	81.3%
	n=1500	0.175(0.007)	0.165(0.004)	0.172(0.007)	0.166(0.007)	0.165(0.004)	**0.158(0.002)**
			52.6%	15.8%	47.4%	52.6%	89.5%
l_{Δ}	n=50	0.184(0.037)	0.166(0.031)	0.179(0.025)	0.167(0.025)	0.165(0.031)	**0.153(0.023)**
			24.7%	6.4%	23.7%	25.6%	42.9%
	n=150	0.153(0.021)	0.141(0.015)	0.148(0.012)	0.142(0.012)	0.140(0.015)	**0.127(0.008)**
0.111			28.6%	12.6%	26.8%	31.5%	61.4%
	n=500	0.135(0.008)	0.128(0.007)	0.132(0.008)	0.129(0.008)	0.128(0.007)	**0.117(0.003)**
			29.2%	13.7%	24.7%	30.1%	76.7%
	n=1500	0.125(0.005)	0.119(0.003)	0.122(0.005)	0.119(0.005)	0.119(0.003)	**0.113(0.002)**
			42.9%	23.3%	41.9%	46.5%	83.7%
F1-score	n=50	0.557(0.106)	0.588(0.107)	0.571(0.075)	0.588(0.076)	0.589(0.106)	**0.597(0.110)**
			12.8%	5.8%	12.8%	13.2%	16.5%
	n=150	0.672(0.063)	0.701(0.055)	0.683(0.026)	0.710(0.026)	0.691(0.054)	**0.719(0.034)**
0.800			22.7%	8.6%	29.7%	14.8%	36.7%
	n=500	0.721(0.016)	0.741(0.017)	0.729(0.015)	0.749(0.014)	0.737(0.017)	**0.754(0.012)**
			25.3%	10.1%	35.4%	20.3%	41.8%
	n=1500	0.746(0.017)	0.763(0.015)	0.755(0.010)	0.763(0.010)	0.764(0.015)	**0.779(0.004)**
			31.5%	16.7%	31.5%	33.3%	61.1%

Table 3: Averaged test errors as well as estimated standard deviations (in parenthesis) of SVM, ψ-learning, HSVM$_c$, SHSVM, HSVM and HPSI over 100 simulation replications of linear learning in Example 3 of Section 4.1, with $n = 50, 150, 500, 1500$. The test errors are computed under the l_{0-1}, l_{Δ}, l_{sib} and l_{sub}. For reference, F1-scores, as defined in Section 4.1, for these classifiers are given as well.

Hughes et al. (2000) demonstrated the effectiveness of gene function prediction through genome-wide expression profiles, and identified and experimentally confirmed eight uncharacterized open reading frames as protein-coding genes. Specifically, three hundred expressions were profiled for the genome of yeast *S. cerevisiae*, in which transcript levels of a mutant or a compound-treated culture were compared against that of a wild-type or a mock-treated culture. Three hundred experiments, consisting of 276 deletion mutants, 11 tetracycline-regulatable alleles of essential genes, and 13 well-characterized compounds. Deletion mutants were selected such that a variety of functional

		Gaussian					
l &	Sample	TE and % of impro.					
Bayes Rule	Size	SVM	ψ-learning	HSVM$_c$	SHSVM	HSVM	HPSI
l_{0-1}	n=50	0.326(0.060)	0.313(0.047)	0.323(0.047)	0.314(0.047)	0.313(0.047)	**0.300(0.045)**
			10.3%	2.4%	9.5%	8.9%	20.6%
	n=150	0.280(0.036)	0.27 (0.027)	0.276(0.030)	0.270(0.030)	0.270(0.027)	**0.261(0.016)**
0.200			12.5%	5.0%	12.5%	12.5%	23.8%
	n=500	0.257(0.022)	0.24 (0.014)	0.252(0.013)	0.239(0.013)	0.240(0.014)	**0.224(0.007)**
			29.8%	8.8%	31.6%	29.8%	57.9%
	n=1500	0.247(0.013)	0.227(0.010)	0.240(0.011)	0.226(0.011)	0.227(0.010)	**0.215(0.003)**
			42.6%	14.9%	44.7%	42.5%	68.1%
l_{sib}	n=50	0.263(0.048)	0.250(0.037)	0.257(0.037)	0.251(0.037)	0.250(0.037)	**0.243(0.035)**
			13.5%	6.3%	12.5%	9.8%	20.8%
	n=150	0.229(0.029)	0.218(0.022)	0.225(0.022)	0.219(0.022)	0.218(0.022)	**0.211(0.012)**
0.167			17.7%	6.5%	16.1%	13.1%	29.0%
	n=500	0.208(0.016)	0.195(0.010)	0.202(0.011)	0.194(0.011)	0.195(0.010)	**0.181(0.005)**
			31.7%	14.6%	34.1%	31.7%	65.9%
	n=1500	0.198(0.008)	0.185(0.006)	0.193(0.005)	0.184(0.005)	0.185(0.006)	**0.173(0.003)**
			41.9%	16.1%	45.2%	40.9%	80.6%
l_{sub}	n=50	0.241(0.044)	0.227(0.034)	0.237(0.041)	0.230(0.041)	0.228(0.034)	**0.222(0.033)**
			16.5%	4.7%	12.9%	11.1%	22.4%
	n=150	0.211(0.027)	0.2 (0.020)	0.207(0.020)	0.202(0.020)	0.201(0.020)	**0.192(0.010)**
0.156			20.0%	7.3%	16.4%	13.5%	34.5%
	n=500	0.192(0.015)	0.179(0.009)	0.187(0.010)	0.179(0.010)	0.180(0.009)	**0.169(0.005)**
			36.1%	13.9%	36.1%	33.3%	63.9%
	n=1500	0.188(0.009)	0.175(0.006)	0.182(0.005)	0.175(0.005)	0.176(0.006)	**0.163(0.003)**
			40.6%	18.8%	40.6%	35.4%	78.1%
l_Δ	n=50	0.175(0.032)	0.167(0.025)	0.171(0.025)	0.167(0.025)	0.166(0.025)	**0.162(0.023)**
			12.5%	6.2%	12.4%	9.8%	20.7%
	n=150	0.153(0.019)	0.145(0.015)	0.150(0.014)	0.146(0.014)	0.145(0.015)	**0.141(0.008)**
0.111			19.0%	6.4%	16.0%	13.1%	28.8%
	n=500	0.139(0.011)	0.13 (0.007)	0.135(0.007)	0.129(0.007)	0.130(0.007)	**0.121(0.003)**
			32.1%	14.5%	33.7%	31.7%	65.1%
	n=1500	0.132(0.005)	0.123(0.004)	0.129(0.004)	0.123(0.004)	0.123(0.004)	**0.115(0.002)**
			42.9%	15.9%	44.4%	41.3%	79.4%
F1-score	n=50	0.559(0.105)	0.589(0.107)	0.573(0.076)	0.590(0.076)	0.591(0.105)	**0.595(0.109)**
			12.4%	5.8%	12.9%	13.3%	14.9%
	n=150	0.674(0.062)	0.703(0.053)	0.686(0.024)	0.713(0.025)	0.695(0.051)	**0.717(0.033)**
0.800			23.0%	9.5%	31.0%	16.7%	34.1%
	n=500	0.723(0.016)	0.744(0.017)	0.732(0.014)	0.752(0.014)	0.740(0.016)	**0.753(0.012)**
			27.3%	11.7%	37.7%	22.1%	39.0%
	n=1500	0.747(0.017)	0.765(0.015)	0.757(0.011)	0.766(0.010)	0.767(0.015)	**0.776(0.004)**
			34.0%	18.9%	35.8%	37.7%	54.7%

Table 4: Averaged test errors as well as estimated standard deviations (in parenthesis) of SVM, ψ-learning, HSVM$_c$, SHSVM, HSVM and HPSI over 100 simulation replications of kernel learning in Example 3 of Section 4.1, with $n = 50, 150, 500, 1500$. The test errors are computed under the l_{0-1}, l_Δ, l_{sib} and l_{sub}. For reference, F1-scores, as defined in Section 4.1, for these classifiers are given as well.

classifications were represented. Experiments were performed under a common condition to allow direct comparison of the behavior of all genes in response to all mutations and treatments. Expressions of the three hundred experiments were profiled through a two-channel cDNA chip technology (or hybridization assay). As suggested in Hughes et al. (2000), the expression profiles were indeed informative to gene function prediction.

In gene function prediction, one major difficulty is the presence of a large number of function categories with relatively small-sample size, which is known as the situation of large number of cate-

gories in classification. To battle the curse of dimensionality, a structured approach needs to be used with built-in biological knowledge presented in a form of annotation system such as MIPS, where a flat approach does not perform better than a classifier that uses a correct hierarchical structure. Comparisons can be found in Shahbaba and Neal (2007), and Cesa-Bianchi and Valentini (2009). The problem of gene function prediction is an ideal test case for hierarchical classification, where accuracy of prediction is key. In the literature, recalibration and combination of different large margin methods, including sequential HSVM and loss scaled SVM, were used in gene function prediction, see, for example, Obozinski et al. (2008), Guan et al. (2008), and Valentini and Re (2009). Astikainen et al. (2008) used a different representation with a loss-scaled SVM. Cesa-Bianchi et al. (2006), and Cesa-Bianchi and Valentini (2009) employed a Bayesian ensemble method.

Through gene expression data in Hughes et al. (2000), we apply HSVM and HPSI to predict gene functions. Of particular consideration is prediction of functional categories of unknown genes within two major branches of MIPS, composed of two functional categories at the highest level: "cell cycle and DNA processing" and "transcription" and their corresponding offsprings. Within these two major branches, we have $n = 1103$ annotated genes together with $p = 300$ expressions for each gene and a tree hierarchy of $K = 23$ functional categories, see Figure 3 for a display of the tree hierarchy. In this case, the predictor $\boldsymbol{x}$ represents the expression levels of a gene, consisting of the log-ratios (base 10) of the mRNA abundance in the test samples relative to the reference samples, and label Y indicates the location within the MIPS hierarchy. For this MIPS data, some genes belong to two or more functional categories. To place the problem of gene function prediction into our framework of hierarchical classification, we may create a new separate category for all genes that are annotated with a common set of categories. For an example, in Figure 2 (b), if we observed cases of common members for Categories 2 and 3, we will create a category, say Category 15, which is the sibling of Categories 9 and 10. Those common members will be assigned to Category 15.

Notice that, although we have nearly balanced situation in this example, in general we may see unbalanced situations.

We now perform some simulations to gain insight into HSVM and HPSI for gene function prediction before applying to predict new gene functions with respect to MIPS. For this purpose, we use the 2005 version of MIPS and proceed 100 randomly partition the entire set of data of 1103 genes into training, tuning and testing sets, with 303, 300, and 500 genes, respectively. Then HSVM and HPSI are trained with training samples, tuned and tested as in Section 4.1, over 100 different random partitions to avoid homologous gene clusters. Their performance is measured by the test errors is averaged over 100 random partitions.

As suggested by Table 5, besides similar results in F1-score, HSVM and HPSI outperform SVM and HSVM$_c$ under l_{0-1}, l_Δ, l_{sib} and l_Δ, in both the linear and Gaussian kernel cases. With respect to these four losses, the amount of improvement of HSVM over SVM ranges from 0.1% to 31.8%, whereas that of HPSI over SVM is from 0.1% to 32.3%. Among these four losses, l_Δ and l_{sub} yield the largest amount of improvement. This suggests that HPSI and HSVM classify more precisely at the top levels than at the bottom levels of the hierarchy, where the inter-class dependencies are weak. Note that l_Δ and l_{sub} penalize misclassification more at relevant nodes at lower levels in deep branches, whereas l_{sib} only does so at upper levels. Interestingly, small and large branches have the same parents, leading to large differences in penalties under different losses. It is also noted that the F1-scores are not significantly above 0 for all the large margin methods we are comparing here.

We are now ready to apply our method to real challenge of predicting unknown gene functional categories that had not been annotated in the 2005 version of MIPS. The predicted gene functions

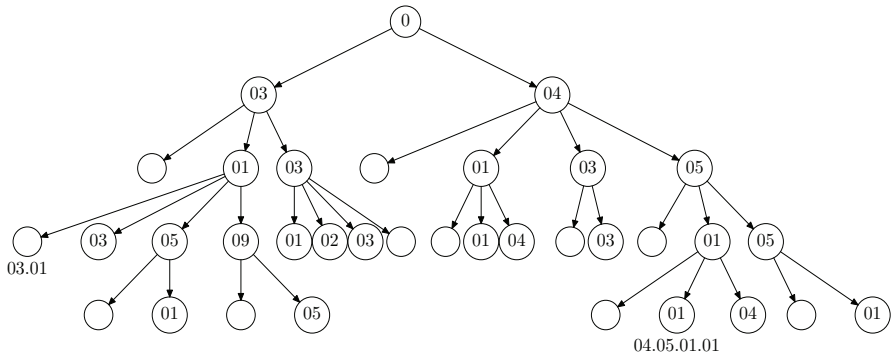


Figure 3: Two major branches of MIPS, with two functional categories at the highest level: "Cell cycle and DNA processing" and "Transcription". For each node, the corresponding functional category is defined by a combination of numbers of itself and all of its ancestors. For instance, the middle node 01 at level 4 indicates functional category 04.05.01.01 in MIPS, which corresponds to "General transcription activities". Notes that a blank node represents its parent itself, for instance, the left blank node at level 3 indicates functional category 03.01.

will be cross-validated by a newer version of MIPS, dated in March 2008, where about 600 additional genes have been added into functional categories, representing the latest biological advance. We proceed in three steps. First, we use the tuned HSVM and HPSI trained through the training samples in the 2005 version of MIPS, which are the best performer over the 100 random partitions. Second, the trained HPSI and HSVM are applied to ten most confident genes for prediction, which are chosen among unannotated genes in the 2005 version but annotated in the 2008 version. Here the confidence is measured by the value of the functional margin. Third, ten predictions from HSVM and HPSI are cross-validated by the 2008 version of MIPS.

As indicated in Table 6, seven out of the ten genes are predicted correctly for both HSVM and HPSI. For an example, gene "YGR054w" is not annotated in the 2005 version of MIPS, and is predicted to belong to functional categories along a path "Protein synthesis" $\rightarrow$ "Ribosome biogenesis" $\rightarrow$ "Ribosomal proteins" by HPSI. This prediction is confirmed to be exactly correct by the newer version of MIPS.

5. Statistical Learning Theory

In the literature, the generalization accuracy for hierarchical classification and the role of $\mathcal{H}$ have not been widely studied. This section develops an asymptotic theory to quantify the generalization accuracy of the proposed hierarchical large margin classifier $d^H(\hat{\boldsymbol{f}})$ defined by (2) for a general loss v. In particular, the rate of convergence of $d^H(\hat{\boldsymbol{f}})$ is derived. Moreover, we apply the theory to one illustrative example to study when and how $\mathcal{H}$ improves the performance over flat classification.

Linear					
Training Methods	Test error				
	l_{0-1}	l_Δ	l_{sib}	l_{sub}	F1-score
SVM	0.972(0.006)	0.651(0.024)	0.926(0.008)	0.593(0.029)	0.007(0.002)
ψ-learning	0.961(0.009)	0.633(0.023)	0.92(0.022)	0.581(0.041)	0.007(0.002)
% of impro.	1.13%	2.8%	0.7%	2.0%	
SHSVM	0.962(0.007)	0.552(0.031)	0.927(0.008)	0.442(0.036)	0.015(0.002)
% of impro.	1.0%	15.2%	0.1%	25.5%	
HSVM	0.960(0.009)	0.520(0.023)	0.918(0.022)	0.433(0.041)	0.008(0.002)
% of impro.	1.2%	20.0%	0.8%	27.0%	
HPSI	**0.958(0.008)**	**0.517(0.020)**	**0.917(0.020)**	**0.430(0.038)**	0.009(0.002)
% of impro.	1.4%	20.6%	1.0%	27.5%	
Gaussian					
Training Methods	Test error				
	l_{0-1}	l_Δ	l_{sib}	l_{sub}	F1-score
SVM	0.976(0.002)	0.669(0.005)	0.921(0.003)	0.617(0.007)	0.007(0.002)
ψ-learning	0.961(0.008)	0.660(0.020)	0.92(0.019)	0.601(0.030)	0.007(0.002)
% of impro.	1.5%	1.3%	0.1%	2.6%	
SHSVM	0.963(0.006)	0.558(0.033)	0.920(0.009)	0.430(0.029)	0.016(0.002)
% of impro.	1.3%	16.6%	0.1%	30.3%	
HSVM	0.961(0.008)	0.515(0.020)	0.920(0.019)	0.421(0.030)	0.008(0.002)
% of impro.	1.5%	23.0%	0.1%	31.8%	
HPSI	**0.960(0.008)**	**0.512(0.021)**	**0.920(0.020)**	**0.418(0.030)**	0.009(0.002)
% of impro.	1.6%	23.5%	0.1%	32.3%	

Table 5: Averaged test errors as well as estimated standard deviations (in parenthesis) of SVM, ψ-learning, SHSVM, HSVM and HPSI, in the gene function example in Section 4.2, over 100 simulation replications. The testing errors are computed under l_{0-1}, l_Δ, l_{H-sib} and l_{H-sub}. The bold face represents the best performance among four competitors for any given loss. For reference, F1-scores, as defined in Section 4.1, for these classifiers are given as well.

5.1 Theory

In classification, the performance of our classifier $d^H(\hat{\boldsymbol{f}})$ is measured by the difference between the actual performance of $\hat{\boldsymbol{f}}$ and the ideal optimal performance of $\bar{\boldsymbol{f}}$, defined as $e(\hat{\boldsymbol{f}},\bar{\boldsymbol{f}}) = GE(d^H(\hat{\boldsymbol{f}})) - GE(d^H(\boldsymbol{f})) = E(l_{0-1}(Y,d^H(\hat{\boldsymbol{f}}(\boldsymbol{X}))) - l_{0-1}(Y,d^H(\boldsymbol{f}(\boldsymbol{X})))) \geq 0$. Here $GE(d^H(\boldsymbol{f}))$ is the optimal performance for any classifier provided that the unknown true distribution $P(\boldsymbol{x},y)$ would have been available. In hierarchical classification with k leaf and $(K-k)$ non-leaf node classes, the Bayes decision function vector $\boldsymbol{f}$ is a decision function vector yielding the Bayes classifier under d^H, that is, $d^H(\boldsymbol{f}(\boldsymbol{x})) = d(\boldsymbol{x})$. In our context, we define $\boldsymbol{f}$ as follows: for each j, $f_j(\boldsymbol{x}) = \max_{t:t\in sub(j)\cap\mathcal{L}} P(Y = t|\boldsymbol{X}=\boldsymbol{x})$ if $j \notin \mathcal{L}$ and $f_j(\boldsymbol{x}) = P(Y=j|\boldsymbol{X}=\boldsymbol{x})$ if $j\in\mathcal{L}$.

Let $e_V(\boldsymbol{f},\bar{\boldsymbol{f}}) = E(V(\boldsymbol{f},\boldsymbol{Z}) - V(\bar{\boldsymbol{f}},\boldsymbol{Z})) \geq 0$ and $\lambda = (nC)^{-1}$, where $V(\boldsymbol{f},\boldsymbol{Z})$ is defined as $v(u_{min}(\boldsymbol{f}(\boldsymbol{X}),Y))$, $\boldsymbol{Z} = (\boldsymbol{X},Y)$, and $v(\cdot)$ is any large margin surrogate loss used in (2).

The following theorem quantifies Bayesian regret $e(\hat{\boldsymbol{f}},\bar{\boldsymbol{f}})$ in terms of the tuning parameter C through $\lambda = \frac{1}{nC}$, the sample size n, the smoothness parameters (α,β) of a surrogate loss V-based classification model, and the complexity of the class of candidate function vectors $\mathcal{F}$. Note that the assumptions below are parallel to those of Theorem 3 in Liu and Shen (2006) for a statistical

Gene	Function category	Prediction verified			
		HSVM	HPSI	HSVM$_c$	SHSVM
YGR054w	translation initiation	Yes	Yes	Yes	Yes
YCR072c	ribosome biogenesis	Yes	Yes	Yes	Yes
YFL044c	transcriptional control	Yes	Yes	Yes	Yes
YNL156c	binding / dissociation	No	No	No	No
YPL201c	C-compound and carbohydrate utilization	Yes	Yes	Yes	Yes
YML069W	mRNA synthesis	Yes	Yes	Yes	Yes
YOR039W	mitotic cell cycle and cell cycle control	Yes	Yes	No	Yes
YNL023C	mRNA synthesis	No	Yes	No	No
YPL007C	mRNA synthesis	No	No	No	No
YDR279W	DNA synthesis and replication	Yes	No	No	No

Table 6: Verification of 10 gene predictions using an updated MIPS system and their functional categories.

learning theory for multiclass SVM and ψ-learning. In particular, Assumptions A-C described in Appendix are used to quantity the error rate of the classifier, in addition to a complexity measure the metric entropy with bracketing H_B for function space $\mathcal{F}$ defined before Assumption C.

Theorem 2 *Under Assumptions A-C in Appendix A, for any large margin hierarchical classifier $\boldsymbol{d}^H(\hat{\boldsymbol{f}})$ defined by (1), there exists a constant $c_6 > 0$ such that for any $x \geq 1$,*

$$P\left(e(\hat{\boldsymbol{f}}, \bar{\boldsymbol{f}}) \geq c_1 x \delta_n^{2\alpha}\right) \leq 3.5 \exp(-c_6 x^{2-\min(\beta,1)} n(\lambda J_0)^{2-\min(\beta,1)}),$$

provided that $\lambda^{-1} \geq 2\delta_n^{-2} J_0$, where $\delta_n^2 = \min(\varepsilon_n^2 + 2e_V(\boldsymbol{f}^*, \bar{\boldsymbol{f}}), 1)$, $\boldsymbol{f}^* \in \mathcal{F}$ is an approximation in $\mathcal{F}$ to f, $J_0 = \max(J(\boldsymbol{f}^*), 1)$ with $J(\boldsymbol{f}) = \frac{1}{2}\sum_{j=1}^K \|f_j\|_{\mathcal{K}}^2$, and α, β, ε_n are defined in Assumptions A-C in Appendix A.

Corollary 1 *Under the assumptions in Theorem 2, $|e(\hat{\boldsymbol{f}}, \boldsymbol{f})| = O_p\left(\delta_n^{2\alpha}\right)$ and $E|e(\hat{\boldsymbol{f}}, \boldsymbol{f})| = O\left(\delta_n^{2\alpha}\right)$, provided that $n(\lambda J_0)^{2-\min(\beta,1)}$ is bounded away from 0 $n \to \infty$.*

The convergence rate for $e(\hat{\boldsymbol{f}}, \bar{\boldsymbol{f}})$ is determined by δ_n^2, $\alpha > 0$ and $\beta > 0$, where δ_n captures the trade-off between the approximation error $e_V(\boldsymbol{f}^*, \bar{\boldsymbol{f}})$ due to use the surrogate loss V and estimation error ε_n^2, where ε_n is defined by the bracketing L_2 entropy of candidate function space $\mathcal{F}^V(t) = \{V^T(\boldsymbol{f}, \boldsymbol{z}) - V(\bar{\boldsymbol{f}}, \boldsymbol{z}) : \boldsymbol{f} \in \mathcal{F}, J(\boldsymbol{f}) \leq J_0 t\}$, and the last two quantify the first and second moments of $EV(\boldsymbol{f}, Z)$, where $\boldsymbol{z} = (\boldsymbol{x}, y)$ and $\boldsymbol{Z} = (\boldsymbol{x}, Y)$.

By comparison, with V induced by a margin loss v, $\mathcal{F}^V(t)$ in multiclass classification is usually larger than its counterpart in hierarchical classification. This is because V is structured in that functional margin $u_{min}(\boldsymbol{f}(\boldsymbol{X}), Y)$ involves a smaller number of pairwise comparisons in hierarchical classification. In fact, only siblings for Y or one of Y's ancestors are compared. In contrast, any two classes need to be compared in multiclass classification without such a hierarchy (Liu and Shen, 2006). A theoretical description regarding the reduced size of the effective parameter space $\mathcal{F}^V(t)$ is given in the following lemma.

With regard to tightness of the bounds derived in Theorem 1, note that it reduces to multiclass margin classification, where the linear example in Shen and Wang (2007) indicates that the n^{-1} rate obtained from the upper bound theory agrees with the optimal rate of convergence.

Lemma 2 *Let $\mathcal{H}$ be a tree hierarchy with K non-root nodes including k leaf nodes. If $\mathcal{F}_1 = \cdots = \mathcal{F}_K$, then $H_B(\varepsilon, \mathcal{F}^V(t)) \leq 2c(\mathcal{H}) H_B(\varepsilon/(2c(\mathcal{H})), \mathcal{F}_1(t))$ with v being the hinge and ψ losses, where $c(\mathcal{H}) = \sum_{j=0}^{K} \frac{|chi(j)|(|chi(j)|-1)}{2} \leq \frac{k(k-1)}{2}$ is the total number of comparisons required for hierarchical classification, and $\mathcal{F}_j(t) = \{f_j : \frac{1}{2}\|f_j\|_{\mathcal{K}} \leq J_0 t\}$; $j = 1, \cdots, K$.*

5.2 Bayes Classifier and Fisher-consistency

To compare different losses for the purpose of hierarchical classification, we introduce a new concept called "Fisher-consistency" with respect to $\mathcal{H}$. Before proceeding, we define the Bayes rule in Lemma 3 for K-class classification with non-exclusive membership, where only $k < K$ classes have mutually exclusive membership, determining the class membership of the other $K-k$ non-exclusive classes.

Lemma 3 *In K-class classification with non-exclusive membership, assume that the k mutually exclusive membership classes uniquely determine the membership of the other $K-k$ non-exclusive classes. That is, for any $t \in E$ and $\tilde{t} \notin E$, either $\{Y = \tilde{t}\} \supseteq \{Y = t\}$, or $\{Y = \tilde{t}\} \subseteq \{Y \neq t\}$, where E is the set of k mutually exclusive membership classes. Then the Bayes classifier $d(\boldsymbol{x}) = \text{argmax}_{j \in E} P(Y = j | \boldsymbol{X} = \boldsymbol{x})$.*

Based on Lemma 3, we define Fisher-consistency with respect to $\mathcal{H}$ in hierarchical classification, which can be regarded as a generalization of Fisher-consistency in multi classification cases.

Definition 1 *In hierarchical classification, denote by $\mathcal{L}$ the set of classes corresponding to the leaf nodes in a tree. With $\mathcal{L}$ being a set of mutually exclusive membership classes, a loss $l(\cdot,\cdot)$ is said to be Fisher-consistent with respect to $\mathcal{H}$ if a global minimizer $El(Y, \boldsymbol{f}(\boldsymbol{X}))$ over all possible $\boldsymbol{f}(\boldsymbol{x})$ is $\boldsymbol{f}$.*

Lemma 4 *Loss l_{0-1} is Fisher-consistent with respect to $\mathcal{H}$; so is l_Δ in the presence of a dominating leaf node class, that is, a class such that for any $\boldsymbol{x} \in S$ there exists a leaf node class j such that $P(Y = j | \boldsymbol{X} = x) > 1/2$.*

As shown in Lemma 4, l_{0-1} and l_Δ are Fisher-consistent with respect to $\mathcal{H}$.

Lemma 5 *Surrogate loss $v(u_{min}(\boldsymbol{f}(\boldsymbol{x}), y))$ is Fisher-consistent with respect to $\mathcal{H}$ when $v(\cdot)$ is either the hinge loss or the ψ loss.*

5.3 Theoretical Examples

Consider hierarchical classification with $\mathcal{H}$ defined by a complete binary tree with depth p. For this tree, there are $k = 2^p$ leaf nodes and $K = 2^{p+1} - 2 = 2k - 2$ non-root nodes, see Figure 2 (b) for an example of $p = 3$. Without loss of generality, denote by $\{j_1, \cdots, j_k\}$ the k leaf nodes. In what follows, we focus on the 0-1 loss with $l = l_{0-1}$.

A random sample is generated: $\boldsymbol{X} = (X_{(1)}, X_{(2)})$ sampled from the uniform distribution over $S = [0,1]^2$. For any leaf node j_i; $i = 1, \cdots, k$, when $X_{(1)} \in [(i-1)/k, i/k)$, $P(Y = j_i | \boldsymbol{X}) = (k-1)/k$,

and $P(Y=j|\boldsymbol{X})=1/[k(k-1)]$ for $j\neq j_i$. For any non-leaf node j_i; $i=k+1,\cdots,K$, $P(Y=j_i|\boldsymbol{X})=\sum_{t\in sub(j_i)\cap\mathcal{L}}P(Y=t|\boldsymbol{X})$. Then the Bayes rule d is defined from the Bayes decision function $\bar{\boldsymbol{f}}=\{f_1,\cdots,f_K\}$ through the top-down rule, where $\bar{\boldsymbol{f}}$ is defined as follows: For leaf nodes, $f_{j_i}(\boldsymbol{x})=\sum_{t=1}^{i}(x_{(1)}-t/k)$; $i=1,\cdots,k$, so that when $x_{(1)}\in[(i_0-1)/k,i_0/k)$, $f_{j_{i_0}}(\boldsymbol{x})=\max_{i=1,\cdots,k}f_{j_i}(\boldsymbol{x})$. For non-leaf nodes, let it be the maximum over the leaf nodes in the subtree, that is, $f_{j_i}(\boldsymbol{x})=\max_{\{t\in sub(j_i)\cap\mathcal{L}\}}f_t$; $i=k+1,\cdots,K$.

Linear learning: Let $\mathcal{F}=\{(f_1,\cdots,f_K):f_j=\boldsymbol{w}_j^T\boldsymbol{x}+b_j\}$ and $J(\boldsymbol{f})=\sum_{j=1}^{K}\|\boldsymbol{w}_j\|^2$, where $\|\cdot\|$ is the EuclideanL_2-norm. We now verify Assumptions A-C for Corollary 1. It follows from Lemma 3 of Shen and Wang (2007) with $\boldsymbol{f}^*=\arg\inf_{\boldsymbol{f}\in\mathcal{F}}El_{0-1}(\boldsymbol{f},\boldsymbol{Z})$ for HSVM and $f_j^*=\sum_{t=1}^{j}n(x_{(1)}-t/k)$ for HPSI; $j=1,\cdots,k$, and $f_j^*=\max_{\{t\in sub(j)\cap\mathcal{L}\}}f_t^*$ otherwise. Assumptions A and B there are met with $\alpha=\frac{1}{2}$ and $\beta=1$ for HSVM, and with $\alpha=\beta=1$ for HPSI. For Assumption C, note that $H_B(\varepsilon,\mathcal{F}_1(t))\leq O(\log(1/\varepsilon))$, by Lemma 2 with $c(\mathcal{H})=\sum_{j=0}^{K}|chi(j)|(|chi(j)|-1)/2=\sum_{j=0}^{K}I\{j\notin\mathcal{L}\}=k-1$, we have, for HSVM and HPSI, $H_B(\varepsilon,\mathcal{F}^V(t))\leq O(k\log(k/\varepsilon))$ (Kolmogorov and Tihomirov, 1959). Consequently, $L\leq O(\varepsilon_n^2)$ in Assumption C, where $\phi(\varepsilon_n,s)=\int_{c_3L}^{c_4^{1/2}L^{\beta/2}}H_B^{1/2}(u,\mathcal{F}^V(s))du/L$ and $\sup_{t\geq 2}\phi(\varepsilon_n,t)\leq O((k\log(k/\varepsilon_n))^{1/2}/\varepsilon_n)$. Solving (7) in Assumption C leads to $\varepsilon_n=(\frac{k\log n}{n})^{1/2}$ for HSVM and HPSI when $C/J_0\sim\delta_n^{-2}/n\sim\frac{1}{n\varepsilon_n^2}$, provided that $\frac{k\log n}{n}\to 0$, with δ_n as defined in Theorem 2. Similarly, for multiclass SVM and ψ-learning, $\varepsilon_n=(\frac{k(k-1)/2\log n}{n})^{1/2}$ (Shen and Wang, 2007).

By Corollary 1, $|e(\hat{\boldsymbol{f}},\bar{\boldsymbol{f}})|=O_p\Big((k\log n/n)^{1/2}\Big)$ and $E\,|e(\hat{\boldsymbol{f}},\bar{\boldsymbol{f}})|=O\Big((k\log n/n)^{1/2}\Big)$ for HSVM, and $|e(\hat{\boldsymbol{f}},\bar{\boldsymbol{f}})|=O_p\Big(k\log n/n\Big)$ and $E\,|e(\hat{\boldsymbol{f}},\bar{\boldsymbol{f}})|=O\Big(k\log n/n\Big)$ for HPSI, when $\frac{k\log n}{n}\to 0$ as $n\to\infty$. By comparison, the rates of convergence for SVM and ψ-learning are $O\Big((\frac{k(k-1)}{2}\log n/n)^{1/2}\Big)$ and $O\Big(\frac{k(k-1)}{2}\log n/n\Big)$. In this case, the hierarchy enables to reduce the order from $\frac{k(k-1)}{2}$ down to k.

Note that $\mathcal{H}$ is a flat tree with only one layer, that is, all the leaf nodes are the direct offsprings of the root node 0, which means that $|chi(0)|=k$. Then $c(\mathcal{H})=\frac{|chi(0)|(|chi(0)|-1)}{2}=\frac{k(k-1)}{2}$. This would lead to the same rates of convergence for HSVM and HPSI as their counterparts.

Gaussian kernel learning: Consider the same setting with candidate function class defined by the Gaussian kernel. By the Aronszajn representation theorem of the reproducing kernel Hilbert spaces (Gu, 2000), it is convenient to embed a finite-dimensional Gaussian kernel representation into an infinite-dimensional space $\mathcal{F}=\{\boldsymbol{x}\in\mathcal{R}^2:\boldsymbol{f}(\boldsymbol{x})=(f_1(\boldsymbol{x}),\ldots,f_K(\boldsymbol{x}))$ with $f_j(\boldsymbol{x})=b_j+\boldsymbol{w}_j^T\phi(\boldsymbol{x})=b_j+\sum_{l=0}^{\infty}w_{j,l}\phi_l(\boldsymbol{x}):\boldsymbol{w}_j\in l_2\}$, and $\langle\phi(\boldsymbol{x}),\phi(\boldsymbol{z})\rangle=\mathcal{K}(\boldsymbol{x},\boldsymbol{z})=\exp(-\frac{\|\boldsymbol{x}-\boldsymbol{z}\|^2}{2\sigma_n^2})$, where σ_n is a scaling tuning parameter for the Gaussian kernel, which may depend on n. In what follows, we verify Assumptions A-C for HSVM and HPSI separately, and calculate δ_n in Corollary 1.

For HSVM, letting $f_j^*=1-(1+\exp(\sum_{t=1}^{j}\tau(x_{(1)}-t/k)))^{-1}$; for $j=1,\cdots,k$, and letting $f_j^*=\max_{\{t\in sub(j)\cap\mathcal{L}\}}f_t^*$ otherwise, $e(\boldsymbol{f}^*,\boldsymbol{f})=O(k/\tau)$ and $J(\boldsymbol{f}^*)=O(ke^{\tau^2\sigma_n^2})$. Assumptions A and B are met with $\alpha=\beta=1$ by Lemmas 6 and 7 of Shen and Wang (2007). For Assumption C, following from Section 5.3 of Shen and Wang (2007), we have $H_B(\varepsilon,\mathcal{F}_1(t))\leq O((\log((J_0t)^{1/2}/\varepsilon))^3)$. By Lemma 2, with $c(\mathcal{H})=k-1$ as calculated in the linear cases, we have that $H_B(\varepsilon,\mathcal{F}^V(t))\leq O(k(\log((J_0t)^{1/2}k/\varepsilon))^3)$, where $J_0=\max(J(\boldsymbol{f}^*),1)$. Note that $L\leq O(\varepsilon_n^2)$. Then $\sup_{t\geq 2}\phi(\varepsilon_n,t)\leq O((k(\log((J_0t)^{1/2}k/\varepsilon_n))^3)^{1/2}/\varepsilon_n)$. Solving (7) in Assumption C leads to $\varepsilon_n^2=kn^{-1}(\log((J_0n)^{1/2}))^3$ when $\lambda J_0\sim\varepsilon_n^2$. By Corollary 1, $e(\hat{\boldsymbol{f}},\boldsymbol{f})=O_p(\delta_n^2)$ and $Ee(\hat{\boldsymbol{f}},\boldsymbol{f})=O(\delta_n^2)$, with $\delta_n^2=\max(kn^{-1}(\tau^2\sigma_n^2$

$+\sigma_n^{-2}+\log n)^3, k/\tau) = O_p(kn^{-1/7})$ with $\tau \sim n^{1/7}$ when σ_n^2 is fixed, and $O_p(kn^{-1/4})$ when $\tau \sim \sigma_n^{-2} \sim n^{1/4}$.

For HPSI, let $f_j^* = \sum_{\tilde{j}=1}^{j} \tau(x_{(1)} - \tilde{j}/k)$; $j = 1, \cdots, k$, and $f_j^* = \max_{\{t \in sub(j) \cap \mathcal{L}\}} f_t^*$ otherwise. Then it can be verified that $e_L(\boldsymbol{f}^*, \boldsymbol{f}) = O(k/\tau)$ and $J(\boldsymbol{f}^*) = O(k\tau^2\sigma_n^2)$. Assumptions A and B are met with $\alpha = \beta = 1$ by Theorem 3.1 of Liu and Shen (2006). Also $H_B(\varepsilon, \mathcal{F}_1(t)) \leq O((\log((J_0 t)^{1/2}/\varepsilon))^3)$, thus $\sup_{t \geq 2} \phi(\varepsilon_n, t) \leq O((k(\log((J_0 t)^{1/2} k/\varepsilon_n))^3)^{1/2}/\varepsilon_n)$. Similarly as in HSVM, solving (7) in Assumption C leads to $\varepsilon_n^2 = kn^{-1}(\log((J_0 n)^{1/2}))^3$ when $\lambda J_0 \sim \varepsilon_n^2$. By Corollary 1, $e(\hat{\boldsymbol{f}}, \boldsymbol{f}) = O_p(\delta_n^2)$ and $Ee(\hat{\boldsymbol{f}}, \boldsymbol{f}) = O(\delta_n^2)$, with $\delta_n^2 = \max(kn^{-1}(\log(n\tau^2\sigma_n^2) + \sigma_n^{-2})^3,\ k/\tau) = O\left(kn^{-1}(\log n)^3\right)$ with $\tau \sim n(\log n)^{-3}$ and fixed σ_n^2, or $\sigma_n^2 \sim 1/\log n$.

An application of Theorem 1 in Shen and Wang (2007) yields the convergence rates of SVM and ψ-learning to be $O\left(\frac{k(k-1)}{2} n^{-1/7}\right)$ and $O\left(\frac{k(k-1)}{2} n^{-1}(\log n)^3\right)$, respectively. Again, the hierarchical structure reduces the order from $k(k-1)/2$ to k as in the linear case.

6. Discussion

This paper proposed a novel large margin method for single-path or partial-path hierarchical classification with mutually exclusive membership at the same level of a hierarchy. In contrast to existing hierarchical classification methods, the proposed method uses inter-class dependencies in a hierarchy. This is achieved through a new concept of generalized functional margins with respect to the hierarchy. By integrating the hierarchical structure into classification, the classification accuracy, or the generalization error defined by hierarchical losses, has been improved over its flat counterpart, as suggested by our theoretical and numerical analyses. Most importantly, the proposed method compares favorably against strong competitors in the large margin classification literature, especially from different settings of our synthetic simulations.

At present, the hierarchical structure is assumed to be correct. However, in applications, some classes may be mislabeled or unlabeled. In such a situation, a further investigation is necessary to generalize the proposed method, and also to allow for novel class detection.

Acknowledgments

This work is supported in part by NSF grants DMS-0604394 and DMS-0906616, NIH grants 1R01GM081535-01 and 2R01GM081535-01, NIH grants HL65462 and R01HL105397, and the Supercomputing Institute at University of Minnesota. The authors thank the action editor and referees for their helpful comments and suggestions.

Appendix A.

The following assumptions are made for Theorem 2.

For a given loss V, we define a truncated $V^T(\boldsymbol{f}, \boldsymbol{Z}) = T \wedge V(\boldsymbol{f}, \boldsymbol{Z})$ for any $\boldsymbol{f} \in \mathcal{F}$ and some truncation constant T, and $e_{V^T}(\boldsymbol{f}, \bar{\boldsymbol{f}}) = E(V^T(\boldsymbol{f}, \boldsymbol{Z}) - V(\bar{\boldsymbol{f}}, \boldsymbol{Z}))$.

Assumption A: There exist constants $0 < \alpha \leq \infty$ and $c_1 > 0$ such that for any small $\varepsilon > 0$,

$$\sup_{\{\boldsymbol{f} \in \mathcal{F}:\ e_{V^T}(\boldsymbol{f}, \boldsymbol{f}^*) \leq \varepsilon\}} |e(\boldsymbol{f}, \bar{\boldsymbol{f}})| \leq c_1 \varepsilon^{\alpha}.$$

Assumption B: There exist constants $\beta \geq 0$ and $c_2 > 0$ such that for any small $\varepsilon > 0$,

$$\sup_{\{\boldsymbol{f} \in \mathcal{F}:\, e_{V^T}(\boldsymbol{f}, \bar{\boldsymbol{f}}) \leq \varepsilon\}} Var(V^T(\boldsymbol{f}, Z) - V(\bar{\boldsymbol{f}}, Z)) \leq c_2 \varepsilon^{\beta}.$$

These assumptions describe local smoothness of $|e(\boldsymbol{f}, \bar{\boldsymbol{f}})|$ and $Var(V^T(\boldsymbol{f}, Z) - V(\bar{\boldsymbol{f}}, Z))$. In particular, Assumption A describes a first moment relationship between the Bayes regret $|e(\boldsymbol{f}, \bar{\boldsymbol{f}})|$ and $e_{V^T}(\boldsymbol{f}, \boldsymbol{f}^*)$. Assumption B is a second moment condition over the neighborhood of $\bar{\boldsymbol{f}}$. The exponents α and β depend on the joint distribution of (X, Y).

We now define a complexity measure of a function space $\mathcal{F}$. Given any $\varepsilon > 0$, denote $\{(f_j^l, f_j^u)\}_{j=1}^m$ as an ε-bracketing function set of $\mathcal{F}$ if for any $f \in \mathcal{F}$, there exists an j such that $f_j^l \leq f \leq f_j^u$ and $\|f_j^l - f_j^u\|_2 \leq \varepsilon; j = 1, \cdots, m$, where $\|f\|_2 = (Ef^2)^{\frac{1}{2}}$ is the L_2-norm. Then the metric entropy with bracketing $H_B(\varepsilon, \mathcal{F})$ is the logarithm of the cardinality of the smallest ε-bracketing set for $\mathcal{F}$. Let $\mathcal{F}^V(t) = \{V^T(\boldsymbol{f}, \boldsymbol{z}) - V(\boldsymbol{f}^*, \boldsymbol{z}) : \boldsymbol{f} \in \mathcal{F}, J(\boldsymbol{f}) \leq J_0 t\}$, where $J(\boldsymbol{f}) = \frac{1}{2}\sum_{j=1}^K \|f_j\|^2$ and $J_0 = \max(J(\boldsymbol{f}^*), 1)$.

Assumption C: For some constants $c_i > 0; i = 3, \ldots, 5$ and $\varepsilon_n > 0$,

$$\sup_{t \geq 2} \phi(\varepsilon_n, s) \leq c_5 n^{1/2}, \quad \phi(\varepsilon_n, s) = \int_{c_3 L}^{c_4^{1/2} L^{\beta/2}} H_B^{1/2}(u, \mathcal{F}^V(s)) du / L, \tag{7}$$

where $L = L(\varepsilon_n, \lambda, s) = \min(\varepsilon_n^2 + \lambda J_0(s/2 - 1), 1)$.

Appendix B.

Proof of Theorem 1: The proof is the same as that of Liu and Shen (2006), and is omitted.

Proof of Lemma 1: When 0-1 loss is used, $l_{0-1}(Y, d(\boldsymbol{X})) = I\{Y \neq d(\boldsymbol{X})\}$. From the sequential decision rule described in Section 2, we know that $y = d(\boldsymbol{x})$ is equivalent to for every $t \in anc(y) \cup \{y\}\}, f_t(\boldsymbol{x}) \geq f_j(\boldsymbol{x}) : j \in sib(t)$. Furthermore, it is also equivalent to $\min\{u_{y,j} : u_{y,j} \in U(\boldsymbol{f}(\boldsymbol{x}), y) = \{u_{y,1}, u_{y,2}, \cdots, u_{y,k_y}\}\} \geq 0$. Therefore, $GE(d) = El_{0-1}(Y, d(\boldsymbol{X})) = EI(u_{min}(f(X), Y) < 0)$ follows.

Proof of Lemma 2: To construct bracket covering for $\mathcal{F}^V(t)$, note that $J(\boldsymbol{f}) \leq J_0 t$ implies $\frac{1}{2}\|f_j\|^2 \leq J_0 t;\ j = 1, \cdots, K$. Furthermore, consider a pairwise difference $f_j - f_{j'}$ with $f_j \in \mathcal{F}_j(t)$ and $f_{j'} \in \mathcal{F}_{j'}(t)$. Let $\{(f_j^{i,l}, f_j^{i,u})_i\}$ be a set of an ε-bracket functions for $\mathcal{F}_j(t)$ in that for any $f_j \in \mathcal{F}_j(t)$, there exists an i such that $f_j^{i,l} \leq f_j \leq f_j^{i,u}$ with $\|f_j^{i,u} - f_j^{i,l}\|_2 \leq \varepsilon;\ j = 1, \cdots, K$. Now construct a set of brackets for $\mathcal{F}^V(t)$. Define $g^u = \max_{\{j' \in sib(j), j \in anc(y) \cup \{y\}\}} v\big(f_j^{i,l} - f_{j'}^{i',u}\big)$ and $g^l = \max_{\{j' \in sib(j), j \in anc(y) \cup \{y\}\}} v\big(f_j^{i,u} - f_{j'}^{i',l}\big)$, where $v(t)$ is $(1-t)_+$ for HSVM and $\psi(t)$ for HPSI. By construction,

$$T \wedge g^l \leq V^T(\boldsymbol{f}, \boldsymbol{z}) = T \wedge \max\{v(f_j - f_{j'}) : j' \in sib(j), j \in anc(y) \cup \{y\}\} \leq T \wedge g^u$$

since $h^T(t) = T \wedge t$ is non-decreasing in t, where $\boldsymbol{z} = (\boldsymbol{x}, y)$. By Lipschitz continuity of $h^T(t)$ in t, $0 \leq (T \wedge g^u - T \wedge g^l) \leq g^u - g^l$, implying

$$\|T \wedge g^u - T \wedge g^l\|_2 \leq \|g^u - g^l\|_2 \leq \sum_{\{j' \in sib(j), j \in anc(y) \cup \{y\}\}} \|(f_j^{i,u} - f_{j'}^{i',l}) - (f_j^{i,l} - f_{j'}^{i',u})\|_2 \leq 2c(\mathcal{H})\varepsilon,$$

with $c(\mathcal{H}) = \sum_{j=0}^K \frac{|chi(j)|(|chi(j)|-1)}{2}$ be the total number of sibling pairs (j, j') in $\mathcal{H}$. It follows that $H_B(2c(\mathcal{H})\varepsilon, \mathcal{F}^V(t)) \leq H_B(2c(\mathcal{H})\varepsilon, \mathcal{F}_1(t))$. The desired result then follows.

To prove that $c(\mathcal{H}) \leq k(k-1)/2$, we count the total number of different paths from the root to a leaf node. On one hand, given each non-leaf node j, there is only one path from the root to the node j but when there are additional $|chi(j)|-1$ paths from the root to its children. An application of this recursively yields that there are $1+\sum_{j\notin\mathcal{L}}(|chi(j)|-1)$ paths from the root of the k leaf nodes. On the other hand, by definition, there are k different paths corresponding to k leaf nodes. Consequently, $k = 1+\sum_{j\notin\mathcal{L}}(|chi(j)|-1)$. For $j \notin \mathcal{L}$, $|chi(j)|-1 \geq 0$. Then

$$\sum_{j\notin\mathcal{L}}(|chi(j)|-1)^2 \leq \Big(\sum_{j\notin\mathcal{L}}(|chi(j)|-1)\Big)^2 = (k-1)^2.$$

This implies

$$2c(\mathcal{H}) = \sum_{j\notin\mathcal{L}}(|chi(j)|-1)^2 + \sum_{j\notin\mathcal{L}}(|chi(j)|-1) \leq (k-1)^2 + k - 1 = k(k-1).$$

This completes the proof.

Proof of Lemma 3: Without loss of generality, assume that the membership is mutually exclusive for the first k classes. The 0-1 loss over K non-exclusive membership classes can be expressed as $\max_{t=1}^{K}(I(Y=t,d(\boldsymbol{X})\neq t)+I(Y\neq t,d(\boldsymbol{X})=t))$, which is the disagreement between the value of Y and that of $d(\boldsymbol{X})$ in $\mathcal{H}$. By assumption, if

$$\max_{t=1}^{k}(I(Y=t,d(\boldsymbol{X})\neq t)+I(Y\neq t,d(\boldsymbol{X})=t)) = 0,$$

then $\max_{t=k+1}^{K}(I(Y=t,d(\boldsymbol{X})\neq t)+I(Y\neq t,d(\boldsymbol{X})=t)) = 0$. On the other hand,

$$\max_{t=1}^{K}(I(Y=t,d(\boldsymbol{X})\neq t)+I(Y\neq t,d(\boldsymbol{X})=t)) \geq \max_{t=1}^{k}(I(Y=t,d(\boldsymbol{X})\neq t)+I(Y\neq t,d(\boldsymbol{X})=t)),$$

which implies that $\max_{t=1}^{K}(I(Y=t,d(\boldsymbol{X})\neq t)+I(Y\neq t,d(\boldsymbol{X})=t)) = 1$ when $\max_{t=1}^{k}(I(Y=t,d(\boldsymbol{X})\neq t)+I(Y\neq t,d(\boldsymbol{X})=t)) = 1$. Consequently

$$l_{0-1}(Y,d(\boldsymbol{X})) = \max_{t=1}^{k}(I(Y=t,d(\boldsymbol{X})\neq t)+I(Y\neq t,d(\boldsymbol{X})=t)) = \sum_{t=1}^{k} I(d(\boldsymbol{X})\neq t)I(Y=t)$$

by exclusiveness of the membership. Finally

$$\begin{aligned} d(\boldsymbol{x}) &= \operatorname*{argmin}_{j=1}^{k} El_{0-1}(Y,d(\boldsymbol{X})=j)|\boldsymbol{X}=\boldsymbol{x}) \\ &= \operatorname*{argmin}_{j=1}^{k}\sum_{t=1}^{k} P(Y=t|\boldsymbol{X}=\boldsymbol{x})I(t\neq j) = \operatorname*{argmin}_{j=1}^{k}\sum_{t\neq j,t=1}^{k} P(Y=t|\boldsymbol{X}=\boldsymbol{x}) \\ &= \operatorname*{argmin}_{j=1}^{k}\Big(1-P(Y=j|\boldsymbol{X}=\boldsymbol{x})\Big) = \operatorname*{argmax}_{j=1}^{k} P(Y=j|\boldsymbol{X}=\boldsymbol{x}). \end{aligned}$$

This completes the proof.

Proof of Lemma 4: The decision function $d(\boldsymbol{x})$, which minimizes $E(l_{0-1}(Y,d(\boldsymbol{X}))|\boldsymbol{X}=\boldsymbol{x})$ for any $\boldsymbol{x}$, thus minimizing its expectation $El_{0-1}(Y,d(\boldsymbol{X}))$.

For $l_\Delta(Y,d(X)) = |anc(Y) \triangle anc(d(\boldsymbol{X}))|$, let $m(j_1,j_2)$ to be $|anc(j_1) \triangle anc(j_2)|$. First note that we have a a length K(size of the tree) vector of bits for each class after introducing the binary 0-1 coding for each node including the ancestor nodes. Therefore $m(\cdot,\cdot)$ satisfies the triangle inequality since it is equivalent to the Hamming distance.

In what follows, we prove that $E(l_\Delta(Y,d(\boldsymbol{X}))|\boldsymbol{X}=\boldsymbol{x}) \le E(l_\Delta(Y,d(\boldsymbol{X}))|\boldsymbol{X}=\boldsymbol{x})$ for any $\boldsymbol{x}$ and classifier $d(\boldsymbol{x})$. Let $\hat{y} = d(\boldsymbol{x})$. By the triangle inequality, $m(y,d(\boldsymbol{x})) - m(y,\hat{y}) \ge -m(d(\boldsymbol{x}),\hat{y})$ for any y. Note that $m(\hat{y},\hat{y}) = 0$ and $m(\hat{y},d(\boldsymbol{x})) = m(d(\boldsymbol{x}),\hat{y}) \ge 0$. Then

$$\begin{aligned}
& E\Big(l_\Delta\big(Y,d(\boldsymbol{X})\big) - l_\Delta\big(Y,d(\boldsymbol{X})\big)\big|\boldsymbol{X}=\boldsymbol{x}\Big) = E\Big(m\big(Y,d(\boldsymbol{x})\big) - m(Y,\hat{y})\big|\boldsymbol{X}=\boldsymbol{x}\Big) \\
= \;& E\bigg(\Big(m\big(Y,d(\boldsymbol{x})\big) - m(Y,\hat{y})\Big)\big(I(Y=\hat{y}) + I(Y\neq\hat{y})\big)\Big|\boldsymbol{X}=\boldsymbol{x}\bigg) \\
= \;& E\bigg(\Big(m\big(\hat{y},d(\boldsymbol{x})\big) - m(\hat{y},\hat{y})\Big)I(Y=\hat{y}) + \Big(m\big(Y,d(\boldsymbol{x})\big) - m(Y,\hat{y})\Big)I(Y\neq\hat{y})\Big|\boldsymbol{X}=\boldsymbol{x}\bigg) \\
\ge \;& E\Big(m\big(\hat{y},d(\boldsymbol{x})\big)I(Y=\hat{y})|\boldsymbol{X}=\boldsymbol{x}\Big) - E\Big(m\big(d(\boldsymbol{x}),\hat{y}\big)I(Y\neq\hat{y})\big|\boldsymbol{X}=\boldsymbol{x}\Big) \\
= \;& m\big(\hat{y},d(\boldsymbol{x})\big)\big(P(Y=\hat{y}|\boldsymbol{X}=\boldsymbol{x}) - P(Y\neq\hat{y}|\boldsymbol{X}=\boldsymbol{x})\big) \ge 0.
\end{aligned}$$

The last inequality follows from the fact that $\hat{y} = \text{argmax}_{j\in\mathcal{L}} P(Y=j|\boldsymbol{X}=x)$ and $P(Y=\hat{y}|\boldsymbol{X}=\boldsymbol{x}) \ge 1/2 \ge P(Y\neq\hat{y}|\boldsymbol{X}=\boldsymbol{x})$ by the assumption of dominating class. The desired result then follows.

Proof of Lemma 5: We prove the case of $v(z) = (1-z)_+$ for HSVM. Denote by $\hat{\boldsymbol{f}}(\boldsymbol{x})$ a minimizer of $E(v(u_{min}(\boldsymbol{f}(\boldsymbol{X}),Y))|\boldsymbol{X}=\boldsymbol{x})$ for any $\boldsymbol{x}$. At a given $\boldsymbol{x}$, without loss of generality, assume $p_j(\boldsymbol{x}) > 0$; $\forall 1 \le j \le k$. Let $\hat{j} = d^H(\hat{\boldsymbol{f}}(\boldsymbol{x}))$ and $\hat{u} = u_{min}(\hat{\boldsymbol{f}}(\boldsymbol{x}),\hat{j})$. By definition, $\hat{f}_{j'}(\boldsymbol{x}) - \hat{f}_{j''}(\boldsymbol{x}) \ge \hat{u} \ge 0$, $\forall j' \in anc(\hat{j})$ and $j'' \in sib(j')$. First consider the case of $\hat{u} > 0$. For all other leaf node $j \neq \hat{j}$, there exists $j_a \in anc(j)$ and $\hat{j}_a \in anc(\hat{j})$ such that $j_a \in sib(\hat{j}_a)$. Then $u_{min}(\hat{\boldsymbol{f}}(\boldsymbol{x}),j) \le \hat{f}_{j_a}(\boldsymbol{x}) - \hat{f}_{\hat{j}_a}(\boldsymbol{x}) \le -u_{min}(\hat{\boldsymbol{f}}(\boldsymbol{x}),\hat{j}) = -\hat{u}$, by the fact that $u_{min}(\hat{\boldsymbol{f}}(\boldsymbol{x}),\hat{j}) \le -(\hat{f}_{j_a}(\boldsymbol{x}) - \hat{f}_{\hat{j}_a}(\boldsymbol{x}))$. Now we prove the equality of $u_{min}(\hat{\boldsymbol{f}}(\boldsymbol{x}),j) \le -\hat{u}$ holds through construction of $\boldsymbol{f}'$: $f'_{j'}(\boldsymbol{x}) - f'_{j''}(\boldsymbol{x}) = \hat{u}$, and $f'_j(\boldsymbol{x}) = 0, \forall j \notin sib \circ anc(\hat{j})$. By construction, $u_{min}(\boldsymbol{f}'(\boldsymbol{x}),j) = -\hat{u}$, for $1 \le j \le k$, $j \neq \hat{j}$, and $u_{min}(\boldsymbol{f}'(\boldsymbol{x}),\hat{j}) = \hat{u}$. Note that

$$\begin{aligned}
& E(v(u_{min}(\hat{\boldsymbol{f}}(\boldsymbol{X}),Y))|\boldsymbol{X}=\boldsymbol{x}) - E(v(u_{min}(\boldsymbol{f}'(\boldsymbol{X}),Y))|\boldsymbol{X}=\boldsymbol{x}) \\
= \;& \sum_{1\le j\le k; j\neq\hat{j}} p_j(\boldsymbol{x})(v(u_{min}(\hat{\boldsymbol{f}}(\boldsymbol{x}),j)) - v(-\hat{u})) \ge 0.
\end{aligned}$$

By the fact that $\hat{\boldsymbol{f}}(\boldsymbol{x})$ is the minimizer, for $1 \le j \le k$, $j \neq \hat{j}$, $v(u_{min}(\hat{\boldsymbol{f}}(\boldsymbol{x}),j)) - v(-\hat{u}) = 0$, then $u_{min}(\hat{\boldsymbol{f}}(\boldsymbol{x}),j) = -\hat{u}$. Moreover, for the Bayes rule $d(\boldsymbol{x})$, if $\hat{j} \neq d(\boldsymbol{x})$, we construct $\boldsymbol{f}^*$ such that $u_{min}(\boldsymbol{f}^*(\boldsymbol{x}),d(\boldsymbol{x})) = \hat{u}$, and $u_{min}(\boldsymbol{f}^*(\boldsymbol{x}),j) = -\hat{u}$, for any leaf node $j \neq d(\boldsymbol{x})$, similar as above. This implies that

$$\begin{aligned}
& E(v(u_{min}(\hat{\boldsymbol{f}}(\boldsymbol{X}),Y))|\boldsymbol{X}=\boldsymbol{x}) - E(v(u_{min}(\boldsymbol{f}^*(\boldsymbol{X}),Y))|\boldsymbol{X}=\boldsymbol{x}) \\
= \;& (p_{\hat{j}}(\boldsymbol{x})v(\hat{u}) + p_{d(\boldsymbol{x})}(\boldsymbol{x})v(-\hat{u})) - (p_{\hat{j}}(\boldsymbol{x})v(-\hat{u}) + p_{d(\boldsymbol{x})}(\boldsymbol{x})v(\hat{u})) \\
= \;& (p_{\hat{j}}(\boldsymbol{x}) - p_{d(\boldsymbol{x})}(\boldsymbol{x}))(v(\hat{u}) - v(-\hat{u})) > 0,
\end{aligned}$$

because $p_{\hat{j}}(\boldsymbol{x}) < p_{d(\boldsymbol{x})}(\boldsymbol{x})$ and $\hat{u} > 0$. This contradicts the fact that $\hat{\boldsymbol{f}}(\boldsymbol{x})$ is the minimizer. Consequently, $\hat{j} = d(\boldsymbol{x})$. For the case of $\hat{u} = 0$, it can be shown that $u_{min}(\hat{\boldsymbol{f}}(\boldsymbol{x}),j) = 0, \forall j = 1,\cdots,k$, and $\hat{f}_j(\boldsymbol{x}) = 0, \forall j = 1,\cdots,K$, which reduces to the trivial case $\hat{\boldsymbol{f}}(\boldsymbol{x}) = \boldsymbol{0}$.

For HPSI, the proof is the same as that of Theorem 2 in Liu and Shen (2006), and is omitted.

Proof of Theorem 2: The proof is similar to that in Shen and Wang (2007) and is omitted.

Proof of Corollary 1: The $O_p(\cdot)$ result follows from the exponential bound in Theorem 2. To see the risk result, note that

$$\delta_n^{-2\alpha} E e(\hat{\boldsymbol{f}}, \bar{\boldsymbol{f}}) = \int_0^\infty P(e(\hat{\boldsymbol{f}}, \bar{\boldsymbol{f}}) > (\delta_n^{2\alpha} t)^{1/2\alpha}) dt.$$

The result then follows.

The primal and the dual of (2) for HSVM: The primal and the dual for HSVM can be obtained from those of HPSI below, with $\nabla \hat{\boldsymbol{w}}_j^{(m-1)} = 0$ and $\nabla \hat{b}_j^{(m-1)} = 0$; $j = 1, \cdots, K$.

The primal and the dual of (3) for HPSI: The primal of (3) is

$$\underset{\boldsymbol{f}}{\operatorname{argmin}} \frac{1}{2} \sum_{j=1}^K \|\boldsymbol{w}_j\|^2 + C \sum_{i=1}^n \xi_i - \sum_{j=1}^K \langle \nabla \hat{\boldsymbol{w}}_j^{(m-1)}, \boldsymbol{w}_j \rangle - \sum_{j=1}^K \langle \nabla \hat{b}_j^{(m-1)}, b_j \rangle, \tag{8}$$

subject to $\xi_i > 0$, $(f_j(x_i) - f_t(x_i)) + \xi_i \geq 1$, $(j,t) \in Q(y_i) = \{(j,t) : t \in sib(j), j \in \{y_i\} \cup anc(y_i)\}$, and $\sum_{\{j \in chi(s), s \notin \mathcal{L}\}} f_j(\boldsymbol{x}_i) = 0$; $i = 1, \cdots, n$, $s = 1, \cdots, K$.

To solve (8), we employ the Lagrange multipliers: $\alpha_i \geq 0$, $\beta_{i,j,t} \geq 0$ and $\delta_{i,s} \geq 0$ for each constraint of (8). Then (8) is equivalent to:

$$\begin{aligned} \max_{\alpha_i, \beta_{i,j,t}, \delta_{i,s}} L \quad = \quad & \frac{1}{2} \sum_{j=1}^K \|\boldsymbol{w}_j\|^2 + C \sum_{i=1}^n \xi_i - \sum_{j=1}^K \langle \nabla \hat{\boldsymbol{w}}_j^{(m-1)}, \boldsymbol{w}_j \rangle - \sum_{j=1}^K \langle \nabla \hat{b}_j^{(m-1)}, b_j \rangle + \\ & \sum_{(j,t) \in Q(y_i): i = 1, \cdots, n} \beta_{i,j,t} \Big(1 - \big((\langle \boldsymbol{w}_j, \boldsymbol{x}_i \rangle + b_j) - (\langle \boldsymbol{w}_t, \boldsymbol{x}_i \rangle + b_t) \big) - \xi_i \Big) \\ & - \sum_{i=1}^n \alpha_i \xi_i + \sum_{(i,s): i = 1, \cdots, n; s \notin \mathcal{L}} \delta_{i,s} \sum_{j \in chi(s)} (\langle \boldsymbol{w}_j, \boldsymbol{x}_i \rangle + b_j). \end{aligned} \tag{9}$$

By letting the partial derivatives be zero, we have that

$$\frac{\partial L}{\partial \boldsymbol{w}_j} = 0, \frac{\partial L}{\partial \xi_i} = 0, \frac{\partial L}{\partial b_j} = 0; i = 1, \cdots, n, j = 1, \cdots, K. \tag{10}$$

implying that $\alpha_i \geq 0; i = 1, \cdots, n$, and

$$\sum_{(j,t) \in Q(y_i)} \beta_{i,j,t} \leq C; i = 1, \cdots, n. \tag{11}$$

After substituting (10) in (9), we obtain a quadratic form of L in terms of $\{\alpha_i, \beta_{i,j,t}, \delta_{i,s}\}$ Maximizing L subject to $\beta_{i,j,t} \geq 0$; $i = 1, \cdots, n$; $(j,t) \in Q(y_i)$, (10) and (11) yields the solution of $\{\alpha_i, \beta_{i,j,t}, \delta_{i,s}\}$. The solution of $\boldsymbol{w}_j$ and ξ_i's can be derived from (10). The solution of b_j is derived from Karush-Kuhn-Tucker's condition: $\beta_{i,j,t} \Big(1 - \big((\langle \boldsymbol{w}_j, \boldsymbol{x}_i \rangle + b_j) - (\langle \boldsymbol{w}_t, \boldsymbol{x}_i \rangle + b_t) \big) - \xi_i \Big) = 0$, $\alpha_i \xi_i = 0$, and $\delta_{i,s} \sum_{j \in chi(s)} (\langle \boldsymbol{w}_j, \boldsymbol{x}_i \rangle + b_j) = 0$, for all suitable i, j, t and s. In case of these conditions are not applicable to b_j's, we substitute the solution of $\boldsymbol{w}_j$'s in (8), and solve b_j's through linear programming.

References

L. An and P. Tao. Solving a class of linearly constrained indefinite quadratic problems by d.c. algorithms. *J. Global Optimization*, 11:253–285, 1997.

K. Astikainen, L. Holm, S. Szedmak E. Pitknen, and J. Rousu. Towards structured output prediction of enzyme function. *BMC Proceedings*, 2(S4):S2, 2008.

B. Boser, I. Guyon, and V. N. Vapnik. A training algorithm for optimal margin classifiers. *Proc. Fifth Ann. Conf on Computat. Learning Theory Pittsburgh, PA*, pages 144–152, 1992.

L. Cai and T. Hofmann. Hierarchical document categorization with support vector machines. *CIKM-04, Washington, DC*, 2004.

N. Cesa-Bianchi and G. Valentini. Hierarchical cost-sensitive algorithms for genome-wide gene function prediction. *MLSB 09: The 3rd International Workshop on Machine Learning in Systems Biology 2009*, 2009.

N. Cesa-Bianchi, A. Conconi, and C. Gentile. Regret bounds for hierarchical classification with linear-threshold functions. *Proc. the 17th Ann. Conf. on Computat. Learning Theory*, pages 93–108, 2004.

N. Cesa-Bianchi, C. Gentile, and L. Zaniboni. Hierarchical classification: Combining bayes with svm. *Proc. of the 23rd Int. Conf. on Machine Learning, ACM Press (2006)*, pages 177–184, 2006.

M. N. Davies, A. Secker, A. A. Freitas, M. Mendao, J. Timmis, and D. R. Flower. On the hierarchical classification of g protein-coupled receptors. *Bioinformatics*, 23(23):3113–3118, 2007.

O. Dekel, J. Keshet, and Y. Singer. An efficient online algorithm for hierarchical phoneme classification. *Proc. the 1st Int. Workshop on Machine Learning for Multimodal Interaction*, pages 146–158, 2004.

L. Dong, E. Frank, and S. Kramer. Ensembles of balanced nested dichotomies for multi-class problems. *Lecture Notes in Computer Science*, 3721/2005:84–95, 2005.

C. Gu. Multidimension smoothing with splines. *Smoothing and Regression: Approaches, Computation and Application*, 2000.

Y. Guan, C. Myers, D. Hess, Z. Barutcuoglu, A. Caudy, and O. Troyanskaya. Predicting gene function in a hierarchical context with an ensemble of classifiers. *Genome Biology*, 9(S2), 2008.

T. Hughes, M. Marton, A. Jones, C. Roberts, R. Stoughton, C. Armour, H. Bennett, E. Coffey, H. Dai, Y. He, M. Kidd, A. King, M. Meyer, D. Slade, P. Lum, S. Stepaniants, D. Shoemaker, D. Gachotte, K. Chakraburtty, J. Simon, M. Bard, and S. Friend. Functional discovery via a compendium of expression profiles. *Cell*, 102:109–126, 2000.

T. Jaakkola, M. Diekhans, and D. Haussler. Using the fisher kernel method to detect remote protein homologizes. *In Proc. the Seventh Int. Conf. on Intelligent Systems for Molecular Biology*, pages 149–158, 1999.

T. Joachims. Text categorization with support vector machines: learning with many relevant features. *Proc. of the 10th European Conf. on Machine Learning (ECML1998)*, 1398:117–142, 1998.

A. N. Kolmogorov and V. M. Tihomirov. ε-entropy and ε-capacity of sets in function spaces. *Uspekhi Mat. Nauk.*, 14:3–86, 1959. In Russian. English translation, *Ameri. Math. Soc. transl.* **2 , 17**, 277-364. (1961).

D. Lewis. Naive (bayes) at forty: The independence assumption in information retrieval. *Proc. of the 10th European Conf. on Machine Learning (ECML1998)*, pages 4–15, 1998.

Y. Lin, Y. Lee, and G. Wahba. Support vector machines for classification in nonstandard situations. *Machine Learning*, 46:191–202, 2002.

S. Liu, X. Shen, and W. Wong. Computational development of ψ-learning. *Proc. SIAM 2005 Int. Data Mining Conf.*, pages 1–12, 2005.

Y. Liu and X. Shen. Multicategory ψ-learning. *J. Amer. Statist. Assoc.*, 101:500–509, 2006.

H. W. Mewes, D. Frishman, U. G'ldener, G. Mannhaupt, K. Mayer, M. Mokrejs, B. Morgenstern, M. M'nsterkoetter, S. Rudd, and B. Weil. Mips: a database for genomes and protein sequences. *Nuclerc Acids Res*, 30:31–34, 2002.

G. Obozinski, G. Lanckriet, C. Grant, M. Jordan, and W. Noble. Consistent probabilistic output for protein function prediction. *Genome Biology*, 9(S6), 2008.

J. Rousu, C. Saunders, S. Szedmak, and J. Shawe-Taylor. Kernel-based learning of hierarchical multilabel classification models. *J. Mach. Leaning Res.*, 7:1601–1626, 2006.

B. Shahbaba and R. Neal. Improving classification when a class hierarchy is available using a hierarchy-based prior. *Bayesian Analysis*, 2:221–238, 2007.

X. Shen and L. Wang. Generalization error for multi-class margin classification. *Electronic J. of Statist.*, 1:307–330, 2007.

X. Shen, G. Tseng, X. Zhang, and W. Wong. On ψ-learning. *J. Amer. Statist. Assoc.*, 98:724–734, 2003.

I. Tsochantaridis, T. Hofmann, T. Joachims, and Y. Altun. Support vector machine learning for interdependent and structured output spaces. *Proc. the 21st Int. Conf. on Machine Leaning*, 2004.

G. Valentini and M. Re. Weighted true path rule: a multilabel hierarchical algorithm for gene function prediction. *The 1st International Workshop on learning from Multi-Label Data, ECML/PKDD 2009*, 2009.

V. Vapnik. *Statistical Learning Theory*. Wiley, New York, NY, 1998.

Y. Yang and X. Liu. A reexamination of text categorization methods. *Proc. the 22nd Annual Int. ACM SIGIR Conf. on Research and Development in Information Retrieval*, pages 42–49, 1999.

J. Zhu and T. Hastie. Kernel logistic regression and the import vector machine. *J. Comput. and Graph. Statist.*, 14:185–205, 2005.

A. Zimek, F. Buchwald, E. Frank, and S. Kramer. A study of hierarchical and flat classification of proteins. *IEEE/ACM Transactions on Computational Biology and Bioinformatics 2008*, 2008.

Journal of Machine Learning Research 12 (2011) 2749-2775 Submitted 10/09; Revised 6/11; Published 10/11

Non-Parametric Estimation of Topic Hierarchies from Texts with Hierarchical Dirichlet Processes

Elias Zavitsanos[*] IZAVITS@IIT.DEMOKRITOS.GR
Georgios Paliouras PALIOURG@IIT.DEMOKRITOS.GR
Institute of Informatics and Telecommunications
NCSR "Demokritos"
Patriarhou Gregoriou and Neapoleos Street
15310 Athens, Greece

George A. Vouros GEORGEV@AEGEAN.GR
Department of Information and Communication Systems Engineering
University of the Aegean
Karlovassi, 83200 Samos Island, Greece

Editor: David Blei

Abstract

This paper presents hHDP, a hierarchical algorithm for representing a document collection as a hierarchy of latent topics, based on Dirichlet process priors. The hierarchical nature of the algorithm refers to the Bayesian hierarchy that it comprises, as well as to the hierarchy of the latent topics. hHDP relies on nonparametric Bayesian priors and it is able to infer a hierarchy of topics, without making any assumption about the depth of the learned hierarchy and the branching factor at each level. We evaluate the proposed method on real-world data sets in document modeling, as well as in ontology learning, and provide qualitative and quantitative evaluation results, showing that the model is robust, it models accurately the training data set and is able to generalize on held-out data.

Keywords: hierarchical Dirichlet processes, probabilistic topic models, topic distributions, ontology learning from text, topic hierarchy

1. Introduction

In this paper we address the problem of modeling the content of a given document collection as a hierarchy of latent topics given no prior knowledge. These topics represent and capture facets of content meaning, by means of multinomial probability distributions over the words of the term space of the documents. The assignment of documents to latent topics without any preclassification is a powerful text mining technique, useful among others for ontology learning from text and document indexing.

In the context of this modeling problem, probabilistic topic models (PTMs) have attracted much attention. While techniques for terminology extraction and concept identification from text rely on the identification of representative terms using various frequency measures, such as the TF/IDF (Salton and McGill, 1986) or C/NC value (Frantzi et al., 2000), PTMs aim to discover topics that are soft clusters of words, by transforming the original term space into a latent one of meaningful features (topics).

∗. Also in the Department of Information and Communication Systems Engineering, University of the Aegean, Greece.

Much work on PTMs focuses on a flat clustering of the term space into topics, while the creation of a hierarchical structure of topics without user involvement or pre-defined parameters still remains a challenging task. The goal of discovering a topic hierarchy that comprises levels of topic abstractions is different from conventional hierarchical clustering. The internal nodes of this type of hierarchy reflect the topics, which correspond to the shared terminology or vocabulary between documents. In contrast, hierarchical clustering usually groups data points, for instance documents, resulting in internal nodes that constitute cluster summaries. Conventional techniques such as agglomerative clustering, allow objects to be grouped together based on a similarity measure, but the hierarchy is generally the result of hard clustering. This form of clustering limits the applicability of the techniques, since a document is assigned to only one topic and may not be retrieved upon a search on a related topic.

Furthermore, conventional clustering models texts based explicitly on syntax. It tends to cluster words that appear in similar local contexts. On the other hand, topic models attempt to capture through syntax, latent semantics. They cluster words that appear in a similar global context, in the sense that they try to generalize beyond their place of appearance in a text collection, in order to reflect their intended meaning.

The role of hierarchical topic models regarding text modeling and natural language processing (NLP) is very important. The hierarchical modeling of topics allows the construction of more accurate and predictive models than the ones constructed by flat models. Models of the former type are more probable to predict unseen documents, than the latter. In most text collections, such as web pages, a hierarchical model, for instance a web directory, is able to describe the structure and organization of the document collection more accurately than flat models. This, however, ultimately depends on the nature of the data set and the true generative process of the documents themselves. Assuming that the higher levels of the hierarchy capture generic topics of a particular domain, while lower-level ones focus on particular aspects of that domain, it is expected that a hierarchical probabilistic topic model would be able to "explain" or could have generated the data set. In other words, the likelihood of such a model given the data set would probably be higher than the likelihood of other flat models (Mimno et al., 2007; Li et al., 2007; Li and McCallum, 2006).

Despite recent activity in the field of HPTMs, determining the hierarchical model that best fits a given data set, in terms of the structure and size of the learned hierarchy, still remains a challenging task and an open issue. In this paper, we propose a method that deals with some of the limitations of the current models, regarding the representation of input data as latent topics. In particular, we aim to infer a hierarchy of topics and subtopics, such that each topic is more general than its subtopics, in the sense that if a document can be indexed by any of the subtopics it should also be indexed by the topic itself. Moreover, we demand to infer the hierarchy without making any assumption either about the number of topics at any level of the hierarchy, or about the height of the hierarchy. The proposed method, given a collection of text documents, produces a hierarchical representation in the form of a topic hierarchy, adopting a nonparametric Bayesian approach. The resulting hierarchy specifies each topic as a multinomial probability distribution over the vocabulary of the documents. Moreover, internal nodes are also represented as multinomial probability distributions over the subtopics of the hierarchy. In addition to the basic model, we also present a variant that produces a topic hierarchy, by modeling the vocabulary only at the leaf level and considering topics in the inner levels to be multinomial distributions over subtopics. Although the evaluation of such models is also an open issue, we demonstrate the effectiveness of the model in different tasks through an extensive evaluation, providing qualitative and quantitative results.

In what follows, we start by a quick review of the family of probabilistic topic models and hierarchical models (Section 2). Section 3 presents the proposed method, namely topic hierarchies of hierarchical Dirichlet processes (hHDP), along with its variant. Section 4 provides an extensive evaluation of hHDP, including comparisons to other models and applications to different tasks, while Section 5 summarizes the paper and presents future directions.

2. Hierarchical Probabilistic Topic Models

Probabilistic topic models (PTMs) (Griffiths and Steyvers, 2002) are generative models for documents. Documents are assumed to be mixtures of topics and topics are probability distributions over the words of some vocabulary. The vocabulary may comprise all the words that appear in the documents or a part of them, for example excluding the stop-words. PTMs are based on the De Finetti theorem (Finetti, 1931), which states that an exchangeable sequence of random variables is a mixture of independent and identically distributed random variables. In the case of text data, PTMs treat documents as "bag-of-words." The words in the documents are infinitely exchangeable without loss of meaning, and thus, the joint probability underlying the data is invariant to permutation. Based on this assumption of exchangeability, the meaning of documents does not depend on the specific sequence of the words, that is, the syntax, but rather on their "ability" to express specific topics either in isolation or in mixture. Given the latent variables, (the topics), the words are assumed to be conditionally independent and identically distributed in the texts.

Figure 1 represents the underlying idea of the generative nature of PTMs. Topics, represented as clouds, are probability distributions over words (puzzle pieces) of a predefined vocabulary. According to the mixture weights that reflect the probability of a topic to participate in a document, words are sampled from the corresponding topics, in order for documents to be generated.

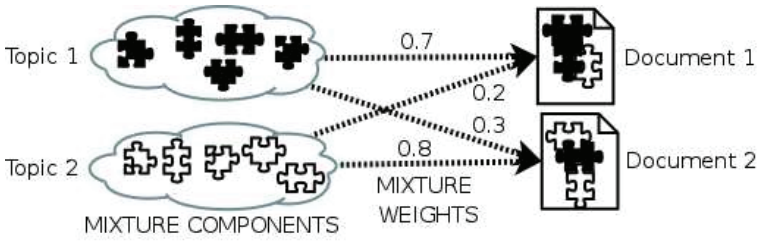


Figure 1: The generative nature of PTMs: Documents are mixtures of topics. Topics are probability distributions over words (puzzle pieces). The probability of participation of a topic in a document is defined by the mixture weights. Inspired by Steyvers and Griffiths (2007).

In the rest of the paper, we will refer to the document collection as D, consisting of $d_1, d_2, ..., d_N$ documents. The set of the latent topics will be defined as T, consisting of $t_1, t_2, ..., t_K$ topics. We will refer to the distribution of topics as θ_K, indicating the dimensionality K of the distribution, and finally, ϕ_V will stand for the distribution of the words of the vocabulary V.

Following the principles of PTMs, the generative model of probabilistic latent semantic analysis (PLSA) (Hofmann, 2001) specifies a simple generative rule for the words in a document d_i, according to which, each word of a training document d_i comes from a randomly chosen topic t_i. The topics are drawn from a document-specific distribution over topics θ_K, and there exists one such

distribution for each d_i. Hence, the set of the training documents D defines an empirical distribution over topics. In PLSA, the observed variable d_i is actually an index into the training set D, and thus, there is no natural way for the model to handle previously unseen documents, except through marginalization (Blei et al., 2003).

The model of PLSA has been extended by latent Dirichlet allocation (LDA) (Blei et al., 2003). The generative model of LDA, being a probabilistic model of a corpus, represents each d_i as random mixture over latent topics T. The mixture indicator is selected once per term, rather than once per document as in PLSA. The estimated topics are represented as multinomial probability distributions over the terms of the documents, while each d_i is represented as a Dirichlet random variable θ, the dimensionality of which, is predefined and equal to the number of estimated latent topics. In contrast to PLSA, LDA states that each word of both the observed and unseen documents is generated by a randomly chosen topic, which is drawn from a distribution with a randomly chosen parameter. This parameter is sampled once per document from a smooth distribution over topics.

A question that usually arises when using models like LDA is how many topics the estimated model should have, given the document collection. The problem is harder when multiple parameters are shared among documents, as in LDA. The problem is addressed by sharing a discrete base distribution among documents. A hierarchical Dirichlet process (HDP) creates such a discrete base distribution for the document Dirichlet processes (DPs) by sampling from another DP. In such a Bayesian hierarchy, the root DP uses the Dirichlet distribution of the topics as a base distribution and each document samples from it.

Although LDA is a true generative probabilistic model for documents and HDP is a convenient mechanism for inferring the number of topics, relations of any type or correlations between the estimated topics are not taken into account. In fact, a flat and soft clustering of the term space of the documents into topics is provided. Thus, there is a need for hierarchical models that are able to capture relations between the latent topics in order to represent common shared structure, as explained in Section 1.

A method for producing a tree-like structure of latent topics is presented in Gaussier et al. (2002), as an extension of the PLSA model. According to hierarchical probabilistic latent semantic analysis (HPLSA), the data set D is assumed to have been generated by a hierarchical model. For each d_i, a document class is picked from a predefined number of classes, with some probability. Then, a d_i is chosen based on the conditional probability of a document given the class. Again, given the class, a topic t_i is sampled for that d_i. Finally, a word is generated given the sampled topic t_i. A class here represents a group of documents sharing some common thematic feature. According to this model, documents and words are conditionally independent given the class. In a typical hierarchy, documents are assigned to classes at the leaves of the hierarchy, while words are sampled from topics which occupy non-leaf nodes of the hierarchy. The number of classes actually defines the number of leaves of the hierarchy. The model extends PLSA in the sense that if one topic per class is sampled, then the result is the flat clustering of PLSA. If on the other hand, a single topic is sampled for more than one class, then it is placed on a higher level and represents shared knowledge between these classes. However, the model inherits known problems of PLSA, such as the large number of parameters that need to be estimated, which grow linearly with the size of the corpus, a problem that LDA seems to deal with, since the latter treats the distribution θ_K as a hidden random variable, rather than a large set of individual parameters which are explicitly linked to the training set.

Another approach to capturing relations between topics is the correlated topic models (CTM) (Blei and Lafferty, 2006), an extension of LDA. The generative process of this model is identical to that of LDA, with the exception that the topic proportions are drawn from a logistic normal distribution, rather than a Dirichlet as in the case of LDA. The parameters of this distribution include a covariance matrix, the entries of which specify the correlations between pairs of topics. Correlations are introduced by topics that appear in the same context, in the sense that they appear together in documents (or parts of documents). The advantage of this model is that the covariance matrix may include positive covariance between two topics that co-occur frequently and negative between two topics that co-occur rarely, while with the Dirichlet approach, we actually express the expectation of each topic to occur, according to the weights of the mixture proportions, and how much we expect any given document to follow these proportions. In CTM only pairwise correlations between topics are modeled. Hence, the number of parameters grows as the square of the number of topics.

The Pachinko allocation model (PAM) (Li and McCallum, 2006) deals with some of the problems of CTM. PAM uses a directed acyclic graph (DAG) structure to represent and learn arbitrary, nested and possibly sparse topic correlations. PAM connects the words of the vocabulary V and topics T on a DAG, where topics occupy the interior nodes and the leaves are words. Each topic t_i is associated with a Dirichlet distribution of dimension equal to the number of children of that topic. The four-level PAM, which is presented in Li and McCallum (2006), is able to model a text collection through a three-level hierarchy of topics with arbitrary connections between them. However, PAM is unable to represent word distributions as parents of other word distributions and also requires the length of the path from the root node to the leaves to be predefined.

The hierarchical latent Dirichlet allocation (hLDA) model (Blei et al., 2004) was the first attempt to represent the distribution of topics as a tree-structure by providing at the same time uncertainty over the branching factor at each level of the tree. In hLDA, each document is modeled as a mixture of L topics defined by θ_L proportions along a path from the root topic to a leaf. Therefore, each document d_i is generated by the topics along a single path of this tree. Hence, each d_i is about a specific topic (a leaf topic) and its abstractions along the path to the root. Multiple inheritance, in the sense of assigning more than one topic to a super-topic, is not modeled. When estimating the model from data, for each d_i, the sampler chooses an existing or a new path through the tree and assigns each word to a topic along the chosen path. Thus, both internal and leaf topics generate words for new documents. In order to learn the structure of the tree, a nested Chinese restaurant process (nCRP) is used as a prior distribution. Assuming that the depth (L) of the hierarchy is provided a priori, the nCRP prior actually controls the branching factor at each level of the hierarchy. It expresses the uncertainty about possible L-level trees and thus, the problem of modeling the corpus is reduced to finding a good, in the sense of maximum likelihood, L-level tree among them.

Aiming to support multiple inheritance between topics, and extending PAM to express word distributions as parents of other word distributions, the work in Mimno et al. (2007) presents the hierarchical Pachinko allocation model (HPAM), in which every node is associated with a distribution over the vocabulary of the text collection. There are actually two variants of the model. In the first variant, each path through the DAG is associated with a multinomial distribution on the levels of the path, which is shared by all documents. In the second one, this distribution does not exist, but the Dirichlet distribution of each internal node has one extra "exit" dimension, which corresponds to the event that a word is produced directly by the internal node, without reaching the leaf topics of the DAG. The three-level model that is presented in Mimno et al. (2007) comprises a root topic, a level of super-topics and a level of sub-topics and it uses $(T + 1)$ Dirichlet distributions to model

the text collection. One distribution incorporates a hyper-parameter α_0 and serves as a prior over the super-topics. The remaining T distributions incorporate a hyper-parameter α_T, which serves as a prior over their sub-topics. The difference between the priors α_0 and α_T is that they produce different distributions θ_0 and θ_T over super-topics and subtopics respectively.

While the models belonging in the PAM family provide a powerful means to describe inter-topic correlations, they have the same practical difficulty as many other topic models in determining the number of topics at the internal levels. For this purpose, a non-parametric Bayes version of PAM has been presented in Li et al. (2007). This model is actually a combination of the hLDA model, in the sense of determining the number of topics T at the internal levels, and of the four-level PAM (Li and McCallum, 2006). Each topic t_i is modeled by a Dirichlet process and the Dirichlet processes at each level are further organized into a hierarchical Dirichlet process (HDP), which is used to estimate the number of topics at this level. Apart from this, the model follows the basic PAM principles. During the generation of a document, after sampling the multinomial distributions over topics from the corresponding HDPs, a topic path is sampled repeatedly according to the multinomials for each word in the document d_i. The resulting hierarchy is limited to three levels and comprises the root topic, the next level of super-topics and the final level of sub-topics, which are the ones that are able to generate words.

Representing all topics as multinomial distributions over words is more appealing, than representing only the leaf topics. For this purpose, the work in Zavitsanos et al. (2008) uses the LDA model iteratively to produce layers of topics and then establishes hierarchical relations between them, based on conditional independences, given candidate parent topics. The branching factor at each level is decided by the number of discovered relations, since topics that are not connected to others are disregarded. The issue of the depth of the hierarchy is addressed in that work by measuring the similarity of the newly generated topics to the existing ones. However, the number of the generated topics at each level is predefined.

In summary, some topic models support a latent hierarchy of topics, but allow the generation of words only at the leaf level. Others are able to generate words at each level, but depend on a predefined depth of the hierarchy. In particular, hLDA is able to infer the branching factor at each level, but still requires the depth of the hierarchy to be known a priori. In addition, in contrast to the simple LDA, in the case of hLDA, documents can only access the topics that lie across a single path in the learned tree. Hence, LDA, which places no such restrictions in the mixture of topics for each document, can be significantly more flexible than hLDA. The models belonging in the PAM family seem to be able to address these issues, especially the non-parametric Bayes version of PAM (Li et al., 2007) that exploits some of the advantages of hLDA. However, the fact that the resulting hierarchy comprises three levels and produces words only at the leaves is limiting. It seems possible to extend the hierarchy to more levels, but this would require the depth to be known a priori and would impose an increase on the number of parameters to be estimated. Finally, parameters such as the number of topics or the number of levels need to be estimated using cross-validation, which is not efficient even for non-hierarchical topic models like LDA. Table 1 summarizes the properties of the aforementioned models.

The evaluation of topic models is also an open issue. The majority of the work reviewed in this section assesses the inferred hierarchy on the basis of how "meaningful" the latent topics are to humans. In this spirit, new evaluation measures (Chang et al., 2009) have been proposed that try to capture aspects of how humans evaluate topic models and especially the inferred hierarchy. Thus,

Model	Topic hierarchy	Infer number of topics	Infer number of levels	Multiple inheritance	Generate words at all nodes
PLSA	×	×	×	×	✓
LDA	×	×	×	×	✓
HDP	×	✓	×	×	✓
HPLSA	✓	×	×	×	✓
CTM	×	×	×	×	✓
PAM	✓	×	×	✓	×
hLDA	✓	✓	×	×	✓
HPAM	✓	×	×	✓	✓
NPPAM	✓	✓	×	✓	×

Table 1: Comparison of topic models. The first column is the acronym of the model. The second column shows whether the model is able to organize the topics hierarchically. The third and fourth columns depict the ability of the model to infer the number of topics and levels respectively. The last two columns indicate whether the model's topics share subtopics, and whether the model produces words at all nodes.

the emphasis is on how topic models infer the latent structure of the input documents, rather than on how well they generate documents. Based on this observation, we propose an algorithm that:

- Determines the depth of the learned hierarchy.
- Infers the number of topics at each level of the hierarchy.
- Allows sharing of topics among different documents.
- Allows topics to share subtopics.
- Allows a topic at any level of the hierarchy to be specified as a distribution over terms.
- Has a non-parametric Bayesian nature and thus exhibits all the advantages of such techniques.

In addition, we present a variant that models only the leaf levels as probability distributions over words and results in a hierarchical topic clustering of the text collection. The basis for the methods proposed in this paper is the model of a hierarchical Dirichlet process (HDP).

3. Topic Hierarchies of Hierarchical Dirichlet Processes (hHDP)

In this section we present the hHDP method in two variants. The first variant results in a hierarchy whose internal nodes are represented as probability distributions over topics and over words. Thus it performs a hierarchical vocabulary clustering (hvc). The second variant provides a hierarchical topic clustering (htc) of the corpus, where only leaf nodes are represented as distributions over words. We will refer to the first variant as hvHDP, and to the second as htHDP. We divide the section into two subsections, providing insights about the proposed method and information regarding the sampling scheme.

3.1 Stacking HDPs

Starting with the criteria that we posed at the end of Section 2, we want to be able to infer the number of topics at each level. For this purpose we use the mixture model of hierarchical Dirichlet processes (HDP) (Teh et al., 2006), which is illustrated in Figure 2.

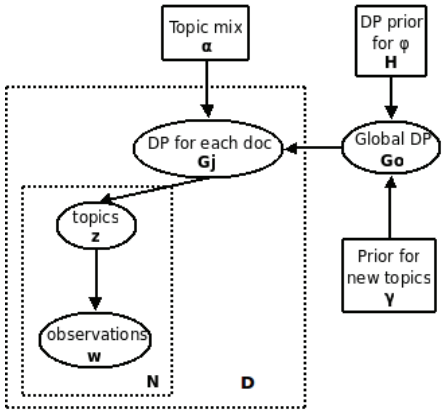


Figure 2: The HDP mixture model. Assuming a text collection of M documents, each of length N, there is a DP G_j for each document to draw word distributions. There is a global, higher-level DP (G_0) that maintains the global distribution of word distributions.

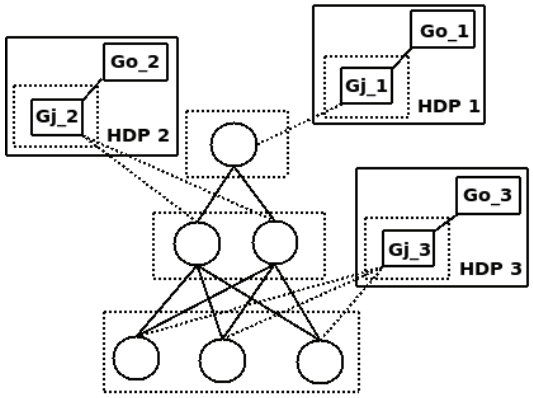


Figure 3: The association of the HDPs with the topic hierarchy. There is an HDP associated with each level. There are as many DPs (G_j) as the documents at each level, connected to all topics of the level. Each level also comprises a global DP (G_0) that is connected to all the G_j in this level.

In the proposed method (Figure 3), at each level of the hierarchy, there is a DP (G_j) for each document and a "global" DP (G_0) over all the DPs at that level. Therefore, each level of the topic hierarchy is associated with a HDP. An important characteristic of this approach is that the number of topics of each level is automatically inferred, due to the non-parametric Bayesian nature of

the HDP. In addition, it allows the topics at each level to be shared among the documents of the collection. Figure 3 depicts the DPs associated with different levels of the topic hierarchy.

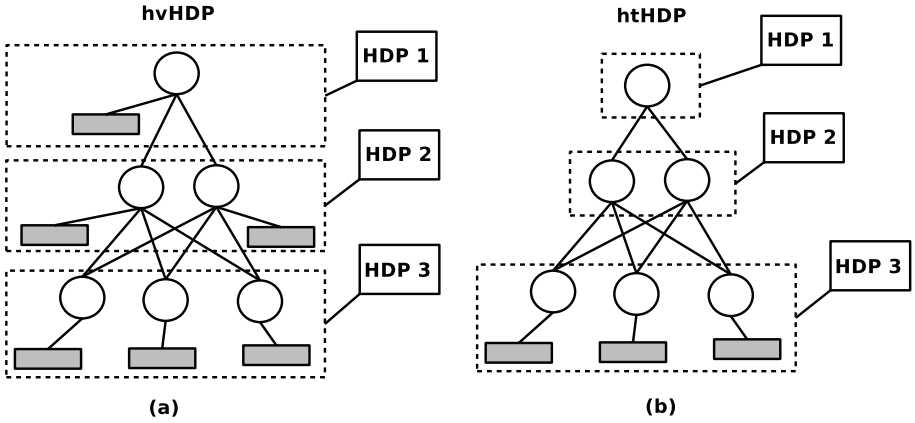


Figure 4: (a) hvHDP. (b) htHDP. Topics are represented as circles, while word distributions as gray boxes. hvHDP consists of topics that are both distributions over subtopics and over words. htHDP represents only leaf topics as distributions over words.

Therefore, at each level, a HDP is assumed, according to Figure 4, which is modeled as shown in Figure 3. The HDP at each level is used to express uncertainty about the possible number of mixture components, that is, the latent topics.

Among the models mentioned in Section 2, hPAM and hLDA are the closest "relatives" of hvHDP in terms of the representation of the corpus through an inferred hierarchy. They both have internal nodes containing words. However, in hLDA a topic is not allowed to have more than one parent, while in hPAM and hHDP this is allowed. On the other hand, while hPAM needs the number of internal topics to be fixed a priori, hLDA and hHDP are able to infer the number of topics at each level of the hierarchy, due to their non-parametric Bayesian nature. Moreover, while the model of hLDA requires that each document is made of topics across a specific path of the hierarchy, hPAM and hHDP provide much more flexibility, since topics can be shared among super-topics. Overall, hHDP combines the strengths of hPAM and hLDA, extending also the non-parametric approach to include the estimation of the depth of the learned hierarchy, which is further explained in the following paragraphs.

The PAM and the non-parametric PAM models are similar to the second version of hHDP (htHDP). The topics of the PAM models generate words at the leaf level and the models are based on a fixed three-level hierarchy. The simple PAM model needs the number of internal topics to be known a priori, while its non-parametric version uses the CRP to decide the number of super-topics and sub-topics. The obvious advantage of htHDP is its full non-parametric nature that does not impose restrictions on the depth and the branching factor at each level of the hierarchy.

3.2 Estimation of the Hierarchy

Regarding the estimation of the latent structure, exact inference of the hierarchy given a document collection is intractable. For this purpose we use Gibbs sampling, which climbs stochastically the

posterior distribution surface to find an area of high posterior probability and explores its curvature (Andrieu et al., 2003). Although the method of Gibbs sampling lacks theoretical guarantees, it has been proven to be appropriate for this type of data analysis and for inferring latent variables given the distribution of the observations and the corresponding priors. More information about sampling methods in machine learning can be found in Andrieu et al. (2003).

The sampling scheme of hHDP estimates both the number of topics at each level and the number of levels of the learned hierarchy. As shown in Figure 5, starting at the leaf level, we use HDP to infer the number of leaf topics as if no hierarchy is to be built. We then build the hierarchy bottom-up until reaching a level with a single node (the root topic). Each level is modeled as a HDP, estimating the appropriate number of topics.

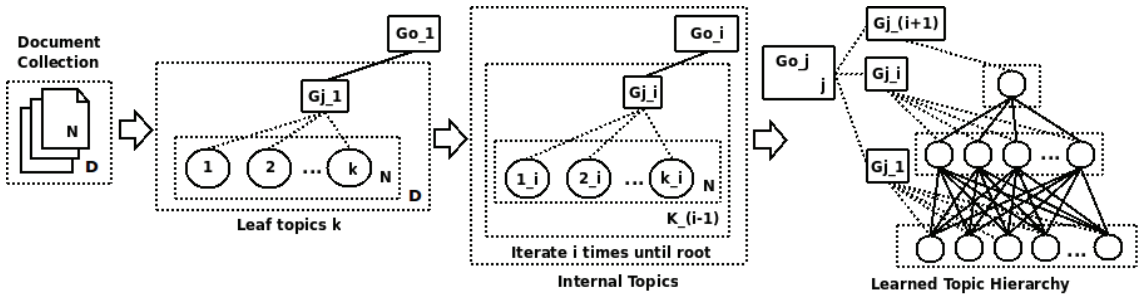


Figure 5: Bottom-up probabilistic estimation of the topic hierarchy: Starting with a corpus of M documents, the leaf topics are inferred first. The word distributions for each leaf topic make up the observations ("documents") for the estimation of the next level up. The procedure is repeated until the root topic is inferred.

Figure 5 presents the steps of the sampling scheme. We start with the text collection, which provides the observations, that is, the words, for the estimation. The words constitute the term space. At the first step that infers the leaf level, in a Chinese Restaurant Franchise analogy, we assume that the documents correspond to restaurants and the words to customers. The next steps differ for the two variants of hHDP.

In hvHDP, where topics are both distributions over subtopics and over words, the inference of the non-leaf levels treats topics, instead of documents, as restaurants. Thus, each inferred leaf topic maintains a distribution over the term space as its representation. Based on this distribution, it is treated as an observation for the inference of the next level up. Having inferred the topics at the leaf level, we know the mixture proportions that the documents of the collection follow. Similarly, each inferred topic maintains a distribution over the term space and a distribution over the subtopics below it, following the corresponding proportions inferred for this topic. Therefore each internal topic maintains a distribution over words and a distribution over subtopics. This procedure is repeated until we infer a single topic, which serves as the root of the hierarchy. In other words, at the leaf level we allocate documents to leaf topics, while at the intermediate levels we allocate topics to super-topics. The sampling scheme that we propose for hvHDP is described in Algorithm 1.

Therefore, the main contribution of this sampling scheme is the estimation of the non-leaf topics from "artificial" documents that correspond to estimated topics of lower levels. This procedure

Data: Term - Document matrix of frequencies
Result: Estimated topic hierarchy
set M=number of documents
set V=vocabulary size
estimate leaf topics K
set $T = K$
while $\mid T \mid > 1$ **do**
  // **transform document space**
  set $M = K$
  set input=MxV matrix of frequencies
  estimate topics K of next level up
  set $T = K$
end

Algorithm 1: Estimation of the topic hierarchy for the hierarchical vocabulary clustering hHDP method (hvHDP).

supports the non-parametric inference of the depth of the hierarchy. Together with the use of the HDP for the estimation of the number of topics at each level, it makes the estimation of the topic hierarchy completely non-parametric.

Regarding the second variant of the model (htHDP), where the internal topics are distributions only over subtopics and not words, the inference procedure differs in the modeling of non-leaf topics. Leaf topics serve now as customers, changing the term space, maintaining at the same time the restaurant space, which consists of the original documents. As observations for the inference of the next level up, we use the distributions of topics at the lower level over the original documents. Therefore, while in the first variant of hHDP, we had a topic - term matrix of frequencies as input for the estimation of an intermediate level of the hierarchy, in htHDP, we have a document - topic matrix of frequencies for the sampling procedure. The hierarchy estimated by htHDP is expected to be shallower than that inferred by hvHDP. This is because the term space is reduced when moving a level up. The procedure is repeated until we infer a single topic, which serves as the root topic. The proposed sampling scheme is described in Algorithm 2.

The last step in Figure 5 shows the overall model that is estimated. A topic hierarchy is derived from the corpus and a non-parametric Bayesian hierarchy is used at each level of the topic hierarchy. The first hHDP variant satisfies the criteria that we set in Section 2: internal topics are represented as distributions over words and over subtopics, topics can share subtopics at the lower level in the hierarchy, and topics across any level of the hierarchy are shared among documents. The degree of sharing topics across documents is expressed through the inferred parameters of the model, and this sharing of topics reflects the sharing of common terminology between documents. The non-parametric nature of this process is due to HDP that models each level of the hierarchy.

3.3 Level-wise Estimation

In hHDP, the estimation of each level is performed through posterior sampling of a HDP. At each level we integrate out all the probability measures G_i, the base measures G_0 and the tables. The metaphor of the "Chinese restaurant franchise" (CRF) is ofter used to illustrate the sampling scheme of the HDP. According to that metaphor, there are D restaurants and each one has an infinite number

Data: Term - Document matrix of frequencies
Result: Estimated coarse topic hierarchy
set M=number of documents
set V=vocabulary size
estimate leaf topics K
set $T = K$
while $\mid T \mid > 1$ **do**
  // **transform term space**
  set $V = K$
  set input=MxV matrix of frequencies
  estimate topics K of level up
  set $T = K$
end

Algorithm 2: Estimation of the topic hierarchy for hierarchical topic clustering version of hHDP (htHDP).

of tables. On each table the restaurant serves one of the infinitely many dishes that other restaurants may serve as well. A customer enters the restaurant. The customer not only chooses a table (which corresponds to topic sampling from Gj appearing in G_0), but also chooses whether she may have a dish popular among several restaurants (topic sharing among documents).

Based on the CRF metaphor, the collapsed sampling scheme includes only the sampling of the dishes, and the calculation of the number of tables that serve a specific dish in each restaurant. Thus, the sampling of an existing topic z at a specific level, given a word w_{ji} and the previous state of the Markov chain $z_{\neg ji}$ uses Equation (1), or equivalently Equation (2). On the other hand, the sampling of a new topic z_{new}, given a word w_{ji} and the previous state of the Markov chain $z_{\neg ji}$ uses Equation (3), or equivalently Equation (4).

$$p(z_{ji} = z \mid w_{ji}, z_{\neg ji}) \propto \frac{n_{j\cdot z} + \frac{\alpha t_z}{t_\cdot + \gamma}}{n_{j\cdot\cdot} + \alpha} \cdot \phi_z(w_{ji}) \tag{1}$$

$$p(z_{ji} = z \mid w_{ji}, z_{\neg ji}) \propto \frac{n_{j\cdot z} + \frac{\alpha t_z}{t_\cdot + \gamma}}{n_{j\cdot\cdot} + \alpha} \cdot \frac{n_{\cdot iz} + H}{n_{\cdot\cdot z} + VH} \tag{2}$$

$$p(z_{ji} = z_{new} \mid w_{ji}, z_{\neg ji}) \propto \frac{\alpha\gamma}{(n_{j\cdot\cdot} + \alpha)(t_\cdot + \gamma)} \cdot \phi_z(w_{ji}) \tag{3}$$

$$p(z_{ji} = z_{new} \mid w_{ji}, z_{\neg ji)} \propto \frac{\alpha\gamma}{(n_{j\cdot\cdot} + \alpha)(t_\cdot + \gamma)} \cdot \frac{1}{V} \tag{4}$$

In Equations (1) to (4), besides the hyper-parameters α and γ, $n_{j\cdot z}$ is the number of words in document j that are associated to topic z, $n_{j\cdot\cdot}$ is the number of words in document j, t_z is the number of tables that serve the dish z, and $t_\cdot$ is the total number of tables. The factor $n_{j\cdot z}$ emulates the draw of an existing dish of restaurant G_j, while the factor $\frac{\alpha t_z}{t_\cdot + \gamma}$ emulates the draw of a dish from the base restaurant G_0 that maintains all the dishes. The factor $\frac{\alpha\gamma}{t_\cdot + \gamma}$ emulates the draw of a new dish from

the global DP with hyper-parameter H. Finally, $\phi_z(w_{ji})$ stands for the word distribution $p(w \mid z)$. In addition, $n_{\cdot iz}$ is the number of occurrences of word i in topic z, $n_{\cdot\cdot z}$ is the total number of words assigned to topic z, and finally, H and V are the prior DP hyper-parameter for word distributions and the total number of words respectively.

Following the sampling of topic indicators, we calculate the number of tables that serve a specific dish at each restaurant, since we need that parameter for the sampling of topics. That is, we calculate the factor t_z, which influences the likelihood of a new table in document j via the factor $\frac{\alpha t_z}{t_{\cdot} + \gamma}$. We estimate this number by simulating a DP with hyper-parameter α, since we are interested in each document that is associated to a probability measure G_j, and parameter α provides control over the topic mixture. Algorithm 3 describes this process.

Data: $n_{j\cdot z}$, hyper-parameter α
Result: Number of tables in document j serving topic z
// **if no words exist then no tables are needed if** $n_{j\cdot z} = 0$ **then**
  return 0
end
// **if only one word exists, one table is needed**
if $n_{j\cdot z} = 1$ **then**
  return 1
end
// **if more words exist, simulate the DP**
set $t_z = 1$
for *all words w in* $[1, n_{j\cdot z}]$ **do**
  draw *rand* from Random
  set $DP_{table} = \alpha/(w+\alpha)$
  if $rand < DP_{table}$ **then**
    set $t_z = m_t + 1$
  end
end

Algorithm 3: Estimation of the number of tables that serve a specific dish (topic) in each restaurant. The parameters $t_z, n_{j\cdot z}$ are the ones used in Equations (1) to (4).

According to Algorithm 3, the estimation of the number of tables is performed for each restaurant, for the customers that have been assigned to new tables, not present in the previous sampling iteration. The factor t_z can only change when a word is assigned to a new topic. Due to the "rich gets richer" property of the DP, some tables become unoccupied. Then, the probability that this table will be occupied again in the future is zero, since this is proportional to $n_{j\cdot z}$, which will be zero. Therefore, when estimating a new level bottom-up, the number of tables tends to decrease. In addition, in hvHDP, at each level of the hierarchy we transform the inferred topics to documents. This introduces a bound on the number of tables, since we decrease the restaurant space, which in turn bounds the number of sharing components, that is, the topics. The same holds for htHDP, where the term space is dramatically reduced at each level, placing in this way a stronger bound on the number of sharing components. For this reason, the second variant of hHDP converges faster to a single topic, producing smaller hierarchies.

More formally, and according to Teh and Jordan (2010), $t_z \in O(\alpha log\frac{n_{j..}}{\alpha})$. Since G_0 is itself a draw from a DP, we have that $K \in O(\gamma log\sum_j \frac{t_z}{\gamma}) = O(\gamma log(\frac{\alpha}{\gamma}\sum_j log\frac{n_{j..}}{\alpha}))$. Assuming J groups, each of average size N, we have that $K \in O(\gamma log\frac{\alpha}{\gamma}Jlog\frac{N}{\alpha}) = O(\gamma log\frac{\alpha}{\gamma} + \gamma logJ + \gamma loglog\frac{N}{\alpha})$. Thus, the number of topics scales doubly logarithmically in the size of each group and logarithmically in the number of groups. In summary, the HDP expresses a prior belief that the number of topics grows very slowly in N.

4. Evaluation and Empirical Results

In this section, we present experiments using real data sets in order to demonstrate and evaluate the proposed method. We perform experiments on two different tasks, in order to obtain a good overview of the performance of the model. The goal is to measure how well the estimated hierarchy fits a heldout data set of a specific domain, given a training data set of that domain and to what extent the proposed method can be used for knowledge representation and help bootstrap an ontology learning method. In particular, we divide the section into two subsections. The first one (Section 4.1) concerns document modeling and provides qualitative and quantitative results, while Section 4.2 applies the model to the task of ontology learning.

4.1 Document Modeling

Given a document collection, the task is to retrieve the latent hierarchy of topics that represents and fits well, in terms of perplexity, to the data set. We fit hHDP and compare it with LDA and hLDA on various data sets using held-out documents.

In particular, we use 10-fold cross validation and report perplexity figures for each method. Perplexity is commonly used to evaluate language models, but it has also been used to evaluate probability models in general (Blei et al., 2003; Teh et al., 2006). Better models that avoid overfitting tend to assign high probabilities to the test events. Such models have lower perplexity as they are less surprised by the test sample. In other words, they can predict well held-out data that are drawn from a similar distribution as the training data. Hence, in our evaluation scenario, a lower perplexity score indicates better generalization performance. Equation (5) defines the perplexity on a test set D consisting of words $w_1, w_2, ..., w_N$.

$$Perplexity(D) = exp\{-\sum_{i=1}^{N}\frac{1}{N}\log p(w_i)\} \tag{5}$$

As an example of the results obtained by hvHDP, Figure 6 presents part of the latent structure that was discovered from the NIPS data set. The NIPS data set is a benchmark corpus that has been used in related work (Blei et al., 2004). It contains abstracts of the corresponding conferences from 1987 to 1999. Specifically, the data set comprises 1732 documents and no pre-processing took place before the learning of the hierarchy, resulting in an unrestricted vocabulary of 46873 terms. The model ran for 1000 iterations of the Gibbs sampler with fixed hyper-parameters. In particular, the Dirichlet process priors H and γ were set to 0.5 and 1.0 respectively, while the parameter α of the topic mixture was set to 10.0. The values selected for the hyper-parameters are similar to the values selected for related tasks in the literature (Mimno et al., 2007).

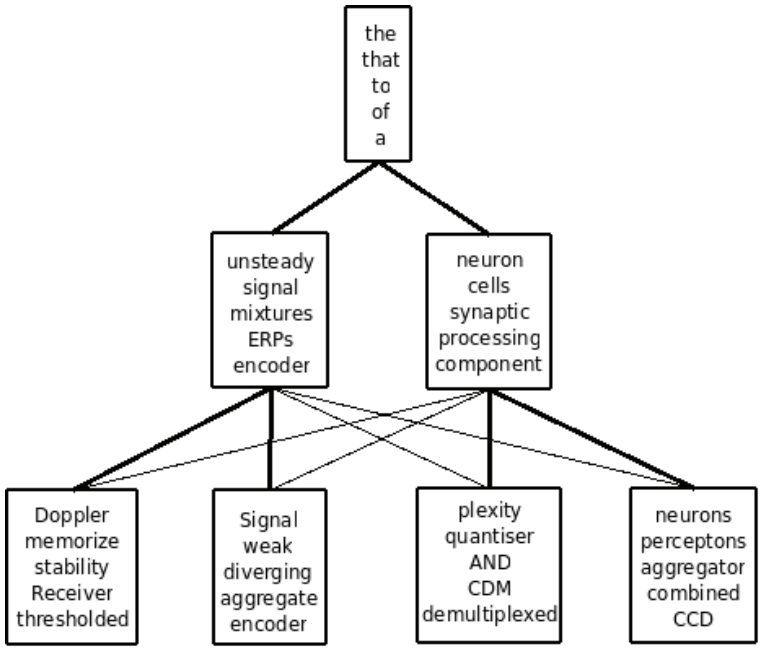


Figure 6: Part of the hierarchy estimated from the NIPS data set. The learned hierarchy contains 54 topics, inferred by the hHDP model without any user-specified parameters. Thick lines represent edges of high probability, while thinner ones stand for edges of lower probability.

As shown in Figure 6, the model discovered interesting topics from the field of the conference. Stop words are first grouped together at the root node representing a very general "topic" that connects equiprobably the two topics of the conference, signal processing and neural networks. Taking into account the context of the NIPS conferences, we believe that we have discovered a rather realistic hierarchical structure of 54 topics that fits well the field in question.

Similarly, Figure 7 illustrates part of the hierarchy that was produced by mixing two corpora together and running hvHDP on the resulting data set. In this experiment we wanted to investigate how the mixing of documents of different domains affects the resulting hierarchy, and in particular to see whether we can identify a sub-hierarchy of one domain inside the complete hierarchy that was learned. For this reason, we used 100 documents from the tourism domain and 1000 documents from the domain of molecular biology, resulting in a total of 1100 documents.

In Figure 7 only edges of high probability are shown for clarity reasons. The two separate sub-hierarchies, corresponding to the different domains are evident. The sub-hierarchy that corresponds to the tourism data set (inside the circle in the figure) is much smaller than that of the domain of molecular biology.

In order to obtain a quantitative evaluation of the method on document modeling, we used five different data sets. We also fitted the models of hLDA and LDA to the same data sets, as well as two other baseline models that we have implemented. The first, based on a uniform model (UM), is not trained and generates words following a uniform distribution, irrespective of the data set.

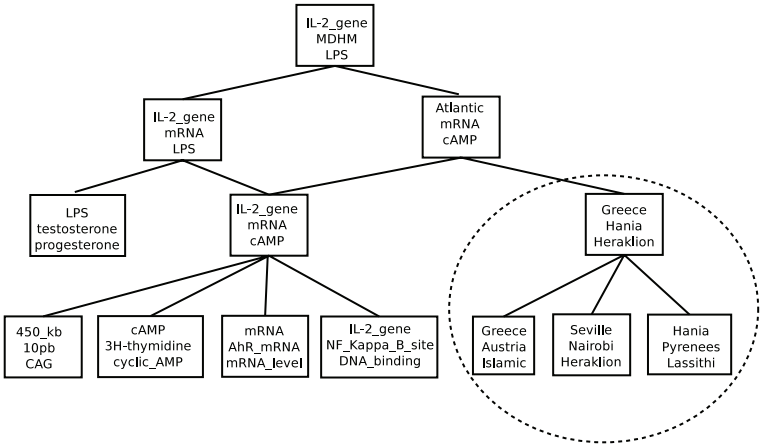


Figure 7: Part of the hierarchy estimated from a data set containing 1000 articles regarding molecular biology and 100 regarding tourism information.

The second that we call memory model (MM), memorizes the given data set and generates words according to the multinomial probability distribution of each document of the data set.

The different evaluation data sets that we used are the following: (a) the Genia data set,[1] from the domain of molecular biology, (b) the Seafood corpus,[2] comprising texts relative to seafood enterprises, (c) the Lonely Planet corpus,[3] consisting of texts from the tourism domain, (d) the Elegance corpus,[4] comprising nematode biology abstracts, and finally, (e) the NIPS data set[5] that includes abstracts from the corresponding conferences between the years 1987 and 1999. Table 2 summarizes basic statistics of the five data sets.

Data Set	#Docs	TermSpace	Domain
Genia	2000	16410	Molecular biology
Seafood	156	13031	Seafood enterprises
Lonely Planet	300	3485	Tourism
Elegance	7300	35890	Nematode biology
NIPS	1732	46873	NIPS conferences

Table 2: Data Sets

In the specific experimental setup we used the same hyper-parameters for all data sets. As mentioned above, for hHDP, $H = 0.5$, $\gamma = 1.0$ and $\alpha = 10.0$. In the case of hLDA, $\eta = 0.5$ and $\gamma = 1.0$, and we varied the number of levels, while in the case of LDA, we varied the number of topics from 10 to 120. Figure 8 illustrates the behavior of the models in the different data sets

1. The GENIA project can be found at `http://www-tsujii.is.s.u-tokyo.ac.jp/GENIA/home/wiki.cgi`.
2. The Seafood corpus can be found at `http://users.iit.demokritos.gr/~izavits/datasets/Seafood_corpus.zip`.
3. The Lonely Planet travel advise and information can be found at `http://www.lonelyplanet.com/`.
4. The Elegance corpus can be found at `http://elegans.swmed.edu/wli/cgcbib`.
5. The NIPS data set can be found at `http://books.nips.cc`.

for different numbers of LDA topics. Specifically, the figures plot the perplexity of the various models against the number of the discovered topics. In the case of hHDP the number of topics is inferred automatically and cannot vary with the LDA or the hLDA parameters. The hLDA model is parameterized by the number of levels. However, when changing the number of levels, the number of topics also changes. The model itself decides the branching factor at each level, and thus the total number of topics changes. A first observation in the results that we obtained is that in all cases, the simple UM results in very high perplexity values, between 2500 and 45000 that we do not depict in Figure 8 for reasons of readability of the graphs. Moreover, the MM performs worse in general than the rest of the models.

In order to interpret the different results obtained in the five different data sets, we measured the heterogeneity between the training and the held-out data in each case. More specifically, we measured the difference in the distribution of words between training and held-out data, using the mean total variational distance (TVD) (Gibbs and Su, 2002), according to Equation (6). The higher the TVD, the bigger the difference between the training and the held-out set. Table 3 presents the results of this measure in terms of the mean TVD in a 10-fold cross measurement. Based on these figures, the Genia data set seems to be the most homogeneous, while NIPS is the least.

$$TVD = \frac{1}{2} \sum_{i} \mid p(i) - q(i) \mid . \tag{6}$$

Data Set	Mean TVD
Genia	$1.2 * 10^{-5}$
Seafood	$3.5 * 10^{-5}$
Lonely Planet	$2.2 * 10^{-5}$
Elegance	$3.8 * 10^{-5}$
NIPS	$5.2 * 10^{-5}$

Table 3: Mean Total Variational Distance between the training and the held-out parts of the data sets.

Additionally, in order to validate the graphs of Figure 8, we measured the significance of the results, using the Wilcoxon signed-rank test. This test is suitable for this kind of experiment, since it is non-parametric and does not assume that the samples follow a specific distribution. In particular, we performed the test for the mean perplexity values, for each value of the number of topics. According to the test, the perplexity of a model is significantly lower than that of another model, if the output probability of the test is below 0.05, which is a threshold that is commonly used in statistical analysis.

In all data sets, the most interesting comparison is that between hHDP and hLDA. Thus, Table 4 depicts the ranges of topics for which the proposed model performs significantly better than the one of hLDA and the one of LDA. These ranges are also marked on the x axis of Figure 8 in all diagrams.

Examining the results on the Genia data set (Figure 8a), the lowest perplexity is achieved by hvHDP, while hLDA approaches the same perplexity for a number of topics around 60. LDA and htHDP obtain higher perplexity.

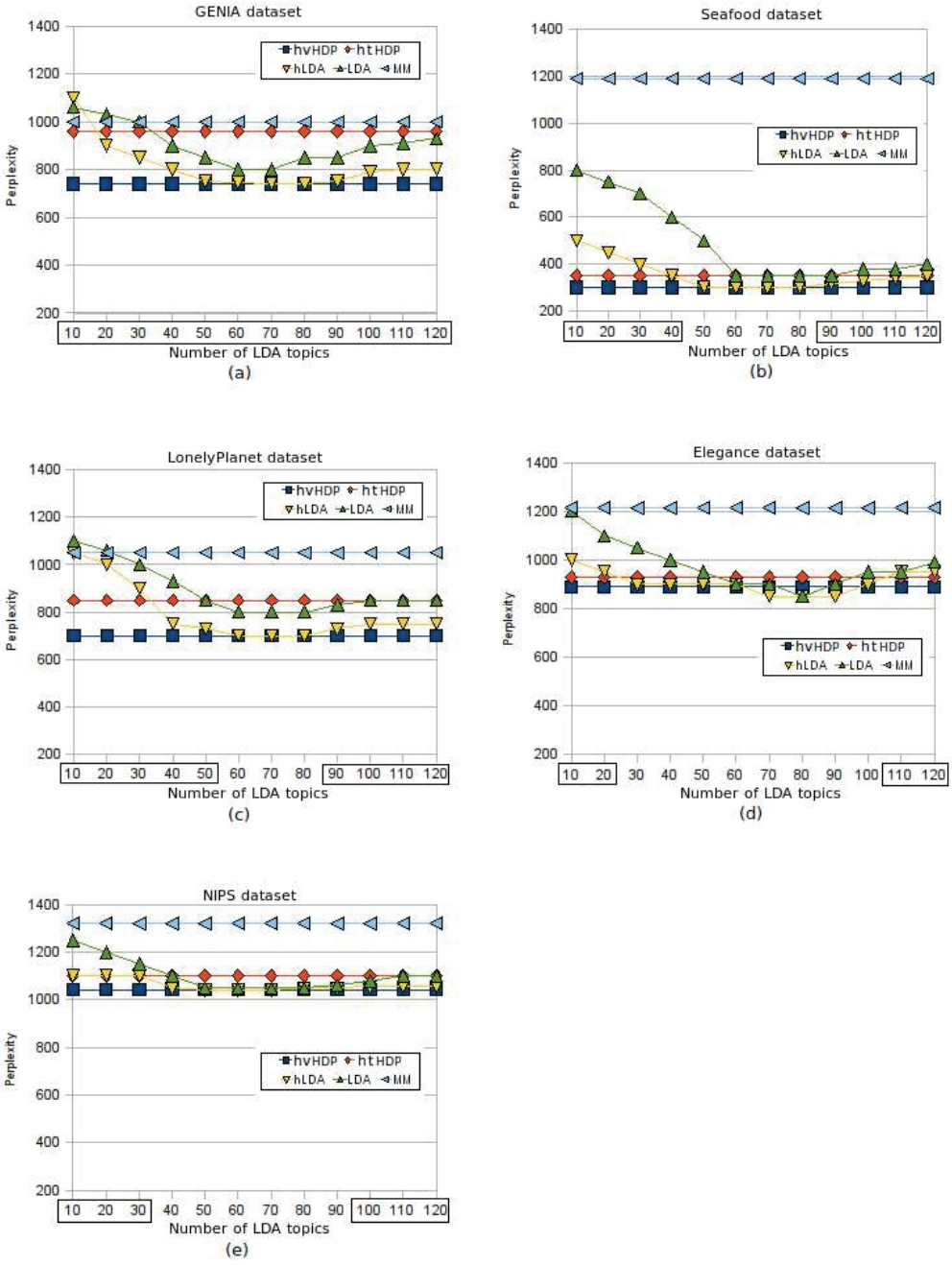


Figure 8: The behavior of the models on the five different data sets in terms of perplexity. The models: hvHDP, htHDP, hierarchical Latent Dirichlet Allocation (hLDA), Latent Dirichlet Allocation (LDA), and Memory Model (MM). Diagrams (a)-(e) illustrate the perplexity of the models for the Genia, Seafood, LonelyPlanet, Elegance and NIPS data sets respectively. Topic ranges where statistically significant improvement over existing models is achieved are marked on the x axis.

Comparison	Genia	Seafood	LP	Elegance	NIPS
hvHDP	$1-120$	$1-40$	$1-50$	$1-20$	$1-30$
hLDA		$90-120$	$90-120$	$110-120$	$100-120$
hvHDP	$1-120$	$1-120$	$1-120$	$1-50$	$1-120$
LDA				$100-120$	

Table 4: Significant differences between hvHDP and hLDA and between hvHDP and LDA in all corpora. Each cell presents topic ranges for which hvHDP performs significantly better than hLDA or LDA.

The comparison between hLDA and hvHDP showed that for all the cases in this data set, hvHDP obtains significantly lower perplexity values (Table 4).

Regarding the Seafood data set (Figure 8b), hLDA and LDA catch up with hvHDP after 40 and 60 topics respectively. htHDP also achieves good performance in this case. Regarding the statistical significance of the differences, Table 4 validates that hvHDP performs better than hLDA for a range of topics between $1-40$ and between $90-120$.

In the LonelyPlanet data set (Figure 8c), only hLDA manages to approach the good performance of hvHDP for a number of topics between $60-80$ (Table 4). The LDA and htHDP models perform worse. The htHDP is again much worse than the first variant of hHDP.

Concerning the Elegance data set (Figure 8d), all models, besides MM, achieve similar performance within a specific range of topics (50 to 120). Furthermore, this is the only data set where hLDA and LDA are observed to achieve better results than hvHDP, though not statistically significant and for a very small range of topics (around 80).

Finally, in the NIPS data set, (Figure 8e), hLDA and LDA manage to equal hvHDP for a certain range of topics and present a better performance than htHDP in a large range of topics. For this data set, the statistical test showed that hvHDP is better than hLDA in the range $10-30$ and $100-120$ topics and better than LDA in the whole range of topics, although for a certain range both models achieve similar perplexity values. On the other hand, the second version of hHDP outperforms only LDA between 10 and 20 topics.

The results illustrate clearly the suitability of hHDP for document modeling tasks. It discovers a hierarchy that fits well the given data sets, without overfitting them, thus achieving low values of perplexity. The competing models of hLDA and LDA manage only at their best to reach the performance of the proposed model. Furthermore, the performance of these models seems to be very sensitive to the chosen number of topics (number of levels in the case of hLDA). This observation makes the non-parametric modeling of hHDP particularly important. Comparing hLDA to the simple LDA, it is also quite clear that the hierarchical modeling of topics adds significant value to the model.

Regarding the naive UM and MM models, these are only used as baselines and they perform poorly. The experiments show that an overfitted model, such as MM, has low predictive performance outside the training set. On the other hand, a uniform model is not able to predict at all the test set, achieving the worst results.

A final important observation that is not evident in the numeric results, is that for a large number of topics, hLDA tends to construct a single path, rather than a hierarchy. Perhaps this can be attributed to the difficulty of identifying sufficiently different topics at various levels of abstraction,

when requesting a large depth for the hierarchy. By assigning all topics to a single branch, the model becomes equivalent to LDA. When this happens, the perplexity value of the two models is also very similar. The Wilcoxon statistical tests have indicated that in the Elegance data set and for a number of topics around 80, hLDA does not perform significantly better than LDA, while in the NIPS data set, the same situation holds for a number of topics between 50 and 100.

Perplexity has been criticized, since it is mainly used for the evaluation of language models. In addition, recent advances in topic modeling evaluation suggest the use of unbiased assessment of topic models. For this reason, we decided to conduct an additional experiment, measuring the log-likelihood of these models using the left-to-right sequential sampler (Buntine, 2009). This sampler improves on the algorithm proposed in Wallach et al. (2009), by providing unbiased estimates of the model likelihood for sufficiently large sample sizes. Since hvHDP, hLDA and LDA achieve the best results in terms of perplexity, we compare these models. Having the models trained on a portion (90%) of the data sets, we calculate the log-likelihood of the models on the remaining 10% that constitute the held-out data, using 10-fold cross validation. Figure 9 presents the results of this experiment, in terms of the mean log-likelihood.

The main result shown in Figure 9 is the same as in Figure 8. hHDP outperforms the other methods with statistical significance in most cases. The other methods, especially hLDA, approach the performance of hHDP if the right number of topics is chosen somehow. Therefore, the experiment has confirmed the value of estimating the number of topics and the depth of the hierarchy in a completely non-parametric way.

As an additional experiment on the task of document modeling, we assessed the ability of the method to estimate a known hierarchy, which is used to generate a set of documents. In particular, based on the hierarchy inferred for the Seafood data set, we generated a set of documents with the same average length as the original data set. Thus, we started at the root node of the hierarchy, and traversed it stochastically, based on the parameters of the model, which are the probabilities of each subtopic. When reaching a leaf topic we chose a word to be generated according to the probability distribution of that topic. In this manner, we generated a total of 156 documents, as many as the original data set, exhibiting similar word distributions. Then, we ran hvHDP on this "artificial" data set, estimating a latent hierarchy, which we compared manually against the one used to create the data set. From this comparison we concluded that all the topics of the estimated hierarchy have been correctly inferred. However, the estimated hierarchy comprises fewer topics, a fact that in terms of quantitative results implies a drop in recall.

4.2 Ontology Learning

The aim of this experiment was to validate the suitability of the proposed method on the task of ontology learning. The vocabulary clustering version of hHDP (hvHDP) estimates topics that are defined as distributions over words. It is, therefore, of particular interest to investigate how close these distributions are to a gold-standard hierarchy, given the corresponding data set. Such an experiment would highlight the potential of the method in other domains, such as automated ontology construction, and would provide qualitative and quantitative results regarding the performance of the method. In this experiment, we also compare hvHDP with hLDA.

Ontology learning (Gomez-Perez and Manzano-Macho, 2003; Maedche and Staab, 2003) refers to the set of methods and techniques used for either building an ontology from scratch, enriching, or adapting an existing ontology in an automated fashion, using various sources of information.

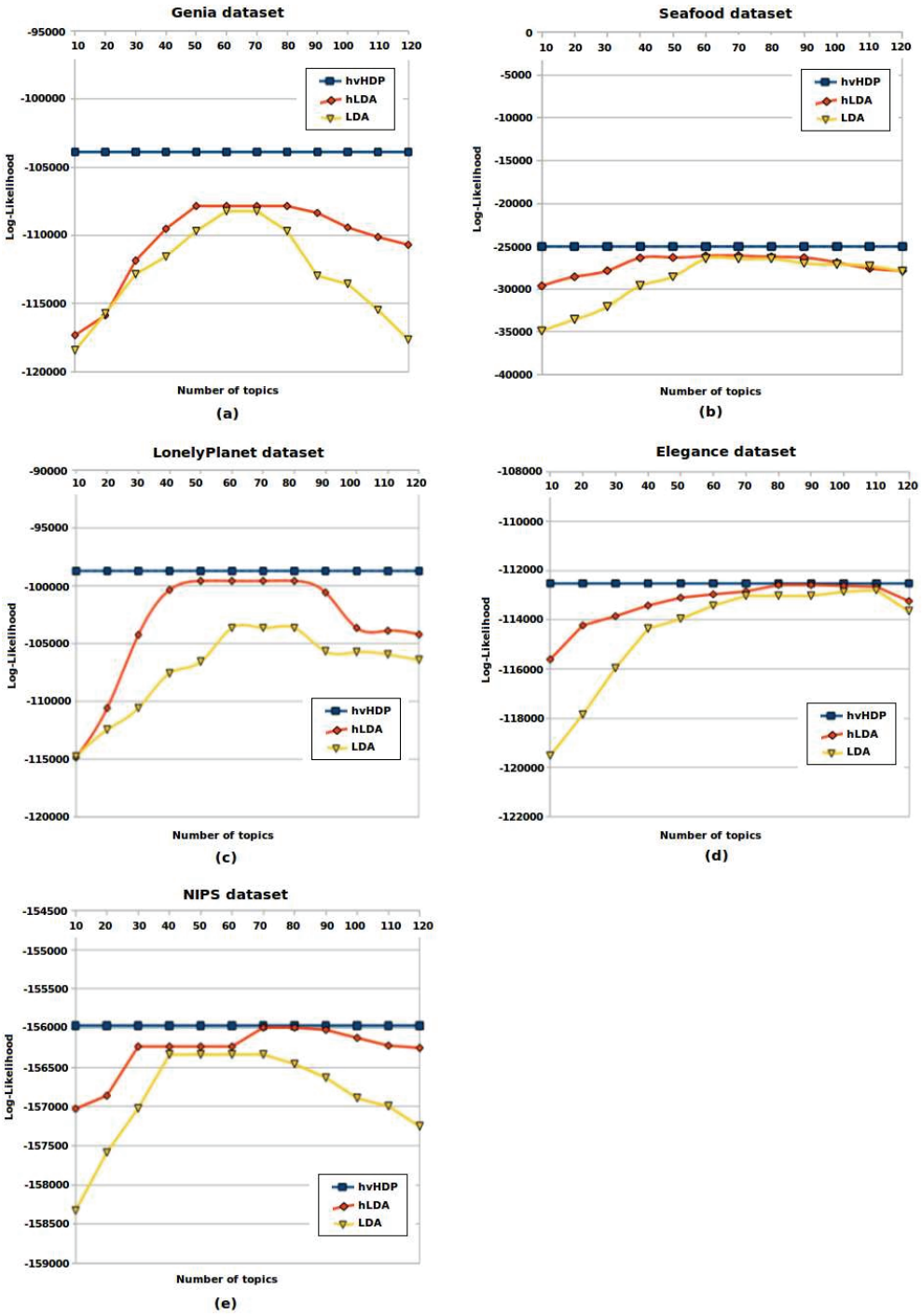


Figure 9: The behavior of the models (hvHDP, hierarchical Latent Dirichlet Allocation (hLDA), and Latent Dirichlet Allocation (LDA)) on the five different data sets in terms of log-likelihood. Diagrams (a)-(e) illustrate the perplexity of the models for the Genia, Seafood, LonelyPlanet, Elegance and NIPS data sets respectively.

This task is usually decomposed into three steps: (a) identification of topics, (b) building of the hierarchical backbone, and (c) enriching with further semantic relations. Regarding the sources of information, we focus here on text collections.

Both hvHDP and hLDA can be used to perform the first two steps of the ontology learning process, that is, identification of concepts and hierarchy construction, given the data set. Thus, we merge the aforementioned two steps into one, and we assume that the estimated latent topics correspond to ontology concepts. Therefore, in this task, we construct a topic ontology from scratch that comprises only hierarchical relations, given a collection of text documents and we compare it to a given gold standard ontology.

For this purpose, we use the Genia and the Lonely Planet data sets and the corresponding ontologies, which serve as gold standards for evaluation. The Genia ontology comprises 43 concepts that are connected by 41 subsumption relations, which is the only type of relation among the concepts. The Lonely Planet ontology contains 60 concepts and 60 subsumption relations among them. For our experiments, the only pre-processing applied to the corpus was to remove stop-words and words appearing fewer than 10 times.

The estimation of the hierarchy was achieved through 1000 iterations of the Gibbs sampler with fixed hyper-parameters $H = 0.5$ and $\gamma = 1.0$ for the Dirichlet priors and $\alpha = 10.0$ for the topic mixture. The evaluation was performed using the method proposed in Zavitsanos et al. (2010). This method is suitable for the evaluation of learned ontologies, since it represents the concepts of the gold ontology as multinomial probability distributions over the term space of the documents and provides measures in the closed interval of [0,1] to assess the quality of the learned structure.

In particular, the evaluation method first transforms the concepts of the gold ontology into probability distributions over the terms of the data set, taking into account the context of each ontology concept. In a second step, the gold ontology is matched to the learned hierarchy, based on how "close" the gold concepts and the learned topics are. The final evaluation is based on the measures of P and R that evaluate the learned hierarchy in the spirit of precision and recall respectively, as well as F that is a combined measure of P and R. The corresponding formulae are given in Equations (7), (8) and (9).

$$P = \frac{1}{M}\sum_{i=1}^{M}(1 - SD_i)PCP_i \tag{7}$$

$$R = \frac{1}{M}\sum_{i=1}^{M}(1 - SD_i)PCR_i \tag{8}$$

$$F = \frac{(\beta^2 + 1)P * R}{(\beta^2 R) + P} \tag{9}$$

In Equations (7) - (9), M is the number of matchings between learned topics and gold concepts and SD is a distance measure between concepts, ranging in $[0,1]$. Specifically, the total variational distance (TVD) (Gibbs and Su, 2002) of Equation (10) was used to assess the similarity between topics and gold concepts.

$$TVD = \frac{1}{2}\sum_{i} \mid P(i) - Q(i) \mid \tag{10}$$

In Equation (10), $P(\cdot)$ and $Q(\cdot)$ are multinomial probability distributions over words that represent a gold concept and a learned topic. The estimated topics are already represented as multinomial probability distributions over the term space of the data set, while the concepts of the gold ontology are also transformed into multinomial probability distributions over the same term space. Thus, the comparison between topics and gold concepts becomes straightforward.

The matching scheme compares the distributional representations of topics and gold concepts and finds the best matching in the sense that the most similar word distributions among the two hierarchies will be matched. More details about how the matching is performed can be found in Zavitsanos et al. (2010). The *PCP* and *PCR* (*probabilistic cotopy precision and recall*) factors in Equations (7) and (8) respectively, are influenced by the notion of semantic cotopy (Maedche and Staab, 2002). The cotopy set of a concept C is the set of all its direct and indirect super and subconcepts, including also the concept C itself. Thus, for a matching i, of a topic T in the learned ontology and a concept C in the gold ontology, PCP_i is defined as the number of topics in the cotopy set of T matched to concepts in the cotopy set of C, divided by the number of topics participating in the cotopy set of T. For the same matching i, PCR_i is defined as the number of topics in the cotopy set of T matched to concepts in the cotopy set of C, divided by the number of topics participating in the cotopy set of C.

Values of the P, R and F measures close to 1 indicate that the resulting hierarchy is close to the gold ontology, while values close to 0 indicate the opposite. Finally, we set $\beta = 1$ in Equation (9), hence using the harmonic mean of P and R.

Figure 10 depicts a part of the gold ontology on the left and a part of the estimated hierarchy on the right. The labels on the latent topics of the learned hierarchy correspond to the best TVD match of each topic with a gold concept. As it is shown in the figure, hHDP estimated a hierarchy very close to the gold standard. Thin edges between topics represent relations of low probability, while thicker edges carry higher probability.

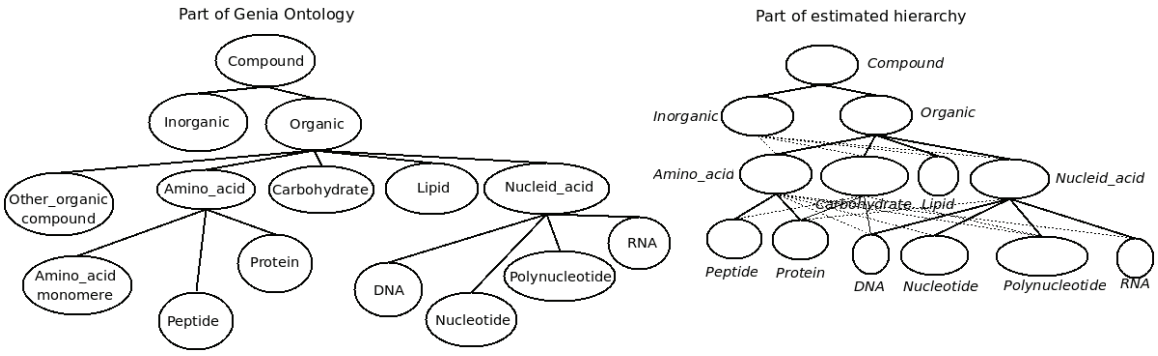


Figure 10: Part of the Genia ontology on the left and part of the estimated hierarchy on the right. The labels on the topics of the learned hierarchy correspond to the best match of each topic to a gold concept, according to TVD.

Regarding the estimated hierarchy, it comprises 38 topics in total, while the gold ontology comprises 43. Recall from Section 3 that the method estimates a probability distribution for each topic over all topics of the next level. Hence, we expect to learn a hierarchy comprising more relations than the gold ontology. However, relations with low probability, as the ones depicted with thin lines

in Figure 10, can be ignored. In addition, the way the hierarchy is estimated, through Gibbs sampling, infers the probability distributions, based on the assignments of words to topics and topics to subtopics. Through sampling, it is possible for fragments of documents not to be allocated to every estimated topic, and for subtopics not to be allocated to every super-topic. This leads to some zero values in the probability distributions of topics. Therefore, there exist cases where the probability of an edge in the resulting hierarchy may be zero. This fact provides extra flexibility to the method, since it permits the construction of unbalanced hierarchies and prunes edges that are definitely not necessary.

In the general case though, the learned hierarchy is expected to have more edges than the gold ontology has. Therefore, pruning mechanisms may be of particular importance for the task of ontology learning.

In the case of the Lonely Planet data set, hvHDP estimated a smaller hierarchy than the gold standard, achieving lower quantitative results in terms of P, R and F. The difficulty in estimating a hierarchy of similar size to the gold standard is due to the nature of the data set and the gold ontology. In particular, half of the gold concepts had only one instance and in general, most of the concepts were insufficiently instantiated in the data set.

Regarding hLDA, in the case of the Genia data set, the best quantitative results were obtained for an estimated hierarchy of depth equal to 6. In this case, hLDA performed similarly to hHDP in terms of P, R and F. However, in the case of the Lonely Planet data set the performance of the model was poor. In particular, the best quantitative results were obtained for an estimated hierarchy of 3 levels. However, these results are much lower than that of hvHDP for the same data set.

Table 5 presents the quantitative results of the experiments, in terms of P, R and F for both hHDP and LDA. For the proposed method, two cases are foreseen. The first case concerns the evaluation of the learned hierarchy as is, without any post-processing. The performance of hHDP is low, because the evaluation method is rather strict. The evaluation method does not take into account the probabilities on the edges connecting a topic to all its sub-topics, but rather assumes that all edges are of equal importance and penalizes the learned hierarchy for its high connectivity.

Therefore, through this first evaluation, we conclude that the original, highly connected hierarchy may not be usable as is. For this reason, we include another set of evaluation results in Table 5 that we call "pruned." This is actually the same method without the low probability relations between the topics. In particular, we keep relations with probability higher than 0.1. The pruned hierarchy is significantly closer to the gold standard than the unpruned one.

	Genia			LonelyPlanet		
Method	P	R	F	P	R	F
hHDP	0.65	0.60	0.624	0.22	0.15	0.17
hHDP-pruned	0.88	0.80	0.838	0.35	0.23	0.27
hLDA	0.62	0.55	0.58	0.07	0.01	0.017

Table 5: Quantitative results for the task of Ontology Learning.

In summary, we conclude that hvHDP can be applied to the task of ontology learning with promising results. Its ability to identify topics and at the same time build the taxonomic backbone can facilitate the learning of ontologies in a purely statistical way, providing a powerful tool that is independent of the language and the domain of the corpus. The proposed method discovered correctly the majority of the identifiable gold concepts in the experiment and constructed a hierarchy

that is very close to the gold standard. Furthermore, it constructed the taxonomy and inferred the correct depth without any user parameters (except the pruning threshold) in a statistical way and without any prior knowledge.

5. Conclusions

We have introduced hHDP, a flexible hierarchical probabilistic algorithm, suitable for learning hierarchies from discrete data. hHDP uses the "bag-of-words" representation of documents. The method is based on Dirichlet process priors that are able to express uncertainty about the number of topics at each level of the hierarchy. We have also presented a bottom-up non-parametric discovery method for the latent hierarchy, given a collection of documents. Since exact inference is known to be intractable in such non-parametric methods, approximate inference was performed, using the Gibbs sampling method, which provided accurate estimates.

An important contribution of this paper is the inference of the correct number of topics at each level of the hierarchy, as well as the depth of the hierarchy. Its Bayesian non-parametric nature requires no user parameters regarding the structure of the latent hierarchy. The Dirichlet process priors, as well as the bottom-up procedure for the estimation of the hierarchy, provide a flexible search in the space of different possible structures, choosing the one that maximizes the likelihood of the hierarchy for the given data set. Moreover, hHDP does not impose restrictions and constraints on the usage of topics, allowing multiple inheritance between topics of different layers and modeling the internal nodes as distributions of both subtopics and words.

We provided extensive experimental results for the proposed method in two different evaluation scenarios: (a) document modeling in five real data sets, comparing against state-of-the-art methods on the basis of perplexity, and (b) applying the method to an ontology learning task, comparing the learned hierarchy against a gold standard. The evaluation showed that hHDP is sufficiently robust and flexible. The proposed method discovered meaningful hierarchies and fitted well the given data sets. Finally, we have concluded that such methods are suitable for the task of ontology learning, since they are able to discover topics and arrange them hierarchically, in a way that is independent of the language and the domain of the data set, and without requiring any prior knowledge of the domain.

The very promising results that we obtained in this work, encouraged us to study and improve hHDP further. One possible improvement is the use of Pitman-Yor processes, which are generalizations of Dirichlet processes and produce power-law distributions. Natural language text is known to follow such distributions and therefore we may be able to model documents more accurately. In addition, we intend to apply the method to different tasks, including the learning of folksonomies from user-generated tags. Also, due to its statistical nature, it would be interesting to evaluate hHDP on different types of data sets, including images, time series and events. Finally, another future direction is to bootstrap hHDP from an existing ontology and infer the remaining parameters using the corresponding data set.

Acknowledgments

We would like to acknowledge support for this work from the research and development project ONTOSUM,[6] funded by the Greek General Secretariat for Research and Technology and the research and development project SYNC3.[7] We would also like to thank David Mimno and Andrew McCallum for helping us with the implementation of hLDA. Finally, we would like to thank Anna Sachinopoulou for providing us the Seafood corpus during her visit at our laboratory in NCSR "Demokritos."

References

C. Andrieu, N.D. Freitas, A. Doucet, and M.I. Jordan. An introduction to MCMC for machine learning. *Machine Learning Journal*, 50:5–43, 2003.

D.M. Blei and J.D. Lafferty. Correlated topic models. In *Advances in Neural Information Processing Systems 18*, 2006.

D.M. Blei, A.Y. Ng, and M.I. Jordan. Latent Dirichlet allocation. *Journal of Machine Learning Research*, 3:993–1022, 2003.

D.M. Blei, T.L. Griffiths, M.I. Jordan, and J.B. Tenenbaum. Hierarchical topic models and the nested Chinese restaurant process. In *Advances in Neural Information Processing Systems*, 2004.

W. Buntine. Estimating likelihoods for topic models. In *Asian Conference in Machine Learning*, 2009.

J. Chang, J. Boyd-Graber, S. Gerrish, C. Wang, and D.M. Blei. Reading tea leaves: How humans interpret topic models. In *Neural Information Processing Systems (NIPS)*, 2009.

B. De Finetti. Funzione caratteristica di un fenomeno aleatorio. In *Atti della R. Academia Nazionale dei Lincei, Serie 6. Memorie, Classe di Scienze Fisiche, Mathematice e Naturale*, pages 251–299. 1931.

K. Frantzi, S. Ananiadou, and H. Mima. Automatic recognition of multi-word terms: The C-value/NC-value method. *International Journal on Digital Libraries*, 3(2):115–130, 2000.

E. Gaussier, C. Goutte, K. Popat, and F. Chen. A hierarchical model for clustering and categorizing documents. In *Advances in Information Retrieval - Proceedings of the 24th BCS-IRSG European Colloquium on IR Research*, pages 229–247, 2002.

A.L. Gibbs and F.E. Su. On choosing and bounding probability metrics. *International Statistical Review*, 70(3):419–435, 2002.

A. Gomez-Perez and D. Manzano-Macho. A survey of ontology learning methods and techniques. Technical Report Deliverable 1.5, 2003. Ontology-based Information Exchange for Knowledge Management and Electronic Commerce.

6. For ONTOSUM see also `http://www.ontosum.org/`.
7. For SYNC3 see also `http://www.sync3.eu/`.

T. Griffiths and M. Steyvers. A probabilistic approach to semantic representation. In *Proceedings of the 24th Annual Conference of the Cognitive Science Society*, pages 381–386, 2002.

T. Hofmann. Unsupervised learning by probabilistic latent semantic analysis. *Machine Learning Journal*, 41:177–196, 2001.

W. Li and A. McCallum. Pachinko allocation: DAG-structured mixture models of topic correlations. In *Proceedings of the 23rd International Conference on Machine Learning*, pages 577–584, 2006.

W. Li, D. Blei, and A. McCallum. Nonparametric Bayes pachinko allocation. In *Uncertainty in Artificial Intelligence*, 2007.

A. Maedche and S. Staab. Measuring similarity between ontologies. In *Proceedings of the European Conference on Knowledge Acquisition and Management (EKAW)*, pages 251–263, 2002.

A. Maedche and S. Staab. Ontology learning. In *Handbook on Ontologies in Information Systems*. Springer, 2003.

D. Mimno, W. Li, and A. McCallum. Mixtures of hierarchical topics with pachinko allocation. In *Proceedings of the 24th International Conference on Machine Learning*, pages 633–640, 2007.

G. Salton and M.H. McGill. *Introduction to Modern Information Retrieval*. McGraw-Hill, 1986.

M. Steyvers and T. Griffiths. Probabilistic topic models. In *Handbook of Latent Semantic Analysis*. Hillsdale, NJ: Erlbaum, 2007.

Y.W. Teh and M.I. Jordan. Hierarchical bayesian nonparametric models with applications. In *Bayesian Nonparametrics: Principles and Practice*. *Cambridge, UK: Cambridge University Press*. 2010.

Y.W. Teh, M.I. Jordan, M.J. Beal, and D.M. Blei. Hierarchical Dirichlet processes. *Journal of the American Statistical Association*, 2006.

H.M. Wallach, I. Murray, R. Salakhutdinov, and D. Mimno. Evaluation methods for topic models. In *International Conference on Machine Learning*, 2009.

E. Zavitsanos, S. Petridis, G. Paliouras, and G.A. Vouros. Determining automatically the size of learned ontologies. In *18th European Conference on Artificial Intelligence, ECAI*, 2008.

E. Zavitsanos, G. Paliouras, and G.A. Vouros. Gold standard evaluation of ontology learning methods through ontology transformation and alignment. *IEEE Transactions on Knowledge and Data Engineering*. *IEEE Computer Society Digital Library*. *IEEE Computer Society, http://doi.ieeecomputersociety.org/10.1109/TKDE.2010.195*, 2010.

Journal of Machine Learning Research 12 (2011) 2777-2824 Submitted 9/09; Revised 3/10; Published 10/11

Structured Variable Selection with Sparsity-Inducing Norms

Rodolphe Jenatton RODOLPHE.JENATTON@INRIA.FR
INRIA - SIERRA Project-team,
Laboratoire d'Informatique de l'Ecole Normale Supérieure (INRIA/ENS/CNRS UMR 8548)
23, avenue d'Italie
75214 Paris, France

Jean-Yves Audibert AUDIBERT@CERTIS.ENPC.FR
Imagine (ENPC/CSTB), Université Paris-Est,
Laboratoire d'Informatique de l'Ecole Normale Supérieure (INRIA/ENS/CNRS UMR 8548)
6 avenue Blaise Pascal
77455 Marne-la-Vallée, France

Francis Bach FRANCIS.BACH@INRIA.FR
INRIA - SIERRA Project-team,
Laboratoire d'Informatique de l'Ecole Normale Supérieure (INRIA/ENS/CNRS UMR 8548),
23, avenue d'Italie
75214 Paris, France

Editor: Bin Yu

Abstract

We consider the empirical risk minimization problem for linear supervised learning, with regularization by structured sparsity-inducing norms. These are defined as sums of Euclidean norms on certain subsets of variables, extending the usual ℓ_1-norm and the group ℓ_1-norm by allowing the subsets to overlap. This leads to a specific set of allowed nonzero patterns for the solutions of such problems. We first explore the relationship between the groups defining the norm and the resulting nonzero patterns, providing both forward and backward algorithms to go back and forth from groups to patterns. This allows the design of norms adapted to specific prior knowledge expressed in terms of nonzero patterns. We also present an efficient active set algorithm, and analyze the consistency of variable selection for least-squares linear regression in low and high-dimensional settings.

Keywords: sparsity, consistency, variable selection, convex optimization, active set algorithm

1. Introduction

Sparse linear models have emerged as a powerful framework to deal with various supervised estimation tasks, in machine learning as well as in statistics and signal processing. These models basically seek to predict an output by linearly combining only a small subset of the features describing the data. To simultaneously address this variable selection and the linear model estimation, ℓ_1-norm regularization has become a popular tool, that benefits both from efficient algorithms (see, e.g., Efron et al., 2004; Lee et al., 2007; Beck and Teboulle, 2009; Yuan et al., 2010; Bach et al., 2011, and multiple references therein) and well-developed theory for generalization properties and variable selection consistency (Zhao and Yu, 2006; Wainwright, 2009; Bickel et al., 2009; Zhang, 2009; Negahban et al., 2009).

When regularizing by the ℓ_1-norm, sparsity is yielded by treating each variable individually, regardless of its position in the input feature vector, so that existing relationships and structures between the variables (e.g., spatial, hierarchical or related to the physics of the problem at hand) are merely disregarded. However, many practical situations could benefit from this type of prior knowledge, potentially both for interpretability purposes and for improved predictive performance.

For instance, in neuroimaging, one is interested in localizing areas in functional magnetic resonance imaging (fMRI) or magnetoencephalography (MEG) signals that are discriminative to distinguish between different brain states (Gramfort and Kowalski, 2009; Xiang et al., 2009; Jenatton et al., 2011a, and references therein). More precisely, fMRI responses consist in voxels whose three-dimensional spatial arrangement respects the anatomy of the brain. The discriminative voxels are thus expected to have a specific localized spatial organization (Xiang et al., 2009), which is important for the subsequent identification task performed by neuroscientists. In this case, regularizing by a plain ℓ_1-norm to deal with the ill-conditionedness of the problem (typically only a few fMRI responses described by tens of thousands of voxels) would ignore this spatial configuration, with a potential loss in interpretability and performance.

Similarly, in face recognition, robustness to occlusions can be increased by considering as features, sets of pixels that form small convex regions on the face images (Jenatton et al., 2010b). Again, a plain ℓ_1-norm regularization fails to encode this specific spatial locality constraint (Jenatton et al., 2010b). The same rationale supports the use of *structured sparsity* for background subtraction tasks (Cevher et al., 2008; Huang et al., 2009; Mairal et al., 2010b). Still in computer vision, object and scene recognition generally seek to extract bounding boxes in either images (Harzallah et al., 2009) or videos (Dalal et al., 2006). These boxes concentrate the predictive power associated with the considered object/scene class, and have to be found by respecting the spatial arrangement of the pixels over the images. In videos, where series of frames are studied over time, the temporal coherence also has to be taken into account. An unstructured sparsity-inducing penalty that would disregard this spatial and temporal information is therefore not adapted to select such boxes.

Another example of the need for higher-order prior knowledge comes from bioinformatics. Indeed, for the diagnosis of tumors, the profiles of array-based comparative genomic hybridization (arrayCGH) can be used as inputs to feed a classifier (Rapaport et al., 2008). These profiles are characterized by plenty of variables, but only a few samples of such profiles are available, prompting the need for variable selection. Because of the specific spatial organization of bacterial artificial chromosomes along the genome, the set of discriminative features is expected to have specific contiguous patterns. Using this prior knowledge on top of a standard sparsity-inducing method leads to improvement in classification accuracy (Rapaport et al., 2008). In the context of multi-task regression, a genetic problem of interest is to find a mapping between a small subset of single nucleotide polymorphisms (SNP's) that have a phenotypic impact on a given family of genes (Kim and Xing, 2010). This target family of genes has its own structure, where some genes share common genetic characteristics, so that these genes can be embedded into a underlying hierarchy (Kim and Xing, 2010). Exploiting directly this hierarchical information in the regularization term outperforms the unstructured approach with a standard ℓ_1-norm. Such hierarchical structures have been likewise useful in the context of wavelet regression (Baraniuk et al., 2010; Zhao et al., 2009; Huang et al., 2009; Jenatton et al., 2011b), kernel-based non linear variable selection (Bach, 2008a), for topic modelling (Jenatton et al., 2011b) and for template selection in natural language processing (Martins et al., 2011).

These real world examples motivate the need for the design of sparsity-inducing regularization schemes, capable of encoding more sophisticated prior knowledge about the expected sparsity patterns.

As mentioned above, the ℓ_1-norm focuses only on *cardinality* and cannot easily specify side information about the patterns of nonzero coefficients ("nonzero patterns") induced in the solution, since they are all theoretically possible. Group ℓ_1-norms (Yuan and Lin, 2006; Roth and Fischer, 2008; Huang and Zhang, 2010) consider a partition of all variables into a certain number of subsets and penalize the sum of the Euclidean norms of each one, leading to selection of groups rather than individual variables. Moreover, recent works have considered overlapping but nested groups in constrained situations such as trees and directed acyclic graphs (Zhao et al., 2009; Bach, 2008a; Kim and Xing, 2010; Jenatton et al., 2010a, 2011b; Schmidt and Murphy, 2010).

In this paper, we consider all possible sets of groups and characterize exactly what type of prior knowledge can be encoded by considering sums of norms of overlapping groups of variables. Before describing how to go from groups to nonzero patterns (or equivalently zero patterns), we show that it is possible to "reverse-engineer" a given set of nonzero patterns, that is, to build the unique minimal set of groups that will generate these patterns. This allows the automatic design of sparsity-inducing norms, adapted to target sparsity patterns. We give in Section 3 some interesting examples of such designs in specific geometric and structured configurations, which covers the type of prior knowledge available in the real world applications described previously.

As will be shown in Section 3, for each set of groups, a notion of hull of a nonzero pattern may be naturally defined. For example, in the particular case of the two-dimensional planar grid considered in this paper, this hull is exactly the axis-aligned bounding box or the regular convex hull. We show that, in our framework, the allowed nonzero patterns are exactly those equal to their hull, and that the hull of the relevant variables is consistently estimated under certain conditions, both in low and high-dimensional settings. Moreover, we present in Section 4 an efficient active set algorithm that scales well to high dimensions. Finally, we illustrate in Section 6 the behavior of our norms with synthetic examples on specific geometric settings, such as lines and two-dimensional grids.

1.1 Notation

For $x \in \mathbb{R}^p$ and $q \in [1,\infty)$, we denote by $\|x\|_q$ its ℓ_q-norm defined as $(\sum_{j=1}^p |x_j|^q)^{1/q}$ and $\|x\|_\infty = \max_{j\in\{1,\ldots,p\}} |x_j|$. Given $w \in \mathbb{R}^p$ and a subset J of $\{1,\ldots,p\}$ with cardinality $|J|$, w_J denotes the vector in $\mathbb{R}^{|J|}$ of elements of w indexed by J. Similarly, for a matrix $M \in \mathbb{R}^{p\times m}$, $M_{IJ} \in \mathbb{R}^{|I|\times|J|}$ denotes the sub-matrix of M reduced to the rows indexed by I and the columns indexed by J. For any finite set A with cardinality $|A|$, we also define the $|A|$-tuple $(y^a)_{a\in A} \in \mathbb{R}^{p\times|A|}$ as the collection of p-dimensional vectors y^a indexed by the elements of A. Furthermore, for two vectors x and y in $\mathbb{R}^p$, we denote by $x\circ y = (x_1y_1,\ldots,x_py_p)^\top \in \mathbb{R}^p$ the elementwise product of x and y.

2. Regularized Risk Minimization

We consider the problem of predicting a random variable $Y \in \mathcal{Y}$ from a (potentially non random) vector $X \in \mathbb{R}^p$, where $\mathcal{Y}$ is the set of responses, typically a subset of $\mathbb{R}$. We assume that we are given n observations $(x_i,y_i) \in \mathbb{R}^p \times \mathcal{Y}$, $i = 1,\ldots,n$. We define the *empirical risk* of a loading vector $w \in \mathbb{R}^p$ as $L(w) = \frac{1}{n}\sum_{i=1}^n \ell\big(y_i, w^\top x_i\big)$, where $\ell : \mathcal{Y}\times\mathbb{R} \mapsto \mathbb{R}^+$ is a *loss function*. We assume that ℓ

is *convex and continuously differentiable* with respect to the second parameter. Typical examples of loss functions are the square loss for least squares regression, that is, $\ell(y,\hat{y}) = \frac{1}{2}(y-\hat{y})^2$ with $y \in \mathbb{R}$, and the logistic loss $\ell(y,\hat{y}) = \log(1+e^{-y\hat{y}})$ for logistic regression, with $y \in \{-1,1\}$.

We focus on a general family of sparsity-inducing norms that allow the penalization of subsets of variables grouped together. Let us denote by $\mathcal{G}$ a subset of the power set of $\{1,\dots,p\}$ such that $\bigcup_{G\in\mathcal{G}} G = \{1,\dots,p\}$, that is, a spanning set of subsets of $\{1,\dots,p\}$. Note that $\mathcal{G}$ does not necessarily define a partition of $\{1,\dots,p\}$, and therefore, *it is possible for elements of $\mathcal{G}$ to overlap*. We consider the norm Ω defined by

$$\Omega(w) = \sum_{G\in\mathcal{G}} \Big(\sum_{j\in G} (d_j^G)^2 |w_j|^2 \Big)^{\frac{1}{2}} = \sum_{G\in\mathcal{G}} \|d^G \circ w\|_2 , \tag{1}$$

where $(d^G)_{G\in\mathcal{G}}$ is a $|\mathcal{G}|$-tuple of p-dimensional vectors such that $d_j^G > 0$ if $j \in G$ and $d_j^G = 0$ otherwise. A same variable w_j belonging to two different groups $G_1, G_2 \in \mathcal{G}$ is allowed to be weighted differently in G_1 and G_2 (by respectively $d_j^{G_1}$ and $d_j^{G_2}$). We do not study the more general setting where each d^G would be a (non-diagonal) positive-definite matrix, which we defer to future work. Note that a larger family of penalties with similar properties may be obtained by replacing the ℓ_2-norm in Equation (1) by other ℓ_q-norm, $q > 1$ (Zhao et al., 2009). Moreover, non-convex alternatives to Equation (1) with quasi-norms in place of norms may also be interesting, in order to yield sparsity more aggressively (see, e.g., Jenatton et al., 2010b).

This general formulation has several important sub-cases that we present below, the goal of this paper being to go beyond these, and to consider norms capable to incorporate richer prior knowledge.

- **ℓ_2-norm**: $\mathcal{G}$ is composed of one element, the full set $\{1,\dots,p\}$.
- **ℓ_1-norm**: $\mathcal{G}$ is the set of all singletons, leading to the Lasso (Tibshirani, 1996) for the square loss.
- **ℓ_2-norm and ℓ_1-norm**: $\mathcal{G}$ is the set of all singletons and the full set $\{1,\dots,p\}$, leading (up to the squaring of the ℓ_2-norm) to the Elastic net (Zou and Hastie, 2005) for the square loss.
- **Group ℓ_1-norm**: $\mathcal{G}$ is any partition of $\{1,\dots,p\}$, leading to the group-Lasso for the square loss (Yuan and Lin, 2006).
- **Hierarchical norms**: when the set $\{1,\dots,p\}$ is embedded into a tree (Zhao et al., 2009) or more generally into a directed acyclic graph (Bach, 2008a), then a set of p groups, each of them composed of descendants of a given variable, is considered.

We study the following regularized problem:

$$\min_{w\in\mathbb{R}^p} \frac{1}{n}\sum_{i=1}^{n} \ell\big(y_i, w^\top x_i\big) + \mu\Omega(w), \tag{2}$$

where $\mu \geq 0$ is a regularization parameter. Note that a non-regularized constant term could be included in this formulation, but it is left out for simplicity. We denote by $\hat{w}$ any solution of Problem (2). Regularizing by linear combinations of (non-squared) ℓ_2-norms is known to induce sparsity in $\hat{w}$ (Zhao et al., 2009); our grouping leads to specific patterns that we describe in the next section.

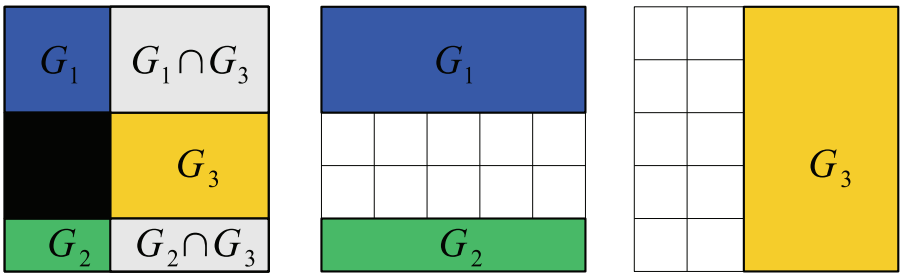


Figure 1: Groups and induced nonzero pattern: three sparsity-inducing groups (middle and right, denoted by $\{G_1, G_2, G_3\}$) with the associated nonzero pattern which is the complement of the union of groups, that is, $(G_1 \cup G_2 \cup G_3)^c$ (left, in black).

3. Groups and Sparsity Patterns

We now study the relationship between the norm Ω defined in Equation (1) and the nonzero patterns the estimated vector $\hat{w}$ is allowed to have. We first characterize the set of nonzero patterns, then we provide forward and backward procedures to go back and forth from groups to patterns.

3.1 Stable Patterns Generated by $\mathcal{G}$

The regularization term $\Omega(w) = \sum_{G \in \mathcal{G}} \|d^G \circ w\|_2$ is a mixed (ℓ_1, ℓ_2)-norm (Zhao et al., 2009). At the group level, it behaves like an ℓ_1-norm and therefore, Ω induces group sparsity. In other words, each $d^G \circ w$, and equivalently each w_G (since the support of d^G is exactly G), is encouraged to go to zero. On the other hand, within the groups $G \in \mathcal{G}$, the ℓ_2-norm does not promote sparsity. Intuitively, for a certain subset of groups $\mathcal{G}' \subseteq \mathcal{G}$, the vectors w_G associated with the groups $G \in \mathcal{G}'$ will be exactly equal to zero, leading to a set of zeros which is the union of these groups, $\bigcup_{G \in \mathcal{G}'} G$. Thus, the set of allowed zero patterns should be the *union-closure* of $\mathcal{G}$, that is, (see Figure 1 for an example):

$$\mathcal{Z} = \Big\{ \bigcup_{G \in \mathcal{G}'} G;\ \mathcal{G}' \subseteq \mathcal{G} \Big\}.$$

The situation is however slightly more subtle as some zeros can be created by chance (just as regularizing by the ℓ_2-norm may lead, though it is unlikely, to some zeros). Nevertheless, Theorem 2 shows that, under mild conditions, the previous intuition about the set of zero patterns is correct. Note that instead of considering the set of zero patterns $\mathcal{Z}$, it is also convenient to manipulate nonzero patterns, and we define

$$\mathcal{P} = \Big\{ \bigcap_{G \in \mathcal{G}'} G^c;\ \mathcal{G}' \subseteq \mathcal{G} \Big\} = \big\{ Z^c;\ Z \in \mathcal{Z} \big\}.$$

We can equivalently use $\mathcal{P}$ or $\mathcal{Z}$ by taking the complement of each element of these sets.

The following two results characterize the solutions of Problem (2). We first gives sufficient conditions under which this problem has a unique solution. We then formally prove the aforementioned intuition about the zero patterns of the solutions of (2), namely they should belong to $\mathcal{Z}$. In the following two results (see proofs in Appendix A and Appendix B), we assume that $\ell : (y, y') \mapsto \ell(y, y')$ is nonnegative, twice continuously differentiable with positive second derivative with respect to the

second variable and non-vanishing mixed derivative, that is, for any y, y' in $\mathbb{R}$, $\frac{\partial^2 \ell}{\partial y'^2}(y,y') > 0$ and $\frac{\partial^2 \ell}{\partial y \partial y'}(y,y') \neq 0$.

Proposition 1 *Let Q denote the Gram matrix $\frac{1}{n}\sum_{i=1}^n x_i x_i^\top$. We consider the optimization problem in Equation (2) with $\mu > 0$. If Q is invertible or if $\{1,\dots,p\}$ belongs to $\mathcal{G}$, then the problem in Equation (2) admits a unique solution.*

Note that the invertibility of the matrix Q requires $p \leq n$. For high-dimensional settings, the uniqueness of the solution will hold when $\{1,\dots,p\}$ belongs to $\mathcal{G}$, or as further discussed at the end of the proof, as soon as for any $j,k \in \{1,\dots,p\}$, there exists a group $G \in \mathcal{G}$ which contains both j and k. Adding the group $\{1,\dots,p\}$ to $\mathcal{G}$ will in general not modify $\mathcal{P}$ (and $\mathcal{Z}$), but it will cause $\mathcal{G}$ to lose its minimality (in a sense introduced in the next subsection). Furthermore, adding the full group $\{1,\dots,p\}$ has to be put in parallel with the equivalent (up to the squaring) ℓ_2-norm term in the elastic-net penalty (Zou and Hastie, 2005), whose effect is to notably ensure strong convexity. For more sophisticated uniqueness conditions that we have not explored here, we refer the readers to Osborne et al. (2000, Theorem 1, 4 and 5), Rosset et al. (2004, Theorem 5) or Dossal (2007, Theorem 3) in the Lasso case, and Roth and Fischer (2008) for the group Lasso setting. We now turn to the result about the zero patterns of the solution of the problem in Equation (2):

Theorem 2 *Assume that $Y = (y_1,\dots,y_n)^\top$ is a realization of an absolutely continuous probability distribution. Let k be the maximal number such that any k rows of the matrix $(x_1,\dots,x_n) \in \mathbb{R}^{p\times n}$ are linearly independent. For $\mu > 0$, any solution of the problem in Equation (2) with at most $k-1$ nonzero coefficients has a zero pattern in $\mathcal{Z} = \left\{\bigcup_{G\in\mathcal{G}'} G;\ \mathcal{G}' \subseteq \mathcal{G}\right\}$ almost surely.*

In other words, when $Y = (y_1,\dots,y_n)^\top$ is a realization of an absolutely continuous probability distribution, the sparse solutions have a zero pattern in $\mathcal{Z} = \left\{\bigcup_{G\in\mathcal{G}'} G;\ \mathcal{G}' \subseteq \mathcal{G}\right\}$ almost surely. As a corollary of our two results, if the Gram matrix Q is invertible, the problem in Equation (2) has a unique solution, whose zero pattern belongs to $\mathcal{Z}$ almost surely. Note that with the assumption made on Y, Theorem 2 is not directly applicable to the classification setting. Based on these previous results, we can look at the following usual special cases from Section 2 (we give more examples in Section 3.5):

- **ℓ_2-norm**: the set of allowed nonzero patterns is composed of the empty set and the full set $\{1,\dots,p\}$.
- **ℓ_1-norm**: $\mathcal{P}$ is the set of all possible subsets.
- **ℓ_2-norm and ℓ_1-norm**: $\mathcal{P}$ is also the set of all possible subsets.
- **Group ℓ_1-norm**: $\mathcal{P} = \mathcal{Z}$ is the set of all possible unions of the elements of the partition defining $\mathcal{G}$.
- **Hierarchical norms**: the set of patterns $\mathcal{P}$ is then all sets J for which all ancestors of elements in J are included in J (Bach, 2008a).

Two natural questions now arise: (1) starting from the groups $\mathcal{G}$, is there an efficient way to generate the set of nonzero patterns $\mathcal{P}$; (2) conversely, and more importantly, given $\mathcal{P}$, how can the groups $\mathcal{G}$—and hence the norm $\Omega(w)$—be designed?

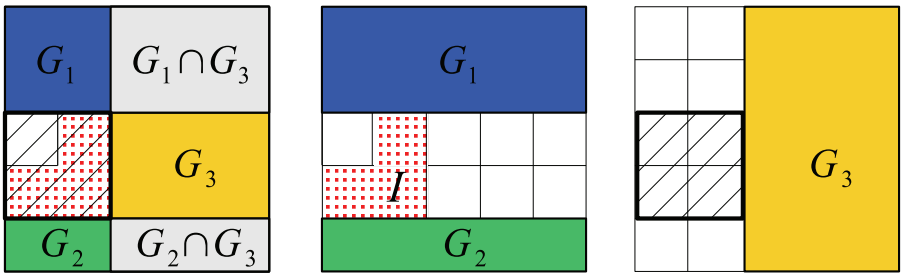


Figure 2: $\mathcal{G}$-adapted hull: the pattern of variables I (left and middle, red dotted surface) and its hull (left and right, hatched square) that is defined by the complement of the union of groups that do not intersect I, that is, $(G_1 \cup G_2 \cup G_3)^c$.

3.2 General Properties of $\mathcal{G}$, $\mathcal{Z}$ and $\mathcal{P}$

We now study the different properties of the set of groups $\mathcal{G}$ and its corresponding sets of patterns $\mathcal{Z}$ and $\mathcal{P}$.

3.2.1 CLOSEDNESS

The set of zero patterns $\mathcal{Z}$ (respectively, the set of nonzero patterns $\mathcal{P}$) is closed under union (respectively, intersection), that is, for all $K \in \mathbb{N}$ and all $z_1, \ldots, z_K \in \mathcal{Z}$, $\bigcup_{k=1}^K z_k \in \mathcal{Z}$ (respectively, $p_1, \ldots, p_K \in \mathcal{P}$, $\bigcap_{k=1}^K p_k \in \mathcal{P}$). This implies that when "reverse-engineering" the set of nonzero patterns, we have to assume it is closed under intersection. Otherwise, the best we can do is to deal with its intersection-closure. For instance, if we consider a sequence (see Figure 4), we cannot take $\mathcal{P}$ to be the set of contiguous patterns with length two, since the intersection of such two patterns may result in a singleton (that does not belong to $\mathcal{P}$).

3.2.2 MINIMALITY

If a group in $\mathcal{G}$ is the union of other groups, it may be removed from $\mathcal{G}$ without changing the sets $\mathcal{Z}$ or $\mathcal{P}$. This is the main argument behind the pruning backward algorithm in Section 3.3. Moreover, this leads to the notion of a *minimal* set $\mathcal{G}$ of groups, which is such that for all $\mathcal{G}' \subseteq \mathcal{Z}$ whose union-closure spans $\mathcal{Z}$, we have $\mathcal{G} \subseteq \mathcal{G}'$. The existence and uniqueness of a minimal set is a consequence of classical results in set theory (Doignon and Falmagne, 1998). The elements of this minimal set are usually referred to as the *atoms* of $\mathcal{Z}$.

Minimal sets of groups are attractive in our setting because they lead to a smaller number of groups and lower computational complexity—for example, for 2 dimensional-grids with rectangular patterns, we have a quadratic possible number of rectangles, that is, $|\mathcal{Z}| = O(p^2)$, that can be generated by a minimal set $\mathcal{G}$ whose size is $|\mathcal{G}| = O(\sqrt{p})$.

3.2.3 HULL

Given a set of groups $\mathcal{G}$, we can define for any subset $I \subseteq \{1, \ldots, p\}$ the $\mathcal{G}$-*adapted hull*, or simply *hull*, as:

$$\text{Hull}(I) = \Big\{ \bigcup_{G \in \mathcal{G},\, G \cap I = \varnothing} G \Big\}^c,$$

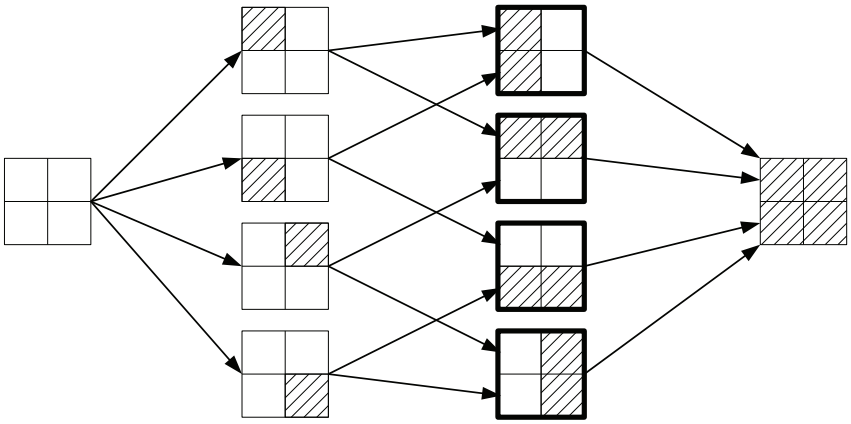


Figure 3: The DAG for the set $\mathcal{Z}$ associated with the 2×2-grid. The members of $\mathcal{Z}$ are the complement of the areas hatched in black. The elements of $\mathcal{G}$ (i.e., the atoms of $\mathcal{Z}$) are highlighted by bold edges.

which is the smallest set in $\mathcal{P}$ containing I (see Figure 2); we always have $I \subseteq \text{Hull}(I)$ with equality if and only if $I \in \mathcal{P}$. The hull has a clear geometrical interpretation for specific sets $\mathcal{G}$ of groups. For instance, if the set $\mathcal{G}$ is formed by all vertical and horizontal half-spaces when the variables are organized in a 2 dimensional-grid (see Figure 5), the hull of a subset $I \subset \{1,\dots,p\}$ is simply the axis-aligned bounding box of I. Similarly, when $\mathcal{G}$ is the set of all half-spaces with all possible orientations (e.g., orientations $\pm\pi/4$ are shown in Figure 6), the hull becomes the regular convex hull.[1] Note that those interpretations of the hull are possible and valid only when we have geometrical information at hand about the set of variables.

3.2.4 GRAPHS OF PATTERNS

We consider the directed acyclic graph (DAG) stemming from the *Hasse diagram* (Cameron, 1994) of the partially ordered set (poset) $(\mathcal{G},\supset)$. By definition, the nodes of this graph are the elements G of $\mathcal{G}$ and there is a directed edge from G_1 to G_2 if and only if $G_1 \supset G_2$ and there exists no $G \in \mathcal{G}$ such that $G_1 \supset G \supset G_2$ (Cameron, 1994). We can also build the corresponding DAG for the set of zero patterns $\mathcal{Z} \supset \mathcal{G}$, which is a super-DAG of the DAG of groups (see Figure 3 for examples). Note that we obtain also the isomorphic DAG for the nonzero patterns $\mathcal{P}$, although it corresponds to the poset $(\mathcal{P},\subset)$: this DAG will be used in the active set algorithm presented in Section 4.

Prior works with nested groups (Zhao et al., 2009; Bach, 2008a; Kim and Xing, 2010; Jenatton et al., 2010a; Schmidt and Murphy, 2010) have also used a similar DAG structure, with the slight difference that in these works, the corresponding hierarchy of variables is built from the prior knowledge about the problem at hand (e.g., the tree of wavelets in Zhao et al., 2009, the decomposition of kernels in Bach, 2008a or the hierarchy of genes in Kim and Xing, 2010). The DAG we introduce here on the set of groups naturally and always comes up, with no assumption on the variables themselves (for which no DAG is defined in general).

1. We use the term *convex* informally here. It can however be made precise with the notion of convex subgraphs (Chung, 1997).

3.3 From Patterns to Groups

We now assume that we want to impose a priori knowledge on the sparsity structure of a solution $\hat{w}$ of our regularized problem in Equation (2). This information can be exploited by restricting the patterns allowed by the norm Ω. Namely, from an intersection-closed set of zero patterns $\mathcal{Z}$, we can build back a minimal set of groups $\mathcal{G}$ by iteratively pruning away in the DAG corresponding to $\mathcal{Z}$, all sets which are unions of their parents. See Algorithm 1. This algorithm can be found under a different form in Doignon and Falmagne (1998)—we present it through a pruning algorithm on the DAG, which is natural in our context (the proof of the minimality of the procedure can be found in Appendix C). The complexity of Algorithm 1 is $O(p|\mathcal{Z}|^2)$. The pruning may reduce significantly the number of groups necessary to generate the whole set of zero patterns, sometimes from exponential in p to polynomial in p (e.g., the ℓ_1-norm). In Section 3.5, we give other examples of interest where $|\mathcal{G}|$ (and $|\mathcal{P}|$) is also polynomial in p.

Algorithm 1 Backward procedure

Input: Intersection-closed family of nonzero patterns $\mathcal{P}$.
Output: Set of groups $\mathcal{G}$.
Initialization: Compute $\mathcal{Z} = \{P^c;\ P \in \mathcal{P}\}$ and set $\mathcal{G} = \mathcal{Z}$.
Build the Hasse diagram for the poset $(\mathcal{Z}, \supset)$.
for $t = \min_{G\in\mathcal{Z}} |G|$ **to** $\max_{G\in\mathcal{Z}} |G|$ **do**
  for each node $G \in \mathcal{Z}$ such that $|G| = t$ **do**
    if $\left(\bigcup_{C\in\text{Children}(G)} C = G\right)$ **then**
      if $(\text{Parents}(G) \neq \varnothing)$ **then**
        Connect children of G to parents of G.
      end if
      Remove G from $\mathcal{G}$.
    end if
  end for
end for

3.4 From Groups to Patterns

The *forward* procedure presented in Algorithm 2, taken from Doignon and Falmagne (1998), allows the construction of $\mathcal{Z}$ from $\mathcal{G}$. It iteratively builds the collection of patterns by taking unions, and has complexity $O(p|\mathcal{Z}||\mathcal{G}|^2)$. The general scheme is straightforward. Namely, by considering increasingly larger sub-families of $\mathcal{G}$ and the collection of patterns already obtained, all possible unions are formed. However, some attention needs to be paid while checking we are not generating a pattern already encountered. Such a verification is performed by the *if* condition within the inner loop of the algorithm. Indeed, we do not have to scan the whole collection of patterns already obtained (whose size can be exponential in $|\mathcal{G}|$), but we rather use the fact that $\mathcal{G}$ generates $\mathcal{Z}$. Note that in general, it is not possible to upper bound the size of $|\mathcal{Z}|$ by a polynomial term in p, even when $\mathcal{G}$ is very small (indeed, $|\mathcal{Z}| = 2^p$ and $|\mathcal{G}| = p$ for the ℓ_1-norm).

Algorithm 2 Forward procedure

Input: Set of groups $\mathcal{G} = \{G_1, \ldots, G_M\}$.
Output: Collection of zero patterns $\mathcal{Z}$ and nonzero patterns $\mathcal{P}$.
Initialization: $\mathcal{Z} = \{\varnothing\}$.
for $m = 1$ **to** M **do**
  $C = \{\varnothing\}$
  for each $Z \in \mathcal{Z}$ **do**
    if $(G_m \not\subseteq Z)$ **and** $(\forall G \in \{G_1, \ldots, G_{m-1}\},\ G \subseteq Z \cup G_m \Rightarrow G \subseteq Z)$ **then**
      $C \leftarrow C \cup \{Z \cup G_m\}$.
    end if
  end for
  $\mathcal{Z} \leftarrow \mathcal{Z} \cup C$.
end for
$\mathcal{P} = \{Z^c;\ Z \in \mathcal{Z}\}$.

3.5 Examples

We now present several examples of sets of groups $\mathcal{G}$, especially suited to encode geometric and temporal prior information.

3.5.1 SEQUENCES

Given p variables organized in a sequence, if we want only contiguous nonzero patterns, the backward algorithm will lead to the set of groups which are intervals $[1,k]_{k\in\{1,\ldots,p-1\}}$ and $[k,p]_{k\in\{2,\ldots,p\}}$, with both $|\mathcal{Z}| = O(p^2)$ and $|\mathcal{G}| = O(p)$ (see Figure 4). Imposing the contiguity of the nonzero patterns is for instance relevant for the diagnosis of tumors, based on the profiles of arrayCGH (Rapaport et al., 2008).

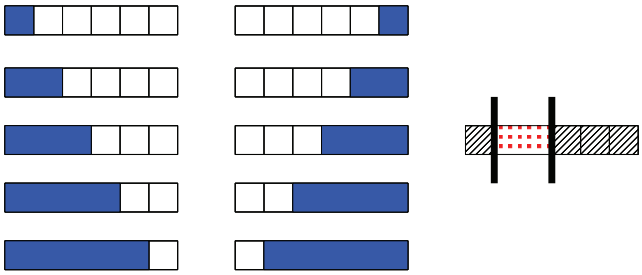


Figure 4: (Left and middle) The set of groups (blue areas) to penalize in order to select contiguous patterns in a sequence. (Right) An example of such a nonzero pattern (red dotted area) with its corresponding zero pattern (hatched area).

3.5.2 TWO-DIMENSIONAL GRIDS

In Section 6, we notably consider for $\mathcal{P}$ the set of all rectangles in two dimensions, leading by the previous algorithm to the set of axis-aligned half-spaces for $\mathcal{G}$ (see Figure 5), with $|\mathcal{Z}| = O(p^2)$

and $|\mathcal{G}| = O(\sqrt{p})$. This type of structure is encountered in object or scene recognition, where the selected rectangle would correspond to a certain box inside an image, that concentrates the predictive power for a given class of object/scene (Harzallah et al., 2009).

Larger set of convex patterns can be obtained by adding in $\mathcal{G}$ half-planes with other orientations than vertical and horizontal. For instance, if we use planes with angles that are multiples of $\pi/4$, the nonzero patterns of $\mathcal{P}$ can have polygonal shapes with up to 8 faces. In this sense, if we keep on adding half-planes with finer orientations, the nonzero patterns of $\mathcal{P}$ can be described by polygonal shapes with an increasingly larger number of faces. The standard notion of convexity defined in $\mathbb{R}^2$ would correspond to the situation where an infinite number of orientations is considered (Soille, 2003). See Figure 6. The number of groups is linear in $\sqrt{p}$ with constant growing linearly with the number of angles, while $|\mathcal{Z}|$ grows more rapidly (typically non-polynomially in the number of angles). Imposing such convex-like regions turns out to be useful in computer vision. For instance, in face recognition, it enables the design of localized features that improve upon the robustness to occlusions (Jenatton et al., 2010b). In the same vein, regularizations with similar two-dimensional sets of groups have led to good performances in background subtraction tasks (Mairal et al., 2010b), where the pixel spatial information is crucial to avoid scattered results. Another application worth being mentioned is the design of topographic dictionaries in the context of image processing (Kavukcuoglu et al., 2009; Mairal et al., 2011). In this case, dictionaries self-organize and adapt to the underlying geometrical structure encoded by the two-dimensional set of groups.

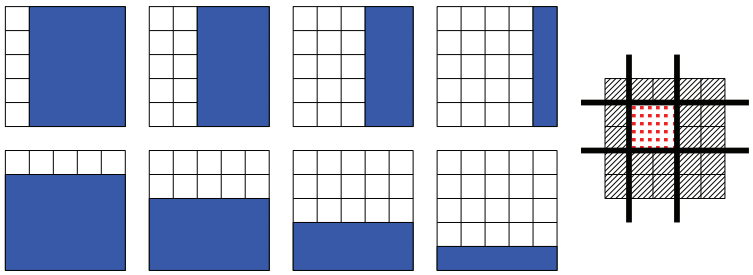


Figure 5: Vertical and horizontal groups: (Left) the set of groups (blue areas) with their (not displayed) complements to penalize in order to select rectangles. (Right) An example of nonzero pattern (red dotted area) recovered in this setting, with its corresponding zero pattern (hatched area).

3.5.3 EXTENSIONS

The sets of groups presented above can be straightforwardly extended to more complicated topologies, such as three-dimensional spaces discretized in cubes or spherical volumes discretized in slices. Similar properties hold for such settings. For instance, if all the axis-aligned half-spaces are considered for $\mathcal{G}$ in a three-dimensional space, then $\mathcal{P}$ is the set of all possible rectangular boxes with $|\mathcal{P}| = O(p^2)$ and $|\mathcal{G}| = O(p^{1/3})$. Such three-dimensional structures are interesting to retrieve discriminative and local sets of voxels from fMRI/MEEG responses. In particular, they have recently proven useful for modelling brain resting-state activity (Varoquaux et al., 2010). Moreover, while the two-dimensional rectangular patterns described previously are adapted to find bounding boxes in static images (Harzallah et al., 2009), scene recognition in videos requires to deal with a third temporal dimension (Dalal et al., 2006). This may be achieved by designing appropriate sets of

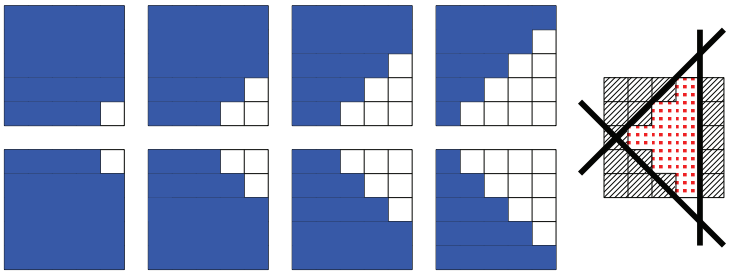


Figure 6: Groups with $\pm\pi/4$ orientations: (Left) the set of groups (blue areas) with their (not displayed) complements to penalize in order to select diamond-shaped patterns. (Right) An example of nonzero pattern (red dotted area) recovered in this setting, with its corresponding zero pattern (hatched area).

groups, embedded in the three-dimensional space obtained by tracking the frames over time. Finally, in the context of matrix-based optimization problems, for example, multi-task learning and dictionary learning, sets of groups $\mathcal{G}$ can also be designed to encode *structural constraints* the solutions must respect. This notably encompasses banded structures (Levina et al., 2008) and *simultaneous* row/column sparsity for CUR matrix factorization (Mairal et al., 2011).

3.5.4 REPRESENTATION AND COMPUTATION OF $\mathcal{G}$

The sets of groups described so far can actually be represented in a same form, that lends itself well to the analysis of the next section. When dealing with a discrete sequence of length l (see Figure 4), we have

$$\begin{aligned}\mathcal{G} &= \{g_-^k;\, k\in\{1,\dots,l-1\}\}\cup\{g_+^k;\, k\in\{2,\dots,l\}\},\\ &= \mathcal{G}_{\text{left}}\cup\mathcal{G}_{\text{right}},\end{aligned}$$

with $g_-^k=\{i;\, 1\le i\le k\}$ and $g_+^k=\{i;\, l\ge i\ge k\}$. In other words, the set of groups $\mathcal{G}$ can be rewritten as a partition[2] in two sets of nested groups, $\mathcal{G}_{\text{left}}$ and $\mathcal{G}_{\text{right}}$.

The same goes for a two-dimensional grid, with dimensions $h\times l$ (see Figure 5 and Figure 6). In this case, the nested groups we consider are defined based on the following groups of variables

$$g^{k,\theta}=\{(i,j)\in\{1,\dots,l\}\times\{1,\dots,h\};\, \cos(\theta)i+\sin(\theta)j\le k\},$$

where $k\in\mathbb{Z}$ is taken in an appropriate range. The nested groups we obtain in this way are therefore parameterized by an angle[3] θ, $\theta\in(-\pi;\pi]$. We refer to this angle as an *orientation*, since it defines the normal vector $\binom{\cos(\theta)}{\sin(\theta)}$ to the line $\{(i,j)\in\mathbb{R}^2;\cos(\theta)i+\sin(\theta)j=k\}$. In the example of the rectangular groups (see Figure 5), we have four orientations, with $\theta\in\{0,\pi/2,-\pi/2,\pi\}$. More generally, if we denote by Θ the set of the orientations, we have

$$\mathcal{G}=\bigcup_{\theta\in\Theta}\mathcal{G}_\theta,$$

2. Note the subtlety: the sets $\mathcal{G}_\theta$ are disjoint, that is $\mathcal{G}_\theta\cap\mathcal{G}_{\theta'}=\varnothing$ for $\theta\neq\theta'$, but groups in $\mathcal{G}_\theta$ and $\mathcal{G}_{\theta'}$ can overlap.
3. Due to the discrete nature of the underlying geometric structure of $\mathcal{G}$, angles θ that are not multiple of $\pi/4$ (i.e., such that $\tan(\theta)\notin\mathbb{Z}$) are dealt with by rounding operations.

where $\theta \in \Theta$ indexes the partition of $\mathcal{G}$ in sets $\mathcal{G}_\theta$ of nested groups of variables. Although we have not detailed the case of $\mathbb{R}^3$, we likewise end up with a similar partition of $\mathcal{G}$.

4. Optimization and Active Set Algorithm

For moderate values of p, one may obtain a solution for Problem (2) using generic toolboxes for second-order cone programming (SOCP) whose time complexity is equal to $O(p^{3.5} + |\mathcal{G}|^{3.5})$ (Boyd and Vandenberghe, 2004), which is not appropriate when p or $|\mathcal{G}|$ are large. This time complexity corresponds to the computation of a solution of Problem (2) for a single value of the regularization parameter μ.

We present in this section an *active set algorithm* (Algorithm 3) that finds a solution for Problem (2) by considering increasingly larger active sets and checking global optimality at each step. When the rectangular groups are used, the total complexity of this method is in $O(s \max\{p^{1.75}, s^{3.5}\})$, where s is the size of the active set at the end of the optimization. Here, the sparsity prior is exploited for computational advantages. Our active set algorithm needs an underlying *black-box* SOCP solver; in this paper, we consider both a first order approach (see Appendix H) and a SOCP method[4]—in our experiments, we use `SDPT3` (Toh et al., 1999; Tütüncü et al., 2003). Our active set algorithm extends to general overlapping groups the work of Bach (2008a), by further assuming that it is computationally possible to have a time complexity polynomial in the number of variables p.

We primarily focus here on finding an efficient active set algorithm; we defer to future work the design of specific SOCP solvers, for example, based on proximal techniques (see, e.g., Nesterov, 2007; Beck and Teboulle, 2009; Combettes and Pesquet, 2010, and numerous references therein), adapted to such non-smooth sparsity-inducing penalties.

4.1 Optimality Conditions: From Reduced Problems to Full Problems

It is simpler to derive the algorithm for the following regularized optimization problem[5] which has the same solution set as the regularized problem of Equation (2) when μ and λ are allowed to vary (Borwein and Lewis, 2006, see Section 3.2):

$$\min_{w \in \mathbb{R}^p} \frac{1}{n} \sum_{i=1}^{n} \ell\big(y_i, w^\top x_i\big) + \frac{\lambda}{2} \big[\Omega(w)\big]^2 . \tag{3}$$

In active set methods, the set of nonzero variables, denoted by J, is built incrementally, and the problem is solved only for this reduced set of variables, adding the constraint $w_{J^c} = 0$ to Equation (3). In the subsequent analysis, we will use arguments based on duality to monitor the optimality of our active set algorithm. We denote by $L(w) = \frac{1}{n} \sum_{i=1}^{n} \ell\big(y_i, w^\top x_i\big)$ the empirical risk (which is by assumption convex and continuously differentiable) and by L^* its *Fenchel-conjugate*, defined as (Boyd and Vandenberghe, 2004; Borwein and Lewis, 2006):

$$L^*(u) = \sup_{w \in \mathbb{R}^p} \{w^\top u - L(w)\}.$$

4. The C++/Matlab code used in the experiments may be downloaded from the authors website.

5. It is also possible to derive the active set algorithm for the constrained formulation $\min_{w \in \mathbb{R}^p} \frac{1}{n} \sum_{i=1}^{n} \ell\big(y_i, w^\top x_i\big)$ such that $\Omega(w) \leq \lambda$. However, we empirically found it more difficult to select λ in this latter formulation.

The restriction of L to $\mathbb{R}^{|J|}$ is denoted $L_J(w_J) = L(\tilde{w})$ for $\tilde{w}_J = w_J$ and $\tilde{w}_{J^c} = 0$, with Fenchel-conjugate L_J^*. Note that, as opposed to L, we do not have in general $L_J^*(\kappa_J) = L^*(\tilde{\kappa})$ for $\tilde{\kappa}_J = \kappa_J$ and $\tilde{\kappa}_{J^c} = 0$.

For a potential active set $J \subset \{1,\dots,p\}$ which belongs to the set of allowed nonzero patterns $\mathcal{P}$, we denote by $\mathcal{G}_J$ the set of active groups, that is, the set of groups $G \in \mathcal{G}$ such that $G \cap J \neq \varnothing$. We consider the reduced norm Ω_J defined on $\mathbb{R}^{|J|}$ as

$$\Omega_J(w_J) = \sum_{G \in \mathcal{G}} \|d_J^G \circ w_J\|_2 = \sum_{G \in \mathcal{G}_J} \|d_J^G \circ w_J\|_2 ,$$

and its *dual norm* $\Omega_J^*(\kappa_J) = \max_{\Omega_J(w_J) \leq 1} w_J^\top \kappa_J$, also defined on $\mathbb{R}^{|J|}$. The next proposition (see proof in Appendix D) gives the optimization problem dual to the reduced problem (Equation (4) below):

Proposition 3 (Dual Problems) *Let $J \subseteq \{1,\dots,p\}$. The following two problems*

$$\min_{w_J \in \mathbb{R}^{|J|}} L_J(w_J) + \frac{\lambda}{2} [\Omega_J(w_J)]^2 , \tag{4}$$

$$\max_{\kappa_J \in \mathbb{R}^{|J|}} -L_J^*(-\kappa_J) - \frac{1}{2\lambda} [\Omega_J^*(\kappa_J)]^2 ,$$

are dual to each other and strong duality holds. The pair of primal-dual variables $\{w_J, \kappa_J\}$ is optimal if and only if we have

$$\begin{cases} \kappa_J & = -\nabla L_J(w_J), \\ w_J^\top \kappa_J & = \frac{1}{\lambda} [\Omega_J^*(\kappa_J)]^2 = \lambda [\Omega_J(w_J)]^2 . \end{cases}$$

As a brief reminder, the duality gap of a minimization problem is defined as the difference between the primal and dual objective functions, evaluated for a feasible pair of primal/dual variables (Boyd and Vandenberghe, 2004, see Section 5.5). This gap serves as a certificate of (sub)optimality: if it is equal to zero, then the optimum is reached, and provided that strong duality holds, the converse is true as well (Boyd and Vandenberghe, 2004, see Section 5.5).

The previous proposition enables us to derive the duality gap for the optimization Problem (4), that is reduced to the active set of variables J. In practice, this duality gap will always vanish (up to the precision of the underlying SOCP solver), since we will sequentially solve Problem (4) for increasingly larger active sets J. We now study how, starting from the optimality of the problem in Equation (4), we can control the optimality, or equivalently the duality gap, for the full problem in Equation (3). More precisely, the duality gap of the optimization problem in Equation (4) is

$$\begin{aligned} & L_J(w_J) + L_J^*(-\kappa_J) + \frac{\lambda}{2} [\Omega_J(w_J)]^2 + \frac{1}{2\lambda} [\Omega_J^*(\kappa_J)]^2 \\ = \; & \left\{ L_J(w_J) + L_J^*(-\kappa_J) + w_J^\top \kappa_J \right\} + \left\{ \frac{\lambda}{2} [\Omega_J(w_J)]^2 + \frac{1}{2\lambda} [\Omega_J^*(\kappa_J)]^2 - w_J^\top \kappa_J \right\} , \end{aligned}$$

which is a sum of two nonnegative terms, the nonnegativity coming from the Fenchel-Young inequality (Borwein and Lewis, 2006; Boyd and Vandenberghe, 2004, Proposition 3.3.4 and Section 3.3.2 respectively). We can think of this duality gap as the sum of two duality gaps, respectively

relative to L_J and Ω_J. Thus, if we have a primal candidate w_J and we choose $\kappa_J = -\nabla L_J(w_J)$, the duality gap relative to L_J vanishes and the total duality gap then reduces to

$$\frac{\lambda}{2}\left[\Omega_J(w_J)\right]^2 + \frac{1}{2\lambda}\left[\Omega_J^*(\kappa_J)\right]^2 - w_J^\top \kappa_J.$$

In order to check that the reduced solution w_J is optimal for the full problem in Equation (3), we pad w_J with zeros on J^c to define w and compute $\kappa = -\nabla L(w)$, which is such that $\kappa_J = -\nabla L_J(w_J)$. For our given candidate pair of primal/dual variables $\{w, \kappa\}$, we then get a duality gap for the full problem in Equation (3) equal to

$$\begin{aligned}
& \frac{\lambda}{2}\left[\Omega(w)\right]^2 + \frac{1}{2\lambda}\left[\Omega^*(\kappa)\right]^2 - w^\top \kappa \\
= & \frac{\lambda}{2}\left[\Omega(w)\right]^2 + \frac{1}{2\lambda}\left[\Omega^*(\kappa)\right]^2 - w_J^\top \kappa_J \\
= & \frac{\lambda}{2}\left[\Omega(w)\right]^2 + \frac{1}{2\lambda}\left[\Omega^*(\kappa)\right]^2 - \frac{\lambda}{2}\left[\Omega_J(w_J)\right]^2 - \frac{1}{2\lambda}\left[\Omega_J^*(\kappa_J)\right]^2 \\
= & \frac{1}{2\lambda}\left(\left[\Omega^*(\kappa)\right]^2 - \left[\Omega_J^*(\kappa_J)\right]^2\right) \\
= & \frac{1}{2\lambda}\left(\left[\Omega^*(\kappa)\right]^2 - \lambda w_J^\top \kappa_J\right).
\end{aligned}$$

Computing this gap requires computing the dual norm which itself is as hard as the original problem, prompting the need for upper and lower bounds on Ω^* (see Propositions 4 and 5 for more details).

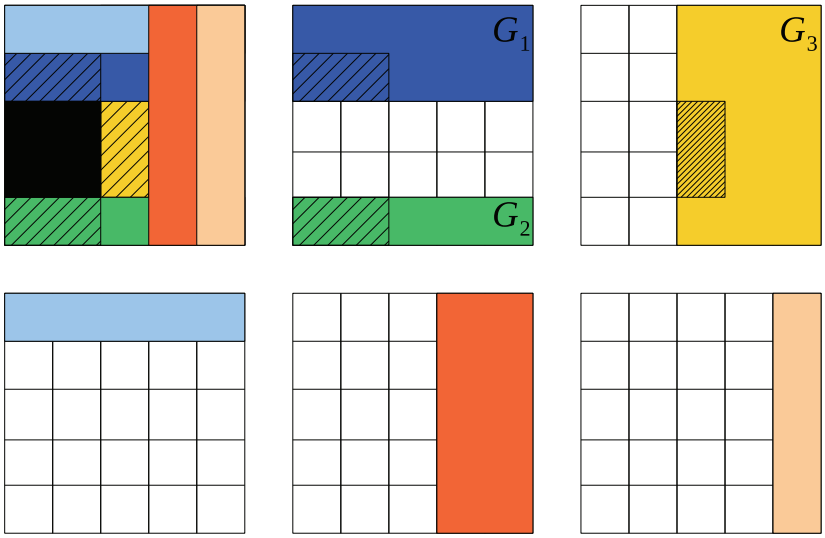


Figure 7: The active set (black) and the candidate patterns of variables, that is, the variables in $K \backslash J$ (hatched in black) that can become active. The fringe groups are exactly the groups that have the hatched areas (i.e., here we have $\mathcal{F}_J = \bigcup_{K \in \Pi_{\mathcal{P}}(J)} \mathcal{G}_K \backslash \mathcal{G}_J = \{G_1, G_2, G_3\}$).

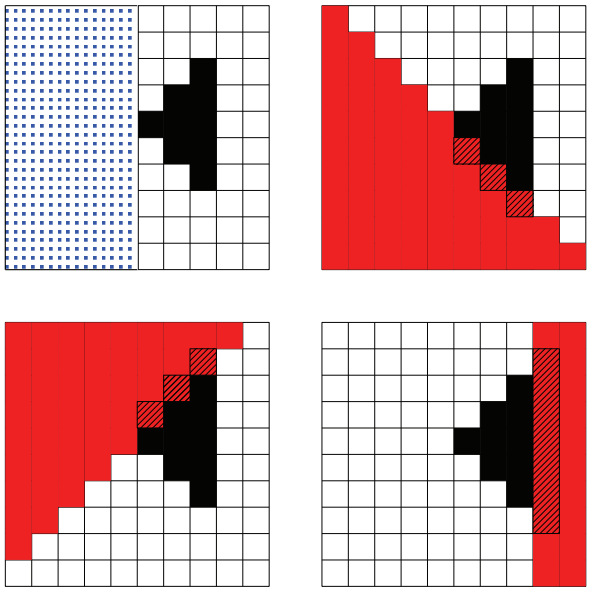


Figure 8: The active set (black) and the candidate patterns of variables, that is, the variables in $K \backslash J$ (hatched in black) that can become active. The groups in red are those in $\bigcup_{K \in \Pi_{\mathcal{P}}(J)} \mathcal{G}_K \backslash \mathcal{G}_J$, while the blue dotted group is in $\mathcal{F}_J \backslash (\bigcup_{K \in \Pi_{\mathcal{P}}(J)} \mathcal{G}_K \backslash \mathcal{G}_J)$. The blue dotted group does not intersect with any patterns in $\Pi_{\mathcal{P}}(J)$.

4.2 Active Set Algorithm

We can interpret the active set algorithm as a walk through the DAG of nonzero patterns allowed by the norm Ω. The parents $\Pi_{\mathcal{P}}(J)$ of J in this DAG are exactly the patterns containing the variables that may enter the active set at the next iteration of Algorithm 3. The groups that are exactly at the boundaries of the active set (referred to as the *fringe groups*) are $\mathcal{F}_J = \{G \in (\mathcal{G}_J)^c \ ; \ \nexists G' \in (\mathcal{G}_J)^c, \ G \subseteq G'\}$, that is, the groups that are not contained by any other inactive groups.

In simple settings, for example, when $\mathcal{G}$ is the set of rectangular groups, the correspondence between groups and variables is straightforward since we have $\mathcal{F}_J = \bigcup_{K \in \Pi_{\mathcal{P}}(J)} \mathcal{G}_K \backslash \mathcal{G}_J$ (see Figure 7). However, in general, we just have the inclusion $(\bigcup_{K \in \Pi_{\mathcal{P}}(J)} \mathcal{G}_K \backslash \mathcal{G}_J) \subseteq \mathcal{F}_J$ and some elements of $\mathcal{F}_J$ might not correspond to any patterns of variables in $\Pi_{\mathcal{P}}(J)$ (see Figure 8).

We now present the optimality conditions (see proofs in Appendix E) that monitor the progress of Algorithm 3:

Proposition 4 (Necessary condition) *If w is optimal for the full problem in Equation (3), then*

$$\max_{K \in \Pi_{\mathcal{P}}(J)} \frac{\left\| \nabla L(w)_{K \backslash J} \right\|_2}{\sum_{H \in \mathcal{G}_K \backslash \mathcal{G}_J} \left\| d^H_{K \backslash J} \right\|_\infty} \leq \left\{ -\lambda w^\top \nabla L(w) \right\}^{\frac{1}{2}}. \tag{N}$$

Proposition 5 (Sufficient condition) *If*

$$\max_{G \in \mathcal{F}_J} \left\{ \sum_{k \in G} \left\{ \frac{\nabla L(w)_k}{\sum_{H \ni k, \ H \in (\mathcal{G}_J)^c} d^H_k} \right\}^2 \right\}^{\frac{1}{2}} \leq \left\{ \lambda (2\varepsilon - w^\top \nabla L(w)) \right\}^{\frac{1}{2}}, \tag{S_ε}$$

then w is an approximate solution for Equation (3) whose duality gap is less than $\varepsilon \geq 0$.

Note that for the Lasso, the conditions (N) and (S_0) (i.e., the sufficient condition taken with $\varepsilon = 0$) are both equivalent (up to the squaring of Ω) to the condition $\|\nabla L(w)_{J^c}\|_\infty \leq -w^\top \nabla L(w)$, which is the usual optimality condition (Fuchs, 2005; Tibshirani, 1996; Wainwright, 2009). Moreover, when they are not satisfied, our two conditions provide good heuristics for choosing which $K \in \Pi_{\mathcal{P}}(J)$ should enter the active set.

More precisely, since the necessary condition (N) directly deals with the *variables* (as opposed to groups) that can become active at the next step of Algorithm 3, it suffices to choose the pattern $K \in \Pi_{\mathcal{P}}(J)$ that violates most the condition.

The heuristics for the sufficient condition (S_ε) implies that, to go from groups to variables, we simply consider the group $G \in \mathcal{F}_J$ violating the sufficient condition the most and then take all the patterns of variables $K \in \Pi_{\mathcal{P}}(J)$ such that $K \cap G \neq \varnothing$ to enter the active set. If $G \cap (\bigcup_{K \in \Pi_{\mathcal{P}}(J)} K) = \varnothing$, we look at all the groups $H \in \mathcal{F}_J$ such that $H \cap G \neq \varnothing$ and apply the scheme described before (see Algorithm 4).

A direct consequence of this heuristics is that it is possible for the algorithm to *jump over* the right active set and to consider instead a (slightly) larger active set as optimal. However if the active set is larger than the optimal set, then (it can be proved that) the sufficient condition (S_0) is satisfied, and the reduced problem, which we solve exactly, will still output the correct nonzero pattern.

Moreover, it is worthwhile to notice that in Algorithm 3, the active set may sometimes be increased only to make sure that the current solution is optimal (we only check a sufficient condition of optimality).

Algorithm 3 Active set algorithm

Input: Data $\{(x_i, y_i), i = 1, \ldots, n\}$, regularization parameter λ,
Duality gap precision ε, maximum number of variables s.
Output: Active set J, loading vector $\hat{w}$.
Initialization: $J = \{\varnothing\}$, $\hat{w} = 0$.
while $\big($ (N) is not satisfied $\big)$ **and** $\big($ $|J| \leq s$ $\big)$ **do**
  Replace J by violating $K \in \Pi_{\mathcal{P}}(J)$ in (N).
  Solve the reduced problem $\min_{w_J \in \mathbb{R}^{|J|}} L_J(w_J) + \frac{\lambda}{2}\left[\Omega_J(w_J)\right]^2$ to get $\hat{w}$.
end while
while $\big($ (S_ε) is not satisfied $\big)$ **and** $\big($ $|J| \leq s$ $\big)$ **do**
  Update J according to Algorithm 4.
  Solve the reduced problem $\min_{w_J \in \mathbb{R}^{|J|}} L_J(w_J) + \frac{\lambda}{2}\left[\Omega_J(w_J)\right]^2$ to get $\hat{w}$.
end while

4.2.1 CONVERGENCE OF THE ACTIVE SET ALGORITHM

The procedure described in Algorithm 3 can terminate in two different states. If the procedure stops because of the limit on the number of active variables s, the solution might be suboptimal. Note that, in any case, we have at our disposal a upper-bound on the duality gap.

Otherwise, the procedure always converges to an optimal solution, either (1) by validating both the necessary and sufficient conditions (see Propositions 4 and 5), ending up with fewer than p active variables and a precision of (at least) ε, or (2) by running until the p variables become active, the precision of the solution being given by the underlying solver.

Algorithm 4 Heuristics for the sufficient condition (S_ε)

Let $G \in \mathcal{F}_J$ be the group that violates (S_ε) most.
if $(G \cap (\bigcup_{K \in \Pi_{\mathcal{P}}(J)} K) \neq \varnothing)$ **then**
  for $K \in \Pi_{\mathcal{P}}(J)$ such that $K \cap G \neq \varnothing$ **do**
    $J \leftarrow J \cup K$.
  end for
else
  for $H \in \mathcal{F}_J$ such that $H \cap G \neq \varnothing$ **do**
    for $K \in \Pi_{\mathcal{P}}(J)$ such that $K \cap H \neq \varnothing$ **do**
      $J \leftarrow J \cup K$.
    end for
  end for
end if

4.2.2 ALGORITHMIC COMPLEXITY

We analyze in detail the time complexity of the active set algorithm when we consider sets of groups $\mathcal{G}$ such as those presented in the examples of Section 3.5. We recall that we denote by Θ the set of orientations in $\mathcal{G}$ (for more details, see Section 3.5). For such choices of $\mathcal{G}$, the fringe groups $\mathcal{F}_J$ reduces to the largest groups of each orientation and therefore $|\mathcal{F}_J| \leq |\Theta|$. We further assume that the groups in $\mathcal{G}_\theta$ are sorted by cardinality, so that computing $\mathcal{F}_J$ costs $O(|\Theta|)$.

Given an active set J, both the necessary and sufficient conditions require to have access to the direct parents $\Pi_{\mathcal{P}}(J)$ of J in the DAG of nonzero patterns. In simple settings, for example, when $\mathcal{G}$ is the set of rectangular groups, this operation can be performed in $O(1)$ (it just corresponds to scan the (up to) four patterns at the edges of the current rectangular hull).

However, for more general orientations, computing $\Pi_{\mathcal{P}}(J)$ requires to find the smallest nonzero patterns that we can generate from the groups in $\mathcal{F}_J$, reduced to the stripe of variables around the current hull. This stripe of variables can be computed as $\left[\bigcup_{G \in (\mathcal{G}_J)^c \setminus \mathcal{F}_J} G\right]^c \setminus J$, so that getting $\Pi_{\mathcal{P}}(J)$ costs $O(s2^{|\Theta|} + p|\mathcal{G}|)$ in total.

Thus, if the number of active variables is upper bounded by $s \ll p$ (which is true if our target is actually sparse), the time complexity of Algorithm 3 is the sum of:

- the computation of the gradient, $O(snp)$ for the square loss.
- if the underlying solver called upon by the active set algorithm is a standard SOCP solver, $O(s \max_{J \in \mathcal{P}, |J| \leq s} |\mathcal{G}_J|^{3.5} + s^{4.5})$ (note that the term $s^{4.5}$ could be improved upon by using warm-restart strategies for the sequence of reduced problems).
- t_1 times the computation of (N), that is $O(t_1(s2^{|\Theta|} + p|\mathcal{G}| + sn_\theta^2) + p|\mathcal{G}|) = O(t_1 p|\mathcal{G}|)$.

  During the initialization (i.e., $J = \varnothing$), we have $|\Pi_{\mathcal{P}}(\varnothing)| = O(p)$ (since we can start with any singletons), and $|\mathcal{G}_K \setminus \mathcal{G}_J| = |\mathcal{G}_K| = |\mathcal{G}|$, which leads to a complexity of $O(p|\mathcal{G}|)$ for the sum $\sum_{G \in \mathcal{G}_K \setminus \mathcal{G}_J} = \sum_{G \in \mathcal{G}_K}$. Note however that this sum does not depend on J and can therefore be cached if we need to make several runs with the same set of groups $\mathcal{G}$.
- t_2 times the computation of (S_ε), that is $O(t_2(s2^{|\Theta|} + p|\mathcal{G}| + |\Theta|^2 + |\Theta|p + p|\mathcal{G}|)) = O(t_2 p|\mathcal{G}|)$, with $t_1 + t_2 \leq s$.

We finally get complexity with a leading term in $O(sp|\mathcal{G}| + s\max_{J\in\mathcal{P},|J|\leq s}|\mathcal{G}_J|^{3.5} + s^{4.5})$, which is much better than $O(p^{3.5} + |\mathcal{G}|^{3.5})$, without an active set method. In the example of the two-dimensional grid (see Section 3.5), we have $|\mathcal{G}| = O(\sqrt{p})$ and $O(s\max\{p^{1.75}, s^{3.5}\})$ as total complexity. The simulations of Section 6 confirm that the active set strategy is indeed useful when s is much smaller than p. Moreover, the two extreme cases where $s \approx p$ or $p \ll 1$ are also shown not to be advantageous for the active set strategy, since either it is cheaper to use the SOCP solver directly on the p variables, or we uselessly pay the additional fixed-cost of the active set machinery (such as computing the optimality conditions). Note that we have derived here the *theoretical* complexity of the active set algorithm when we use an interior point method as underlying solver. With the first order method presented in Appendix H, we would instead get a total complexity in $O(sp^{1.5})$.

4.3 Intersecting Nonzero Patterns

We have seen so far how overlapping groups can encore prior information about a desired set of (non)zero patterns. In practice, controlling these overlaps may be delicate and hinges on the choice of the weights $(d^G)_{G\in\mathcal{G}}$ (see the experiments in Section 6). In particular, the weights have to take into account that some variables belonging to several overlapping groups are penalized multiple times.

However, it is possible to keep the benefit of overlapping groups whilst limiting their side effects, by taking up the idea of support intersection (Bach, 2008c; Meinshausen and Bühlmann, 2010). First introduced to stabilize the set of variables recovered by the Lasso, we reuse this technique in a different context, based on the fact that $\mathcal{Z}$ is closed under union.

If we deal with the same sets of groups as those considered in Section 3.5, it is natural to rewrite $\mathcal{G}$ as $\bigcup_{\theta\in\Theta}\mathcal{G}_\theta$, where Θ is the set of the orientations of the groups in $\mathcal{G}$ (for more details, see Section 3.5). Let us denote by $\hat{w}$ and $\hat{w}^\theta$ the solutions of Problem (3), where the regularization term Ω is respectively defined by the groups in $\mathcal{G}$ and by the groups[6] in $\mathcal{G}_\theta$.

The main point is that, since $\mathcal{P}$ is closed under intersection, the two procedures described below actually lead to the same set of allowed nonzero patterns:

a) Simply considering the nonzero pattern of $\hat{w}$.

b) Taking the *intersection* of the nonzero patterns obtained for each $\hat{w}^\theta$, θ in Θ.

With the latter procedure, although the learning of several models $\hat{w}^\theta$ is required (a number of times equals to the number of orientations considered, for example, 2 for the sequence, 4 for the rectangular groups and more generally $|\Theta|$ times), each of those learning tasks involves a smaller number of groups (that is, just the ones belonging to $\mathcal{G}_\theta$). In addition, this procedure is a *variable selection* technique that therefore needs a second step for estimating the loadings (restricted to the selected nonzero pattern). In the experiments, we follow Bach (2008c) and we use an ordinary least squares (OLS). The simulations of Section 6 will show the benefits of this variable selection approach.

6. To be more precise, in order to regularize every variable, we add the full group $\{1,\ldots,p\}$ to $\mathcal{G}_\theta$, which does not modify $\mathcal{P}$.

5. Pattern Consistency

In this section, we analyze the model consistency of the solution of the problem in Equation (2) for the square loss. Considering the set of nonzero patterns $\mathcal{P}$ derived in Section 3, we can only hope to estimate the correct hull of the generating sparsity pattern, since Theorem 2 states that other patterns occur with zero probability. We derive necessary and sufficient conditions for model consistency in a low-dimensional setting, and then consider a high-dimensional result.

We consider the square loss and a fixed-design analysis (i.e., $x_1, \ldots, x_n$ are fixed). The extension of the following consistency results to other loss functions is beyond the scope of the paper (see for instance Bach, 2009). We assume that for all $i \in \{1, \ldots, n\}$, $y_i = \mathbf{w}^\top x_i + \varepsilon_i$ where the vector ε is an i.i.d. vector with Gaussian distributions with mean zero and variance $\sigma^2 > 0$, and $\mathbf{w} \in \mathbb{R}^p$ is the population sparse vector; we denote by $\mathbf{J}$ the $\mathcal{G}$-adapted hull of its nonzero pattern. Note that estimating the $\mathcal{G}$-adapted hull of $\mathbf{w}$ is equivalent to estimating the nonzero pattern of $\mathbf{w}$ if and only if this nonzero pattern belongs to $\mathcal{P}$. This happens when our prior information has led us to consider an appropriate set of groups $\mathcal{G}$. Conversely, if $\mathcal{G}$ is misspecified, recovering the hull of the nonzero pattern of $\mathbf{w}$ may be irrelevant, which is for instance the case if $\mathbf{w} = \binom{\mathbf{w}_1}{0} \in \mathbb{R}^2$ and $\mathcal{G} = \{\{1\}, \{1,2\}\}$. Finding the appropriate structure of $\mathcal{G}$ *directly from the data* would therefore be interesting future work.

5.1 Consistency Condition

We begin with the low-dimensional setting where n is tending to infinity with p *fixed*. In addition, we also assume that the design is *fixed* and that the Gram matrix $Q = \frac{1}{n}\sum_{i=1}^n x_i x_i^\top$ is invertible with positive-definite (i.e., invertible) limit: $\lim_{n\to\infty} Q = \mathbf{Q} \succ 0$. In this setting, the noise is the only source of randomness. We denote by $\mathbf{r}_\mathbf{J}$ the vector defined as

$$\forall j \in \mathbf{J},\ \mathbf{r}_j = \mathbf{w}_j \Bigg(\sum_{G \in \mathcal{G}_\mathbf{J}, G \ni j} (d_j^G)^2 \, \|d^G \circ \mathbf{w}\|_2^{-1} \Bigg).$$

In the Lasso and group Lasso setting, the vector $\mathbf{r}_\mathbf{J}$ is respectively the sign vector $\operatorname{sign}(\mathbf{w}_\mathbf{J})$ and the vector defined by the blocks $(\frac{\mathbf{w}_G}{\|\mathbf{w}_G\|_2})_{G \in \mathcal{G}_\mathbf{J}}$.

We define $\Omega_\mathbf{J}^c(w_{\mathbf{J}^c}) = \sum_{G \in (\mathcal{G}_\mathbf{J})^c} \left\| d_{\mathbf{J}^c}^G \circ w_{\mathbf{J}^c} \right\|_2$ (which is the norm composed of inactive groups) with its dual norm $(\Omega_\mathbf{J}^c)^*$; note the difference with the norm reduced to $\mathbf{J}^c$, defined as $\Omega_{\mathbf{J}^c}(w_{\mathbf{J}^c}) = \sum_{G \in \mathcal{G}} \left\| d_{\mathbf{J}^c}^G \circ w_{\mathbf{J}^c} \right\|_2$.

The following Theorem gives the sufficient and necessary conditions under which the hull of the generating pattern is consistently estimated. Those conditions naturally extend the results of Zhao and Yu (2006) and Bach (2008b) for the Lasso and the group Lasso respectively (see proof in Appendix F).

Theorem 6 (Consistency condition) *Assume $\mu \to 0$, $\mu\sqrt{n} \to \infty$ in Equation (2). If the hull is consistently estimated, then $(\Omega_\mathbf{J}^c)^*[\mathbf{Q}_{\mathbf{J}^c\mathbf{J}}\mathbf{Q}_{\mathbf{J}\mathbf{J}}^{-1}\mathbf{r}_\mathbf{J}] \leq 1$. Conversely, if $(\Omega_\mathbf{J}^c)^*[\mathbf{Q}_{\mathbf{J}^c\mathbf{J}}\mathbf{Q}_{\mathbf{J}\mathbf{J}}^{-1}\mathbf{r}_\mathbf{J}] < 1$, then the hull is consistently estimated, that is,*

$$\mathbb{P}(\{j \in \{1, \ldots, p\}, \hat{w}_j \neq 0\} = \mathbf{J}) \underset{n\to+\infty}{\longrightarrow} 1.$$

The two previous propositions bring into play the dual norm $(\Omega_\mathbf{J}^c)^*$ that we cannot compute in closed form, but requires to solve an optimization problem as complex as the initial problem in

Equation (3). However, we can prove bounds similar to those obtained in Propositions 4 and 5 for the necessary and sufficient conditions.

5.1.1 COMPARISON WITH THE LASSO AND GROUP LASSO

For the ℓ_1-norm, our two bounds lead to the usual consistency conditions for the Lasso, that is, the quantity $\|\mathbf{Q}_{\mathbf{J}^c\mathbf{J}}\mathbf{Q}_{\mathbf{JJ}}^{-1}\operatorname{sign}(\mathbf{w}_\mathbf{J})\|_\infty$ must be less or strictly less than one. Similarly, when $\mathcal{G}$ defines a partition of $\{1,\dots,p\}$ and if all the weights equal one, our two bounds lead in turn to the consistency conditions for the group Lasso, that is, the quantity $\max_{G\in(\mathcal{G}_\mathbf{J})^c}\|\mathbf{Q}_{G\,\mathrm{Hull}(\mathbf{J})}\mathbf{Q}^{-1}_{\mathrm{Hull}(\mathbf{J})\mathrm{Hull}(\mathbf{J})}(\frac{\mathbf{w}_G}{\|\mathbf{w}_G\|_2})_{G\in\mathcal{G}_\mathbf{J}}\|_2$ must be less or strictly less than one.

5.2 High-Dimensional Analysis

We prove a high-dimensional variable consistency result (see proof in Appendix G) that extends the corresponding result for the Lasso (Zhao and Yu, 2006; Wainwright, 2009), by assuming that the consistency condition in Theorem 6 is satisfied.

Theorem 7 *Assume that Q has unit diagonal, $\kappa = \lambda_{\min}(Q_{\mathbf{JJ}}) > 0$ and $(\Omega^c_\mathbf{J})^*[Q_{\mathbf{J}^c\mathbf{J}}Q^{-1}_{\mathbf{JJ}}\mathbf{r}_\mathbf{J}] < 1-\tau$, with $\tau > 0$. If $\tau\mu\sqrt{n} \geqslant \sigma C_3(\mathcal{G},\mathbf{J})$, and $\mu|\mathbf{J}|^{1/2} \leqslant C_4(\mathcal{G},\mathbf{J})$, then the probability of incorrect hull selection is upper bounded by:*

$$\exp\left(-\frac{n\mu^2\tau^2 C_1(\mathcal{G},\mathbf{J})}{2\sigma^2}\right) + 2|\mathbf{J}|\exp\left(-\frac{nC_2(\mathcal{G},\mathbf{J})}{2|\mathbf{J}|\sigma^2}\right),$$

where $C_1(\mathcal{G},\mathbf{J})$, $C_2(\mathcal{G},\mathbf{J})$, $C_3(\mathcal{G},\mathbf{J})$ and $C_4(\mathcal{G},\mathbf{J})$ are constants defined in Appendix G, which essentially depend on the groups, the smallest nonzero coefficient of $\mathbf{w}$ and how close the support $\{j\in\mathbf{J} : \mathbf{w}_j \neq 0\}$ of $\mathbf{w}$ is to its hull $\mathbf{J}$, that is the relevance of the prior information encoded by $\mathcal{G}$.

In the Lasso case, we have $C_1(\mathcal{G},\mathbf{J}) = O(1), C_2(\mathcal{G},\mathbf{J}) = O(|\mathbf{J}|^{-2}), C_3(\mathcal{G},\mathbf{J}) = O((\log p)^{1/2})$ and $C_4(\mathcal{G},\mathbf{J}) = O(|\mathbf{J}|^{-1})$, leading to the usual scaling $n \approx \log p$ and $\mu \approx \sigma(\log p/n)^{1/2}$.

We can also give the scaling of these constants in simple settings where groups overlap. For instance, let us consider that the variables are organized in a sequence (see Figure 4). Let us further assume that the weights $(d^G)_{G\in\mathcal{G}}$ satisfy the following two properties:

a) The weights take into account the overlaps, that is,

$$d^G_j = \beta(|\{H\in\mathcal{G}\,; H\ni j,\ H\subset G \text{ and } H\neq G\}|),$$

with $t\mapsto\beta(t)\in(0,1]$ a non-increasing function such that $\beta(0)=1$,

b) The term

$$\max_{j\in\{1,\dots,p\}}\sum_{G\ni j, G\in\mathcal{G}} d^G_j$$

is upper bounded by a constant $\mathcal{K}$ independent of p.

Note that we consider such weights in the experiments (see Section 6). Based on these assumptions, some algebra directly leads to

$$\|u\|_1 \leqslant \Omega(u) \leqslant 2\mathcal{K}\|u\|_1, \text{ for all } u\in\mathbb{R}^p.$$

We thus obtain a scaling similar to the Lasso (with, *in addition*, a control of the allowed nonzero patterns). With stronger assumptions on the possible positions of $\mathbf{J}$, we may have better scalings, but these are problem-dependent (a careful analysis of the group-dependent constants would still be needed in all cases).

6. Experiments

In this section, we carry out several experiments to illustrate the behavior of the sparsity-inducing norm Ω. We denote by *Structured-lasso*, or simply *Slasso*, the models regularized by the norm Ω. In addition, the procedure (introduced in Section 4.3) that consists in intersecting the nonzero patterns obtained for different models of Slasso will be referred to as *Intersected Structured-lasso*, or simply *ISlasso*.

Throughout the experiments, we consider noisy linear models $Y = X\mathbf{w} + \varepsilon$, where $\mathbf{w} \in \mathbb{R}^p$ is the generating loading vector and ε is a standard Gaussian noise vector with its variance set to satisfy $\|X\mathbf{w}\|_2 = 3\,\|\varepsilon\|_2$. This consequently leads to a fixed signal-to-noise ratio. We assume that the vector $\mathbf{w}$ is sparse, that is, it has only a few nonzero components, that is, $|\mathbf{J}| \ll p$. We further assume that these nonzero components are either organized on a sequence or on a two-dimensional grid (see Figure 9). Moreover, we consider sets of groups $\mathcal{G}$ such as those presented in Section 3.5. We also consider different choices for the weights $(d^G)_{G\in\mathcal{G}}$ that we denote by **(W1)**, **(W2)** and **(W3)** (we will keep this notation throughout the following experiments):

(W1): Uniform weights, $d_j^G = 1$,

(W2): Weights depending on the size of the groups, $d_j^G = |G|^{-2}$,

(W3): Weights for overlapping groups, $d_j^G = \rho^{|\{H\in\mathcal{G}; H\ni j,\ H\subset G \text{ and } H\neq G\}|}$, for some $\rho \in (0,1)$.

For each orientation in $\mathcal{G}$, the third type of weights **(W3)** aims at reducing the unbalance caused by the overlapping groups. Specifically, given a group $G \in \mathcal{G}$ and a variable $j \in G$, the corresponding weight d_j^G is all the more small as the variable j already belongs to other groups with the same orientation. Unless otherwise specified, we use the third type of weights **(W3)** with $\rho = 0.5$. In the following experiments, the loadings $w_{\mathbf{J}}$, as well as the design matrices, are generated from a standard Gaussian distribution with identity covariance matrix. The positions of $\mathbf{J}$ are also random and are uniformly drawn.

6.1 Consistent Hull Estimation

We first illustrate Theorem 6 that establishes necessary and sufficient conditions for consistent hull estimation. To this end, we compute the probability of correct hull estimation when we consider diamond-shaped generating patterns of $|\mathbf{J}| = 24$ variables on a 20×20-dimensional grid (see Figure 9h). Specifically, we generate 500 covariance matrices $\mathbf{Q}$ distributed according to a Wishart distribution with δ degrees of freedom, where δ is uniformly drawn in $\{1, 2, \ldots, 10p\}$.[7] The diagonal terms of $\mathbf{Q}$ are then re-normalized to one. For each of these covariance matrices, we compute an

7. We have empirically observed that this choice of degrees of freedom enables to cover well the consistency transition regime around zero in Figure 10.

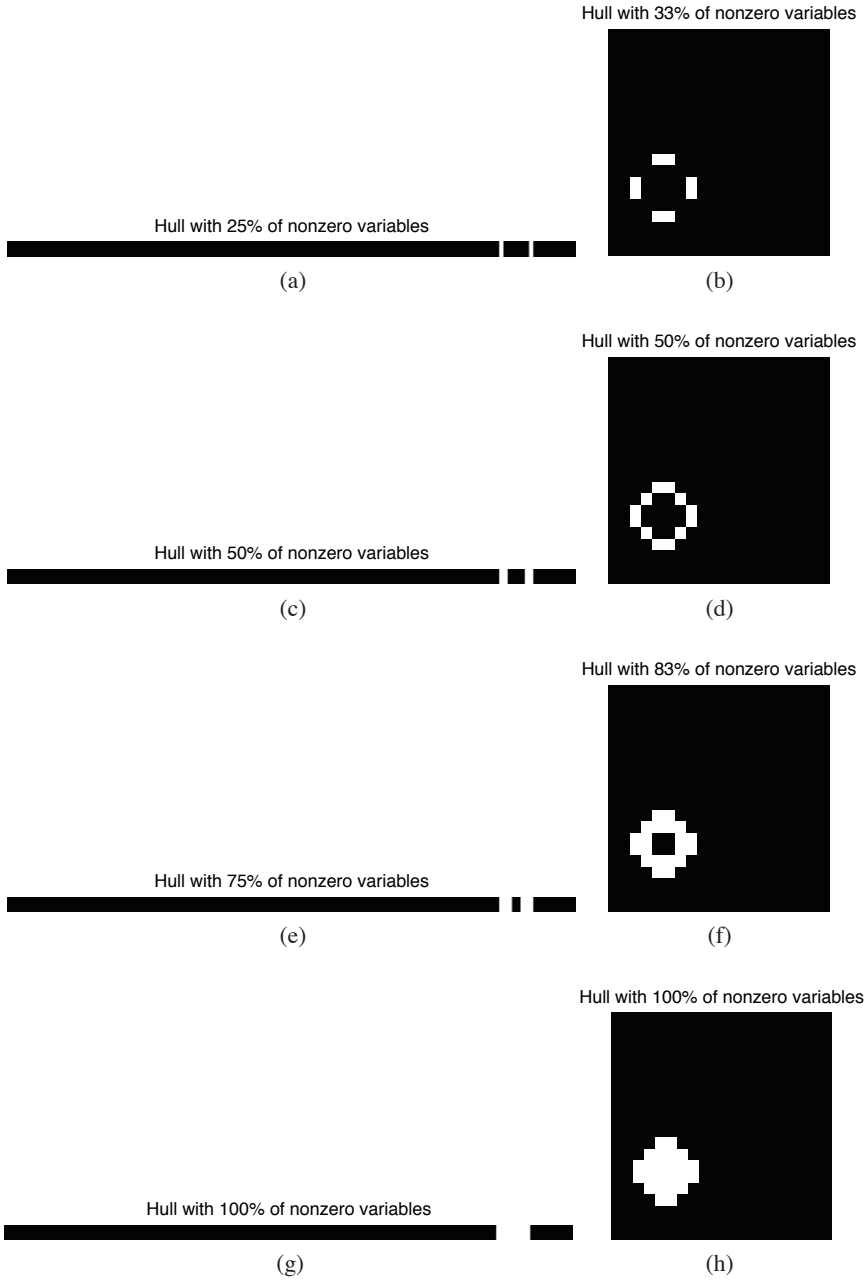


Figure 9: Examples of generating patterns (the zero variables are represented in black, while the nonzero ones are in white): (Left column, in white) generating patterns that are used for the experiments on 400-dimensional sequences; those patterns all form the same hull of 24 variables, that is, the contiguous sequence in (g). (Right column, in white) generating patterns that we use for the 20×20-dimensional grid experiments; again, those patterns all form the same hull of 24 variables, that is, the diamond-shaped convex in (h). The positions of these generating patterns are randomly selected during the experiments. For the grid setting, the hull is defined based on the set of groups that are half-planes, with orientations that are multiple of $\pi/4$ (see Section 3.5).

Figure 10: Consistent hull estimation: probability of correct hull estimation versus the consistency condition $(\Omega^c_{\mathbf{J}})^*[\mathbf{Q}_{\mathbf{J}^c\mathbf{J}}\mathbf{Q}_{\mathbf{JJ}}^{-1}\mathbf{r}_{\mathbf{J}}]$. The transition appears at zero, in good agreement with Theorem 6.

entire regularization path based on one realization of $\{\mathbf{J},\mathbf{w},X,\varepsilon\}$, with $n=3000$ samples. The quantities $\{\mathbf{J},\mathbf{w},\varepsilon\}$ are generated as described previously, while the n rows of X are Gaussian with covariance $\mathbf{Q}$. After repeating 20 times this computation for each $\mathbf{Q}$, we eventually report in Figure 10 the probabilities of correct hull estimation versus the consistency condition $(\Omega^c_{\mathbf{J}})^*[\mathbf{Q}_{\mathbf{J}^c\mathbf{J}}\mathbf{Q}_{\mathbf{JJ}}^{-1}\mathbf{r}_{\mathbf{J}}]$. In good agreement with Theorem 6, comparing $(\Omega^c_{\mathbf{J}})^*[\mathbf{Q}_{\mathbf{J}^c\mathbf{J}}\mathbf{Q}_{\mathbf{JJ}}^{-1}\mathbf{r}_{\mathbf{J}}]$ to 1 determines whether the hull is consistently estimated.

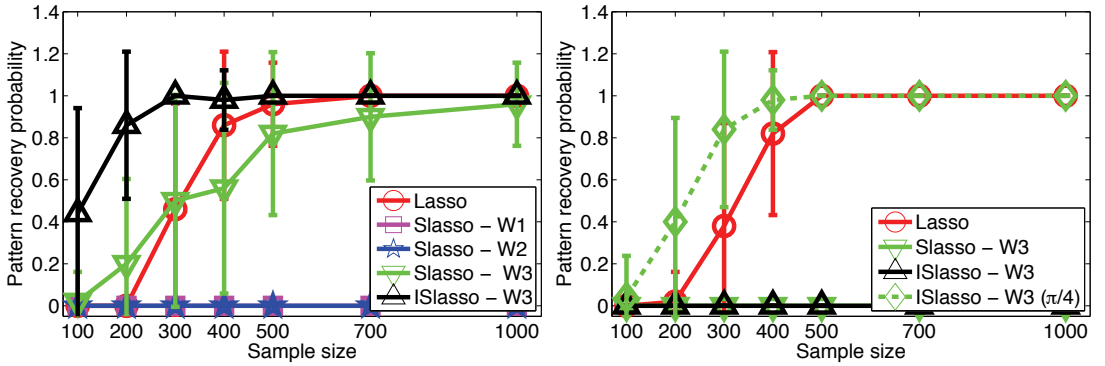
6.2 Structured Variable Selection

We show in this experiment that the prior information we put through the norm Ω improves upon the ability of the model to recover spatially structured nonzero patterns. We are looking at two situations where we can express such a prior through Ω, namely (1) the selection of a contiguous pattern on a sequence (see Figure 9g) and (2) the selection of a convex pattern on a grid (see Figure 9h).

In what follows, we consider $p=400$ variables with generating patterns $\mathbf{w}$ whose hulls are composed of $|\mathbf{J}|=24$ variables. For different sample sizes $n\in\{100,200,300,400,500,700,1000\}$, we consider the probabilities of correct recovery and the (normalized) Hamming distance to the true nonzero patterns. For the realization of a random setting $\{\mathbf{J},\mathbf{w},X,\varepsilon\}$, we compute an entire regularization path over which we collect the closest Hamming distance to the true nonzero pattern and whether it has been exactly recovered for some μ. After repeating 50 times this computation for each sample size n, we report the results in Figure 11.

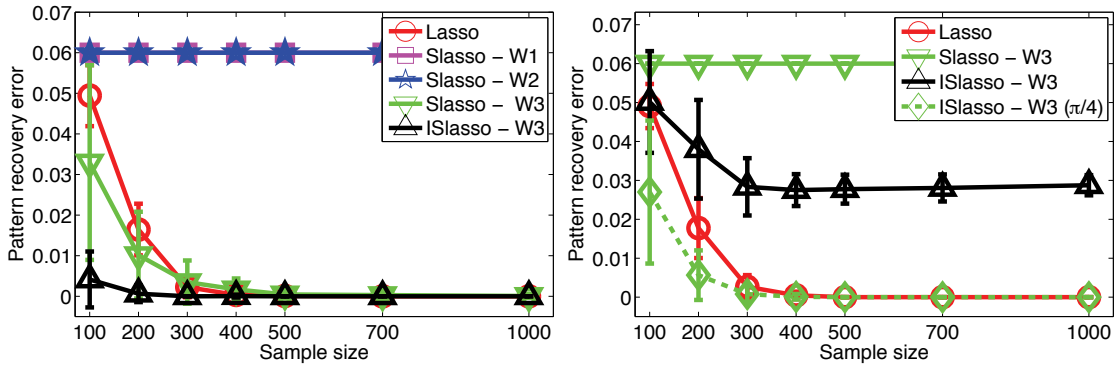
First and foremost, the simulations highlight how important the weights $(d^G)_{G\in\mathcal{G}}$ are. In particular, the uniform **(W1)** and size-dependent weights **(W2)** perform poorly since they do not take into account the overlapping groups. The models learned with such weights do not manage to recover the correct nonzero patterns (in that case, the best model found on the path corresponds to the empty solution, with a normalized Hamming distance of $|\mathbf{J}|/p=0.06$—see Figure 11c).

Although groups that moderately overlap do help (e.g., see Slasso with the weights **(W3)** on Figure 11c), it remains delicate to handle groups with many overlaps, even with an appropriate choice of $(d^G)_{G\in\mathcal{G}}$ (e.g., see Slasso on Figure 11d). The ISlasso procedure does not suffer from this issue since it reduces the number of overlaps whilst keeping the desirable effects of overlapping



(a) Probability of recovery for the sequence

(b) Probability of recovery for the grid



(c) Distance to the true pattern for the sequence

(d) Distance to the true pattern for the grid

Figure 11: For different sample sizes, the probabilities of correct recovery and the (normalized) Hamming distance to the true nonzero patterns are displayed. In the grid case, two sets of groups $\mathcal{G}$ are considered, the rectangular groups with or without the $\pm\pi/4$-groups (denoted by $(\pi/4)$ in the legend). The points and the error bars on the curves respectively represent the mean and the standard deviation, based on 50 random settings $\{\mathbf{J}, \mathbf{w}, X, \varepsilon\}$.

groups. Another way to yield a better level of sparsity, even with many overlaps, would be to consider non-convex alternatives to Ω (see, e.g., Jenatton et al., 2010b). Moreover, adding the $\pm\pi/4$-groups to the rectangular groups enables to recover a nonzero pattern closer to the generating pattern. This is illustrated on Figure 11d where the error of ISlasso with only rectangular groups (in black) corresponds to the selection of the smallest rectangular box around the generating pattern.

6.3 Prediction Error and Relevance of the Structured Prior

In the next simulation, we start from the same setting as Section 6.2 where we additionally evaluate the relevance of the contiguous (or convex) prior by varying the number of zero variables that are contained in the hull (see Figure 9). We then compute the prediction error for different sample sizes $n \in \{250, 500, 1000\}$. The prediction error is understood here as $\|X^{\text{test}}(\mathbf{w} - \hat{w})\|_2^2 / \|X^{\text{test}}\mathbf{w}\|_2^2$, where $\hat{w}$ denotes the OLS estimate, performed on the nonzero pattern found by the model considered (i.e., either Lasso, Slasso or ISlasso). The regularization parameter is chosen by 5-fold cross-validation and the test set consists of 500 samples. For each value of n, we display on Figure 12 the median errors over 50 random settings $\{\mathbf{J}, \mathbf{w}, X, \varepsilon\}$, for respectively the sequence and the grid. Note that we have dropped for clarity the models that performed already poorly in Section 6.2.

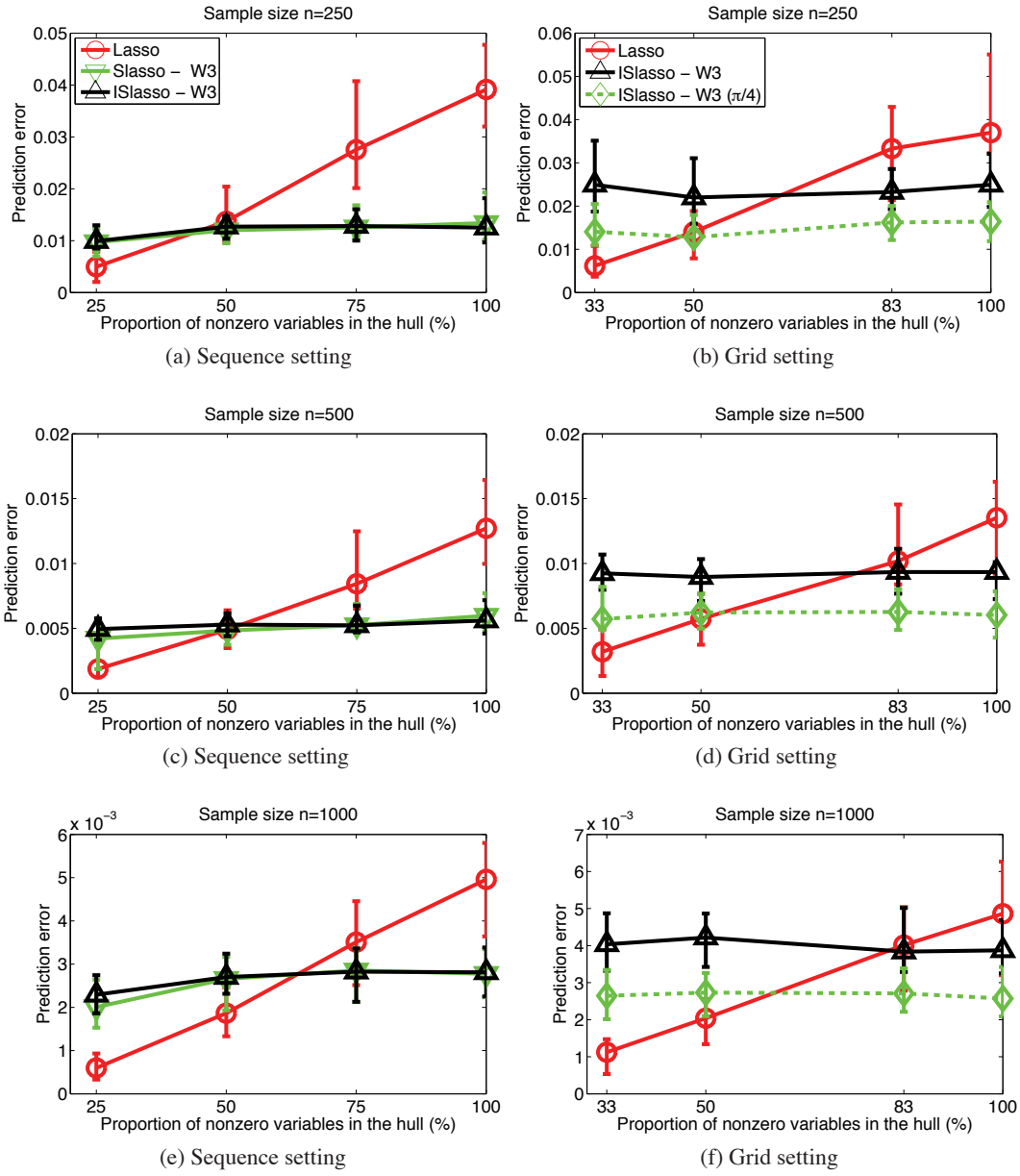
The experiments show that if the prior about the generating pattern is relevant, then our structured approach performs better that the standard Lasso. Indeed, as soon as the hull of the generating pattern does not contain too many zero variables, Slasso/ISlasso outperform Lasso. In fact, the sample complexity of the Lasso depends on the number of nonzero variables in $\mathbf{w}$ (Wainwright, 2009) as opposed to the size of the hull for Slasso/ISlasso. This also explains why the error for Slasso/ISlasso is almost constant with respect to the number of nonzero variables (since the hull has a constant size).

6.4 Active Set Algorithm

We finally focus on the active set algorithm (see Section 4) and compare its time complexity to the SOCP solver when we are looking for a sparse structured target. More precisely, for a fixed level of sparsity $|\mathbf{J}| = 24$ and a fixed number of observations $n = 3500$, we analyze the complexity with respect to the number of variables p that varies in $\{100, 225, 400, 900, 1600, 2500\}$. We consider the same experimental protocol as above except that we display the median CPU time based only[8] on 5 random settings $\{\mathbf{J}, \mathbf{w}, X, \varepsilon\}$. We assume that we have a rough idea of the level of sparsity of the true vector and we set the stopping criterion $s = 4|\mathbf{J}|$ (see Algorithm 3), which is a rather conservative choice. We show on Figure 13 that we considerably lower the computational cost for the same level of performance.[9] As predicted by the complexity analysis of the active set algorithm (see the end of Section 4), considering the set of rectangular groups with or without the $\pm\pi/4$-groups results in the same complexity (up to constant terms). We empirically obtain an average complexity of $\approx O(p^{2.13})$ for the SOCP solver and of $\approx O(p^{0.45})$ for the active set algorithm.

Not surprisingly, for small values of p, the SOCP solver is faster than the active set algorithm, since the latter has to check its optimality by computing necessary and sufficient conditions (see Algorithm 3 and the discussion in the algorithmic complexity paragraph of Section 4).

8. Note that it already corresponds to several hundreds of runs for both the SOCP and the active set algorithms since we compute a 5-fold cross-validation for each regularization parameter of the (approximate) regularization path.
9. We have not displayed this second figure that is just the superposition of the error curves for the SOCP and the active set algorithms.



(a) Sequence setting (b) Grid setting

(c) Sequence setting (d) Grid setting

(e) Sequence setting (f) Grid setting

Figure 12: For the sample size $n \in \{250, 500, 1000\}$, we plot the prediction error versus the proportion of nonzero variables in the hull of the generating pattern. In the grid case, two sets of groups $\mathcal{G}$ are considered, the rectangular groups with or without the $\pm\pi/4$-groups (denoted by $(\pi/4)$ in the legend). The points, the lower and upper error bars on the curves respectively represent the median, the first and third quartile, based on 50 random settings $\{\mathbf{J}, \mathbf{w}, X, \varepsilon\}$.

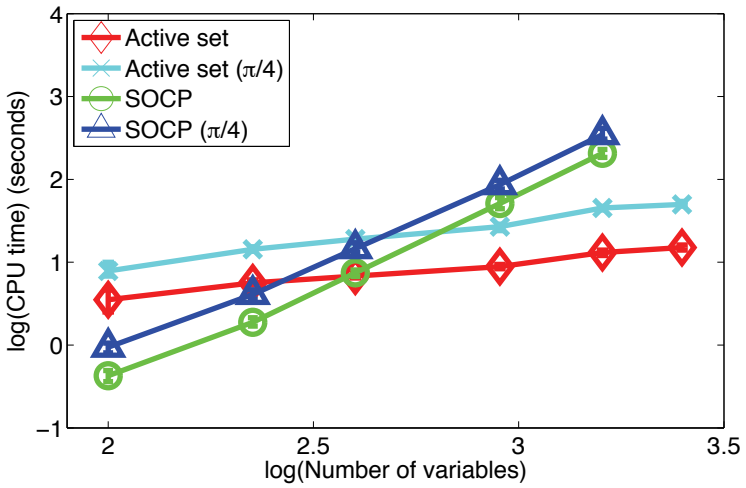


Figure 13: Computational benefit of the active set algorithm: CPU time (in seconds) versus the number of variables p, displayed in log-log scale. The points, the lower and upper error bars on the curves respectively represent the median, the first and third quartile. Two sets of groups $\mathcal{G}$ are considered, the rectangular groups with or without the $\pm\pi/4$-groups (denoted by $(\pi/4)$ in the legend). Due to the computational burden, we could not obtain the SOCP's results for $p = 2500$.

7. Conclusion

We have shown how to incorporate prior knowledge on the form of nonzero patterns for linear supervised learning. Our solution relies on a regularizing term which linearly combines ℓ_2-norms of possibly overlapping groups of variables. Our framework brings into play intersection-closed families of nonzero patterns, such as all rectangles on a two-dimensional grid. We have studied the design of these groups, efficient algorithms and theoretical guarantees of the structured sparsity-inducing method. Our experiments have shown to which extent our model leads to better prediction, depending on the relevance of the prior information.

A natural extension to this work is to consider bootstrapping since this may improve theoretical guarantees and result in better variable selection (Bach, 2008c; Meinshausen and Bühlmann, 2010). In order to deal with broader families of (non)zero patterns, it would be interesting to combine our approach with recent work on structured sparsity: for instance, Baraniuk et al. (2010) and Jacob et al. (2009) consider union-closed collections of nonzero patterns, He and Carin (2009) exploit structure through a Bayesian prior while Huang et al. (2009) handle non-convex penalties based on information-theoretic criteria.

More generally, our regularization scheme could also be used for various learning tasks, as soon as prior knowledge on the structure of the sparse representation is available, for example, for multiple kernel learning (Micchelli and Pontil, 2006), multi-task learning (Argyriou et al., 2008; Obozinski et al., 2009; Kim and Xing, 2010) and sparse matrix factorization problems (Mairal et al., 2010a; Jenatton et al., 2010b, 2011b).

Finally, although we have mostly explored in this paper the algorithmic and theoretical issues related to these norms, this type of prior knowledge is of clear interest for the spatially and tem-

porally structured data typical in bioinformatics (Kim and Xing, 2010), computer vision (Jenatton et al., 2010b; Mairal et al., 2010b) and neuroscience applications (see, e.g., Varoquaux et al., 2010).

Acknowledgments

We would like to thank the anonymous reviewers for their constructive comments that improve the clarity and the overall quality of the manuscript. We also thank Julien Mairal and Guillaume Obozinski for insightful discussions. This work was supported in part by a grant from the Agence Nationale de la Recherche (MGA Project) and a grant from the European Research Council (SIERRA Project).

Appendix A. Proof of Proposition 1

We recall that $L(w) = \frac{1}{n}\sum_{i=1}^{n} \ell\big(y_i, w^\top x_i\big)$. Since $w \mapsto \Omega(w)$ is convex and goes to infinite when $\|w\|_2$ goes to infinite, and since L is lower bounded, by Weierstrass' theorem, the problem in Equation (2) admits at least one global solution.

•*First case: Q invertible*. The Hessian of L is

$$\frac{1}{n}\sum_{i=1}^{n} x_i x_i^\top \frac{\partial^2 \ell}{\partial y'^2}(y_i, w^\top x_i).$$

It is positive definite since Q is positive definite and $\min_{i\in\{1,\dots,n\}} \frac{\partial^2 \ell}{\partial y'^2}(y_i, w^\top x_i) > 0$. So L is strictly convex. Consequently the objective function $L + \mu\Omega$ is strictly convex, hence the uniqueness of its minimizer.

•*Second case: $\{1,\dots,p\}$ belongs to $\mathcal{G}$*. We prove the uniqueness by contradiction. Assume that the problem in Equation (2) admits two different solutions w and $\tilde{w}$. Then one of the two solutions is different from 0, say $w \neq 0$.

By convexity, it means that any point of the segment $[w,\tilde{w}] = \big\{aw + (1-a)\tilde{w};\ a \in [0,1]\big\}$ also minimizes the objective function $L + \mu\Omega$. Since both L and $\mu\Omega$ are convex functions, it means that they are both linear when restricted to $[w,\tilde{w}]$.

Now, $\mu\Omega$ is only linear on segments of the form $[v,tv]$ with $v \in \mathbb{R}^p$ and $t > 0$. So we necessarily have $\tilde{w} = tw$ for some positive t. We now show that L is strictly convex on $[w,tw]$, which will contradict that it is linear on $[w,\tilde{w}]$. Let $E = \mathrm{Span}(x_1,\dots,x_n)$ and $E^\perp$ be the orthogonal of E in $\mathbb{R}^p$. The vector w can be decomposed in $w = w' + w''$ with $w' \in E$ and $w'' \in E^\perp$. Note that we have $w' \neq 0$ (since if it was equal to 0, w'' would be the minimizer of $\mu\Omega$, which would imply $w'' = 0$ and contradict $w \neq 0$). We thus have $(w^\top x_1,\dots,w^\top x_n) = (w'^\top x_1,\dots,w'^\top x_n) \neq 0$.

This implies that the function $s \mapsto \ell(y_i, sw^\top x_i)$ is a polynomial of degree 2. So it is not linear. This contradicts the existence of two different solutions, and concludes the proof of uniqueness.

Remark 8 *Still by using that a sum of convex functions is constant on a segment if and only if the functions are linear on this segment, the proof can be extended in order to replace the alternative assumption "$\{1,\dots,p\}$ belongs to $\mathcal{G}$" by the weaker but more involved assumption: for any $(j,k) \in \{1,\dots,p\}^2$, there exists a group $G \in \mathcal{G}$ which contains both j and k.*

Appendix B. Proof of Theorem 2

For $w \in \mathbb{R}^p$, we denote by $Z(w)$ its zero pattern (i.e., the indices of zero-components of w). To prove the result, it suffices to prove that for any set $I \subset \{1,\dots,p\}$ with $I^c \notin \mathcal{Z}$ and $|I| \leq k-1$, the probability of

$$\mathcal{E}_I = \big\{Y \in \mathbb{R}^n\text{: there exists } w \text{ solution of the problem in Equation (2) with } Z(w) = I^c\big\}$$

is equal to 0. We will prove this by contradiction: assume that there exists a set $I \subset \{1,\dots,p\}$ with $I^c \notin \mathcal{Z}$, $|I| \leq k-1$ and $\mathbb{P}(\mathcal{E}_I) > 0$. Since $I^c \notin \mathcal{Z}$, there exists $\alpha \in \mathrm{Hull}(I) \setminus I$. Let $J = I \cup \{\alpha\}$ and $\mathcal{G}_I = \{G \in \mathcal{G} : G \cap I \neq \emptyset\}$ be the set of active groups. Define $\mathbb{R}^J = \{w \in \mathbb{R}^p : w_{J^c} = 0\}$. The restrictions $L_J : \mathbb{R}^J \to \mathbb{R}$ and $\Omega_J : \mathbb{R}^J \to \mathbb{R}$ of L and Ω are continuously differentiable functions on $\big\{w \in \mathbb{R}^J : w_I \neq 0\big\}$ with respective gradients

$$\nabla L_J(w) = \left(\frac{\partial L_J}{\partial w_j}(w)\right)_{j\in J}^\top \quad \text{and} \quad \nabla \Omega_J(w) = \left(w_j \Big(\sum_{\substack{G\in\mathcal{G}_I,\\ G\ni j}} (d_j^G)^2 \, \|d^G \circ w\|_2^{-1}\Big)\right)_{j\in J}^\top.$$

Let $f(w,Y) = \nabla L_J(w) + \mu \nabla \Omega_J(w)$, where the dependence in Y of $f(w,Y)$ is hidden in $\nabla L_J(w) = \frac{1}{n}\sum_{i=1}^n (x_i)_J \frac{\partial \ell}{\partial y'}(y_i, w^\top x_i)$.

For $Y \in \mathcal{E}_I$, there exists $w \in \mathbb{R}^J$ with $Z(w) = I^c$, which minimizes the convex function $L_J + \mu\Omega_J$. The vector w satisfies $f(w,Y) = 0$. So we have proved $\mathcal{E}_I \subset \mathcal{E}'_I$, where

$$\mathcal{E}'_I = \big\{Y \in \mathbb{R}^n : \text{ there exists } w \in \mathbb{R}^J \text{ with } Z(w) = I^c \text{ and } f(w,Y) = 0\big\}.$$

Let $\tilde{y} \in \mathcal{E}_I$. Consider the equation $f(w,\tilde{y}) = 0$ on $\big\{w \in \mathbb{R}^J : w_j \neq 0 \text{ for any } j \in I\big\}$. By construction, we have $|J| \leq k$, and thus, by assumption, the matrix $X^J = \big((x_1)_J, ..., (x_n)_J\big)^\top \in \mathbb{R}^{n\times|J|}$ has rank $|J|$. As in the proof of Proposition 1, this implies that the function L_J is strictly convex, and then, the uniqueness of the minimizer of $L_J + \mu\Omega$, and also the uniqueness of the point at which the gradient of this function vanishes. So the equation $f(w,\tilde{y}) = 0$ on $\big\{w \in \mathbb{R}^J : w_j \neq 0 \text{ for any } j \in I\big\}$ has a unique solution, which we will write $w^{\tilde{y}}$.

On a small enough ball around $(w_J^{\tilde{y}}, \tilde{y})$, f is continuously differentiable since none of the norms vanishes at $w_J^{\tilde{y}}$. Let $(f_j)_{j\in J}$ be the components of f and $H_{JJ} = \big(\frac{\partial f_j}{\partial w_k}\big)_{j\in J, k\in J}$. The matrix H_{JJ} is actually the sum of:

a) the Hessian of L_J, which is positive definite (still from the same argument as in the proof of Theorem 1),

b) the Hessian of the norm Ω_J around $(w_J^{\tilde{y}}, \tilde{y})$ that is positive semidefinite on this small ball according to the Hessian characterization of convexity (Borwein and Lewis, 2006, Theorem 3.1.11).

Consequently, H_{JJ} is invertible. We can now apply the implicit function theorem to obtain that for Y in a neighborhood of $\tilde{y}$,

$$w^Y = \psi(Y),$$

with $\psi = (\psi_j)_{j\in J}$ a continuously differentiable function satisfying the matricial relation

$$(\dots, \nabla\psi_j, \dots) H_{JJ} + (\dots, \nabla_y f_j, \dots) = 0.$$

Let C_α denote the column vector of H_{JJ}^{-1} corresponding to the index α, and let D the diagonal matrix whose (i,i)-th element is $\frac{\partial^2 \ell}{\partial y \partial y'}(y_i, w^\top x_i)$. Since $n(\ldots, \nabla_y f_j, \ldots) = DX^J$, we have

$$n\nabla\psi_\alpha = -DX^J C_\alpha.$$

Now, since X^J has full rank, $C_\alpha \neq 0$ and none of the diagonal elements of D is null (by assumption on ℓ), we have $\nabla\psi_\alpha \neq 0$. Without loss of generality, we may assume that $\partial\psi_\alpha/\partial y_1 \neq 0$ on a neighborhood of $\tilde{y}$.

We can apply again the implicit function theorem to show that on an open ball in $\mathbb{R}^n$ centered at $\tilde{y}$, say $\mathcal{B}_{\tilde{y}}$, the solution to $\psi_\alpha(Y) = 0$ can be written $y_1 = \varphi(y_2, \ldots, y_n)$ with φ a continuously differentiable function.

By Fubini's theorem and by using the fact that the Lebesgue measure of a singleton in $\mathbb{R}^n$ equals zero, we get that the set $A(\tilde{y}) = \left\{Y \in \mathcal{B}_{\tilde{y}} : \psi_\alpha(Y) = 0\right\}$ has thus zero probability. Let $\mathcal{S} \subset \mathcal{E}_I$ be a compact set. We thus have $\mathcal{S} \subset \mathcal{E}_I'$.

By compacity, the set $\mathcal{S}$ can be covered by a finite number of ball $\mathcal{B}_{\tilde{y}}$. So there exist $\tilde{y}_1, \ldots, \tilde{y}_m$ such that we have $\mathcal{S} \subset A(\tilde{y}_1) \cup \cdots \cup A(\tilde{y}_m)$. Consequently, we have $\mathbb{P}(\mathcal{S}) = 0$.

Since this holds for any compact set in $\mathcal{E}_I$ and since the Lebesgue measure is regular, we have $\mathbb{P}(\mathcal{E}_I) = 0$, which contradicts the definition of I, and concludes the proof.

Appendix C. Proof of the Minimality of the Backward Procedure (See Algorithm 1)

There are essentially two points to show: (1) $\mathcal{G}$ spans $\mathcal{Z}$, and (2) $\mathcal{G}$ is minimal.

The first point can be shown by a proof by recurrence on the depth of the DAG. At step t, the base $\mathcal{G}^{(t)}$ verifies $\{\bigcup_{G\in\mathcal{G}'} G,\ \forall \mathcal{G}' \subseteq \mathcal{G}^{(t)}\} = \{G \in \mathcal{Z}, |G| \leq t\}$ because an element $G \in \mathcal{Z}$ is either the union of itself or the union of elements strictly smaller. The initialization $t = \min_{G\in\mathcal{Z}} |G|$ is easily verified, the leafs of the DAG being necessarily in $\mathcal{G}$.

As for the second point, we proceed by contradiction. If there exists another base $\mathcal{G}^*$ that spans $\mathcal{Z}$ such that $\mathcal{G}^* \subset \mathcal{G}$, then

$$\exists\, e \in \mathcal{G},\ e \notin \mathcal{G}^*.$$

By definition of the set $\mathcal{Z}$, there exists in turn $\mathcal{G}' \subseteq \mathcal{G}^*,\ \mathcal{G}' \neq \{e\}$ (otherwise, e would belong to $\mathcal{G}^*$), verifying $e = \bigcup_{G\in\mathcal{G}'} G$, which is impossible by construction of $\mathcal{G}$ whose members cannot be the union of elements of $\mathcal{Z}$.

Appendix D. Proof of Proposition 3

The proposition comes from a classic result of Fenchel Duality (Borwein and Lewis, 2006, Theorem 3.3.5 and Exercise 3.3.9) when we consider the convex function

$$h_J : w_J \mapsto \frac{\lambda}{2}\left[\Omega_J(w_J)\right]^2,$$

whose Fenchel conjugate h_J^* is given by $\kappa_J \mapsto \frac{1}{2\lambda}\left[\Omega_J^*(\kappa_J)\right]^2$ (Boyd and Vandenberghe, 2004, example 3.27). Since the set

$$\{w_J \in \mathbb{R}^{|J|};\ h_J(w_J) < \infty\} \cap \{w_J \in \mathbb{R}^{|J|};\ L_J(w_J) < \infty \text{ and } L_J \text{ is continuous at } w_J\}$$

is not empty, we get the first part of the proposition. Moreover, the primal-dual variables $\{w_J, \kappa_J\}$ is optimal if and only if

$$\begin{cases} -\kappa_J & \in \quad \partial L_J(w_J), \\ \kappa_J & \in \quad \partial[\frac{\lambda}{2}[\Omega_J(w_J)]^2] = \lambda\Omega_J(w_J)\partial\Omega_J(w_J), \end{cases}$$

where $\partial\Omega_J(w_J)$ denotes the subdifferential of Ω_J at w_J. The differentiability of L_J at w_J then gives $\partial L_J(w_J) = \{\nabla L_J(w_J)\}$. It now remains to show that

$$\kappa_J \in \lambda\Omega_J(w_J)\partial\Omega_J(w_J) \tag{5}$$

is equivalent to

$$w_J^\top \kappa_J = \frac{1}{\lambda}[\Omega_J^*(\kappa_J)]^2 = \lambda[\Omega_J(w_J)]^2. \tag{6}$$

As a starting point, the Fenchel-Young inequality (Borwein and Lewis, 2006, Proposition 3.3.4) gives the equivalence between Equation (5) and

$$\frac{\lambda}{2}[\Omega_J(w_J)]^2 + \frac{1}{2\lambda}[\Omega_J^*(\kappa_J)]^2 = w_J^\top \kappa_J. \tag{7}$$

In addition, we have (Rockafellar, 1970)

$$\partial\Omega_J(w_J) = \{u_J \in \mathbb{R}^{|J|}; u_J^\top w_J = \Omega_J(w_J) \text{ and } \Omega_J^*(u_J) \leq 1\}. \tag{8}$$

Thus, if $\kappa_J \in \lambda\Omega_J(w_J)\partial\Omega_J(w_J)$ then $w_J^\top \kappa_J = \lambda[\Omega_J(w_J)]^2$. Combined with Equation (7), we obtain $w_J^\top \kappa_J = \frac{1}{\lambda}[\Omega_J^*(\kappa_J)]^2$.

Reciprocally, starting from Equation (6), we notably have

$$w_J^\top \kappa_J = \lambda[\Omega_J(w_J)]^2.$$

In light of Equation (8), it suffices to check that $\Omega_J^*(\kappa_J) \leq \lambda\Omega_J(w_J)$ in order to have Equation (5). Combining Equation (6) with the definition of the dual norm, it comes

$$\frac{1}{\lambda}[\Omega_J^*(\kappa_J)]^2 = w_J^\top \kappa_J \leq \Omega_J^*(\kappa_J)\Omega_J(w_J),$$

which concludes the proof of the equivalence between Equation (5) and Equation (6).

Appendix E. Proofs of Propositions 4 and 5

In order to check that the reduced solution w_J is optimal for the full problem in Equation (3), we complete with zeros on J^c to define w, compute $\kappa = -\nabla L(w)$, which is such that $\kappa_J = -\nabla L_J(w_J)$, and get a duality gap for the full problem equal to

$$\frac{1}{2\lambda}\left([\Omega^*(\kappa)]^2 - \lambda w_J^\top \kappa_J\right).$$

By designing upper and lower bounds for $\Omega^*(\kappa)$, we get sufficient and necessary conditions.

E.1 Proof of Proposition 4

Let us suppose that $w^* = \begin{pmatrix} w_J^* \\ 0_{J^c} \end{pmatrix}$ is optimal for the full problem in Equation (3). Following the same derivation as in Lemma 14 (up to the squaring of the regularization Ω), we have that w^* is a solution of Equation (3) if and only if for all $u \in \mathbb{R}^p$,

$$u^\top \nabla L(w^*) + \lambda \Omega(w^*)(u_J^\top r_J + (\Omega_J^c)[u_{J^c}]) \geq 0,$$

with

$$r = \sum_{G \in \mathcal{G}_J} \frac{d^G \circ d^G \circ w^*}{\|d^G \circ w^*\|_2}.$$

We project the optimality condition onto the variables that can possibly enter the active set, that is, the variables in $\Pi_{\mathcal{P}}(J)$. Thus, for each $K \in \Pi_{\mathcal{P}}(J)$, we have for all $u_{K\backslash J} \in \mathbb{R}^{|K\backslash J|}$,

$$u_{K\backslash J}^\top \nabla L(w^*)_{K\backslash J} + \lambda \Omega(w^*) \sum_{G \in \mathcal{G}_{K\backslash J} \cap (\mathcal{G}_J)^c} \left\| d^G_{K\backslash J} \circ u_{G \cap K\backslash J} \right\|_2 \geq 0.$$

By combining Lemma 13 and the fact that $\mathcal{G}_{K\backslash J} \cap (\mathcal{G}_J)^c = \mathcal{G}_K \backslash \mathcal{G}_J$, we have for all $G \in \mathcal{G}_K \backslash \mathcal{G}_J$, $K\backslash J \subseteq G$ and therefore $u_{G \cap K\backslash J} = u_{K\backslash J}$. Since we cannot compute the dual norm of $u_{K\backslash J} \mapsto \|d^G_{K\backslash J} \circ u_{K\backslash J}\|_2$ in closed-form, we instead use the following upperbound

$$\left\| d^G_{K\backslash J} \circ u_{K\backslash J} \right\|_2 \leq \|d^G_{K\backslash J}\|_\infty \left\| u_{K\backslash J} \right\|_2,$$

so that we get for all $u_{K\backslash J} \in \mathbb{R}^{|K\backslash J|}$,

$$u_{K\backslash J}^\top \nabla L(w^*)_{K\backslash J} + \lambda \Omega(w^*) \sum_{G \in \mathcal{G}_K \backslash \mathcal{G}_J} \|d^G_{K\backslash J}\|_\infty \left\| u_{K\backslash J} \right\|_2 \geq 0.$$

Finally, Proposition 3 gives $\lambda \Omega(w^*) = \left\{ -\lambda {w^*}^\top \nabla L(w^*) \right\}^{\frac{1}{2}}$, which leads to the desired result.

E.2 Proof of Proposition 5

The goal of the proof is to upper bound the dual norm $\Omega^*(\kappa)$ by taking advantage of the structure of $\mathcal{G}$; we first show how we can upper bound $\Omega^*(\kappa)$ by $(\Omega_J^c)^*[\kappa_{J^c}]$. We indeed have:

$$\begin{aligned}
\Omega^*(\kappa) &= \max_{\sum_{G \in \mathcal{G}_J} \|d^G \circ v\|_2 + \sum_{G \in (\mathcal{G}_J)^c} \|d^G \circ v\|_2 \leq 1} v^\top \kappa \\
&\leq \max_{\sum_{G \in \mathcal{G}_J} \left\|d^G_J \circ v_J\right\|_2 + \sum_{G \in (\mathcal{G}_J)^c} \|d^G \circ v\|_2 \leq 1} v^\top \kappa \\
&= \max_{\Omega_J(v_J) + (\Omega_J^c)(v_{J^c}) \leq 1} v^\top \kappa \\
&= \max\left\{ \Omega_J^*(\kappa_J), (\Omega_J^c)^*[\kappa_{J^c}] \right\},
\end{aligned}$$

where in the last line, we use Lemma 15. Thus the duality gap is less than

$$\frac{1}{2\lambda}\left([\Omega^*(\kappa)]^2 - [\Omega_J^*(\kappa_J)]^2 \right) \leq \frac{1}{2\lambda} \max\{0, [(\Omega_J^c)^*[\kappa_{J^c}]]^2 - [\Omega_J^*(\kappa_J)]^2\},$$

and a sufficient condition for the duality gap to be smaller than ε is

$$(\Omega_J^c)^*[\kappa_{J^c}] \leq (2\lambda\varepsilon + [\Omega_J^*(\kappa_J)]^2)^{\frac{1}{2}}.$$

Using Proposition 3, we have $-\lambda w^\top \nabla L(w) = [\Omega_J^*(\kappa_J)]^2$ and we get the right-hand side of Proposition 5. It now remains to upper bound $(\Omega_J^c)^*[\kappa_{J^c}]$. To this end, we call upon Lemma 11 to obtain:

$$(\Omega_J^c)^*[\kappa_{J^c}] \leq \max_{G \in (\mathcal{G}_J)^c} \left\{ \sum_{j \in G} \left\{ \frac{\kappa_j}{\sum_{H \in j, H \in (\mathcal{G}_J)^c} d_j^H} \right\}^2 \right\}^{\frac{1}{2}}.$$

Among all groups $G \in (\mathcal{G}_J)^c$, the ones with the maximum values are the largest ones, that is, those in the fringe groups $\mathcal{F}_J = \{G \in (\mathcal{G}_J)^c \,;\, \nexists G' \in (\mathcal{G}_J)^c, G \subseteq G'\}$. This argument leads to the result of Proposition 5.

Appendix F. Proof of Theorem 6

Necessary condition: We mostly follow the proof of Zou (2006) and Bach (2008b). Let $\hat{w} \in \mathbb{R}^p$ be the unique solution of

$$\min_{w \in \mathbb{R}^p} L(w) + \mu\,\Omega(w) = \min_{w \in \mathbb{R}^p} F(w).$$

The quantity $\hat{\Delta} = (\hat{w} - \mathbf{w})/\mu$ is the minimizer of $\tilde{F}$ defined as

$$\tilde{F}(\Delta) = \frac{1}{2}\Delta^\top Q \Delta - \mu^{-1} q^\top \Delta + \mu^{-1}\left[\Omega(\mathbf{w} + \mu\Delta) - \Omega(\mathbf{w})\right],$$

where $q = \frac{1}{n}\sum_{i=1}^n \varepsilon_i x_i$. The random variable $\mu^{-1} q^\top \Delta$ is a centered Gaussian with variance $\sqrt{\Delta^\top Q \Delta}/(n\mu^2)$. Since $Q \to \mathbf{Q}$, we obtain that for all $\Delta \in \mathbb{R}^p$,

$$\mu^{-1} q^\top \Delta = o_p(1).$$

Since $\mu \to 0$, we also have by taking the directional derivative of Ω at $\mathbf{w}$ in the direction of Δ

$$\mu^{-1}\left[\Omega(\mathbf{w} + \mu\Delta) - \Omega(\mathbf{w})\right] = \mathbf{r}_\mathbf{J}^\top \Delta_\mathbf{J} + \Omega_\mathbf{J}^c(\Delta_{\mathbf{J}^c}) + o(1),$$

so that for all $\Delta \in \mathbb{R}^p$

$$\tilde{F}(\Delta) = \Delta^\top \mathbf{Q}\Delta + \mathbf{r}_\mathbf{J}^\top \Delta_\mathbf{J} + \Omega_\mathbf{J}^c(\Delta_{\mathbf{J}^c}) + o_p(1) = \tilde{F}_{\lim}(\Delta) + o_p(1).$$

The limiting function $\tilde{F}_{\lim}$ being stricly convex (because $\mathbf{Q} \succ 0$) and $\tilde{F}$ being convex, we have that the minimizer $\hat{\Delta}$ of $\tilde{F}$ tends in probability to the unique minimizer of $\tilde{F}_{\lim}$ (Fu and Knight, 2000) referred to as Δ^*.

By assumption, with probability tending to one, we have $\mathbf{J} = \{j \in \{1,\dots,p\}, \hat{w}_j \neq 0\}$, hence for any $j \in \mathbf{J}^c$ $\mu\hat{\Delta}_j = (\hat{w} - \mathbf{w})_j = 0$. This implies that the nonrandom vector Δ^* verifies $\Delta^*_{\mathbf{J}^c} = 0$.

As a consequence, $\Delta^*_\mathbf{J}$ minimizes $\Delta_\mathbf{J}^\top \mathbf{Q}_{\mathbf{JJ}} \Delta_\mathbf{J} + \mathbf{r}_\mathbf{J}^\top \Delta_\mathbf{J}$, hence $\mathbf{r}_\mathbf{J} = -\mathbf{Q}_{\mathbf{JJ}} \Delta^*_\mathbf{J}$. Besides, since Δ^* is the minimizer of $\tilde{F}_{\lim}$, by taking the directional derivatives as in the proof of Lemma 14, we have

$$(\Omega_\mathbf{J}^c)^*[\mathbf{Q}_{\mathbf{J}^c\mathbf{J}} \Delta^*_\mathbf{J}] \leq 1.$$

This gives the necessary condition.

Sufficient condition: We turn to the sufficient condition. We first consider the problem reduced to the hull $\mathbf{J}$,

$$\min_{w\in\mathbb{R}^{|\mathbf{J}|}} L_{\mathbf{J}}(w_{\mathbf{J}}) + \mu\Omega_{\mathbf{J}}(w_{\mathbf{J}}).$$

that is strongly convex since $Q_{\mathbf{JJ}}$ is positive definite and thus admits a unique solution $\hat{w}_{\mathbf{J}}$. With similar arguments as the ones used in the necessary condition, we can show that $\hat{w}_{\mathbf{J}}$ tends in probability to the true vector $\mathbf{w}_{\mathbf{J}}$. We now consider the vector $\hat{w}\in\mathbb{R}^p$ which is the vector $\hat{w}_{\mathbf{J}}$ padded with zeros on $\mathbf{J}^c$. Since, from Theorem 2, we almost surely have $\mathrm{Hull}(\{j\in\{1,\dots,p\},\hat{w}_j\neq 0\}) = \{j\in\{1,\dots,p\},\hat{w}_j\neq 0\}$, we have already that the vector $\hat{w}$ consistently estimates the hull of $\mathbf{w}$ and we have that $\hat{w}$ tends in probability to $\mathbf{w}$. From now on, we consider that $\hat{w}$ is sufficiently close to $\mathbf{w}$, so that for any $G\in\mathcal{G}_{\mathbf{J}}$, $\|d^G\circ\hat{w}\|_2\neq 0$. We may thus introduce

$$\hat{r} = \sum_{G\in\mathcal{G}_{\mathbf{J}}} \frac{d^G\circ d^G\circ\hat{w}}{\|d^G\circ\hat{w}\|_2}.$$

It remains to show that $\hat{w}$ is indeed optimal for the full problem (that admits a unique solution due to the positiveness of Q). By construction, the optimality condition (see Lemma 14) relative to the active variables $\mathbf{J}$ is already verified. More precisely, we have

$$\nabla L(\hat{w})_{\mathbf{J}} + \mu\hat{r}_{\mathbf{J}} = Q_{\mathbf{JJ}}(\hat{w}_{\mathbf{J}} - \mathbf{w}_{\mathbf{J}}) - q_{\mathbf{J}} + \mu\hat{r}_{\mathbf{J}} = 0.$$

Moreover, for all $u_{\mathbf{J}^c}\in\mathbb{R}^{|\mathbf{J}^c|}$, by using the previous expression and the invertibily of Q, we have

$$u_{\mathbf{J}^c}^\top\nabla L(\hat{w})_{\mathbf{J}^c} = u_{\mathbf{J}^c}^\top\left\{-\mu Q_{\mathbf{J}^c\mathbf{J}}Q_{\mathbf{JJ}}^{-1}\hat{r}_{\mathbf{J}} + Q_{\mathbf{J}^c\mathbf{J}}Q_{\mathbf{JJ}}^{-1}q_{\mathbf{J}} - q_{\mathbf{J}^c}\right\}.$$

The terms related to the noise vanish, having actually $q = o_p(1)$. Since $Q\to\mathbf{Q}$ and $\hat{r}_{\mathbf{J}}\to\mathbf{r}_{\mathbf{J}}$, we get for all $u_{\mathbf{J}^c}\in\mathbb{R}^{|\mathbf{J}^c|}$

$$u_{\mathbf{J}^c}^\top\nabla L(\hat{w})_{\mathbf{J}^c} = -\mu u_{\mathbf{J}^c}^\top\left\{\mathbf{Q}_{\mathbf{J}^c\mathbf{J}}\mathbf{Q}_{\mathbf{JJ}}^{-1}\mathbf{r}_{\mathbf{J}}\right\} + o_p(\mu).$$

Since we assume $(\Omega_{\mathbf{J}}^c)^*[\mathbf{Q}_{\mathbf{J}^c\mathbf{J}}\mathbf{Q}_{\mathbf{JJ}}^{-1}\mathbf{r}_{\mathbf{J}}] < 1$, we obtain

$$-u_{\mathbf{J}^c}^\top\nabla L(\hat{w})_{\mathbf{J}^c} < \mu(\Omega_{\mathbf{J}}^c)[u_{\mathbf{J}^c}] + o_p(\mu),$$

which proves the optimality condition of Lemma 14 relative to the inactive variables: $\hat{w}$ is therefore optimal for the full problem.

Appendix G. Proof of Theorem 7

Since our analysis takes place in a finite-dimensional space, all the norms defined on this space are equivalent. Therefore, we introduce the equivalence parameters $a(\mathbf{J}), A(\mathbf{J}) > 0$ such that

$$\forall u\in\mathbb{R}^{|\mathbf{J}|},\ a(\mathbf{J})\|u\|_1 \le \Omega_{\mathbf{J}}[u] \le A(\mathbf{J})\|u\|_1.$$

We similarly define $a(\mathbf{J}^c), A(\mathbf{J}^c) > 0$ for the norm $(\Omega_{\mathbf{J}}^c)$ on $\mathbb{R}^{|\mathbf{J}^c|}$. In addition, we immediately get by order-reversing:

$$\forall u\in\mathbb{R}^{|\mathbf{J}|},\ A(\mathbf{J})^{-1}\|u\|_\infty \le (\Omega_{\mathbf{J}})^*[u] \le a(\mathbf{J})^{-1}\|u\|_\infty.$$

For any matrix Γ, we also introduce the operator norm $\|\Gamma\|_{m,s}$ defined as

$$\|\Gamma\|_{m,s} = \sup_{\|u\|_s \leq 1} \|\Gamma u\|_m.$$

Moreover, our proof will rely on the control of the *expected dual norm for isonormal vectors*: $\mathbb{E}\left[(\Omega_{\mathbf{J}}^c)^*(W)\right]$ with W a centered Gaussian random variable with unit covariance matrix. In the case of the Lasso, it is of order $(\log p)^{1/2}$.

Following Bach (2008b) and Nardi and Rinaldo (2008), we consider the reduced problem on $\mathbf{J}$,

$$\min_{w \in \mathbb{R}^p} L_{\mathbf{J}}(w_{\mathbf{J}}) + \mu \Omega_{\mathbf{J}}(w_{\mathbf{J}})$$

with solution $\hat{w}_{\mathbf{J}}$, which can be extended to $\mathbf{J}^c$ with zeros. From optimality conditions (see Lemma 14), we know that

$$\Omega_{\mathbf{J}}^*[Q_{\mathbf{JJ}}(\hat{w}_{\mathbf{J}} - \mathbf{w}_{\mathbf{J}}) - q_{\mathbf{J}}] \leq \mu, \tag{9}$$

where the vector $q \in \mathbb{R}^p$ is defined as $q = \frac{1}{n}\sum_{i=1}^n \varepsilon_i x_i$. We denote by $\nu = \min\{|\mathbf{w}_j|;\ \mathbf{w}_j \neq 0\}$ the smallest nonzero components of $\mathbf{w}$. We first prove that we must have with high probability $\|\hat{w}_G\|_\infty > 0$ for all $G \in \mathcal{G}_{\mathbf{J}}$, proving that the hull of the active set of $\hat{w}_{\mathbf{J}}$ is exactly $\mathbf{J}$ (i.e., no active group is missing).

We have

$$\begin{aligned}\|\hat{w}_{\mathbf{J}} - \mathbf{w}_{\mathbf{J}}\|_\infty &\leq \|Q_{\mathbf{JJ}}^{-1}\|_{\infty,\infty} \|Q_{\mathbf{JJ}}(\hat{w}_{\mathbf{J}} - \mathbf{w}_{\mathbf{J}})\|_\infty \\ &\leq |\mathbf{J}|^{1/2}\kappa^{-1}\left(\|Q_{\mathbf{JJ}}(\hat{w}_{\mathbf{J}} - \mathbf{w}_{\mathbf{J}}) - q_{\mathbf{J}}\|_\infty + \|q_{\mathbf{J}}\|_\infty\right),\end{aligned}$$

hence from (9) and the definition of $A(\mathbf{J})$,

$$\|\hat{w}_{\mathbf{J}} - \mathbf{w}_{\mathbf{J}}\|_\infty \leq |\mathbf{J}|^{1/2}\kappa^{-1}\left(\mu A(\mathbf{J}) + \|q_{\mathbf{J}}\|_\infty\right). \tag{10}$$

Thus, if we assume $\mu \leq \frac{\kappa\nu}{3|\mathbf{J}|^{1/2}A(\mathbf{J})}$ and

$$\|q_{\mathbf{J}}\|_\infty \leq \frac{\kappa\nu}{3|\mathbf{J}|^{1/2}}, \tag{11}$$

we get

$$\|\hat{w}_{\mathbf{J}} - \mathbf{w}_{\mathbf{J}}\|_\infty \leq 2\nu/3, \tag{12}$$

so that for all $G \in \mathcal{G}_{\mathbf{J}}$, $\|\hat{w}_G\|_\infty \geq \frac{\nu}{3}$, hence the hull is indeed selected.

This also ensures that $\hat{w}_{\mathbf{J}}$ satisfies the equation (see Lemma 14)

$$Q_{\mathbf{JJ}}(\hat{w}_{\mathbf{J}} - \mathbf{w}_{\mathbf{J}}) - q_{\mathbf{J}} + \mu\hat{r}_{\mathbf{J}} = 0, \tag{13}$$

where

$$\hat{r} = \sum_{G \in \mathcal{G}_{\mathbf{J}}} \frac{d^G \circ d^G \circ \hat{w}}{\|d^G \circ \hat{w}\|_2}.$$

We now prove that the $\hat{w}$ padded with zeros on $\mathbf{J}^c$ is indeed optimal for the full problem with high probability. According to Lemma 14, since we have already proved (13), it suffices to show that

$$(\Omega_{\mathbf{J}}^c)^*[\nabla L(\hat{w})_{\mathbf{J}^c}] \leq \mu.$$

Defining $q_{\mathbf{J}^c|\mathbf{J}} = q_{\mathbf{J}^c} - Q_{\mathbf{J}^c\mathbf{J}}Q_{\mathbf{JJ}}^{-1}q_{\mathbf{J}}$, we can write the gradient of L on $\mathbf{J}^c$ as

$$\nabla L(\hat{w})_{\mathbf{J}^c} = -q_{\mathbf{J}^c|\mathbf{J}} - \mu Q_{\mathbf{J}^c\mathbf{J}}Q_{\mathbf{JJ}}^{-1}\hat{r}_{\mathbf{J}} = -q_{\mathbf{J}^c|\mathbf{J}} - \mu Q_{\mathbf{J}^c\mathbf{J}}Q_{\mathbf{JJ}}^{-1}(\hat{r}_{\mathbf{J}} - \mathbf{r}_{\mathbf{J}}) - \mu Q_{\mathbf{J}^c\mathbf{J}}Q_{\mathbf{JJ}}^{-1}\mathbf{r}_{\mathbf{J}},$$

which leads us to control the difference $\hat{r}_{\mathbf{J}} - \mathbf{r}_{\mathbf{J}}$. Using Lemma 12, we get

$$\|\hat{r}_{\mathbf{J}} - \mathbf{r}_{\mathbf{J}}\|_1 \leq \|\hat{w}_{\mathbf{J}} - \mathbf{w}_{\mathbf{J}}\|_\infty \left(\sum_{G\in\mathcal{G}_{\mathbf{J}}} \frac{\left\|d^G_{\mathbf{J}}\right\|_2^2}{\|d^G \circ w\|_2} + \sum_{G\in\mathcal{G}_{\mathbf{J}}} \frac{\|d^G \circ d^G \circ w\|_1^2}{\|d^G \circ w\|_2^3} \right),$$

where $w = t_0\hat{w} + (1-t_0)\mathbf{w}$ for some $t_0 \in (0,1)$.

Let $\overline{\mathbf{J}} = \{k \in \mathbf{J} : \mathbf{w}_k \neq 0\}$ and let φ be defined as

$$\varphi = \sup_{\substack{u\in\mathbb{R}^p : \overline{\mathbf{J}} \subset \{k\in\mathbf{J} : u_k \neq 0\} \subset \mathbf{J} \\ G\in\mathcal{G}_{\mathbf{J}}}} \frac{\|d^G \circ d^G \circ u\|_1}{\|d^G_{\overline{\mathbf{J}}} \circ d^G_{\overline{\mathbf{J}}} \circ u_{\overline{\mathbf{J}}}\|_1} \geq 1.$$

The term φ basically measures how close $\mathbf{J}$ and $\overline{\mathbf{J}}$ are, that is, how relevant the prior encoded by $\mathcal{G}$ about the hull $\mathbf{J}$ is. By using (12), we have

$$\|d^G \circ w\|_2^2 \geq \left\|d^G_{\overline{\mathbf{J}}} \circ w_{\overline{\mathbf{J}}}\right\|_2^2 \geq \|d^G_{\overline{\mathbf{J}}} \circ d^G_{\overline{\mathbf{J}}} \circ w_{\overline{\mathbf{J}}}\|_1 \frac{\nu}{3} \geq \|d^G \circ d^G \circ w\|_1 \frac{\nu}{3\varphi},$$

$$\|d^G \circ w\|_2 \geq \|d^G_{\overline{\mathbf{J}}} \circ w_{\overline{\mathbf{J}}}\|_2 \geq \|d^G_{\overline{\mathbf{J}}}\|_2 \frac{\nu}{3} \geq \left\|d^G_{\mathbf{J}}\right\|_2 \frac{\nu}{3\sqrt{\varphi}}$$

and

$$\|w\|_\infty \leq \frac{5}{3}\|\mathbf{w}\|_\infty.$$

Therefore we have

$$\begin{aligned}\|\hat{r}_{\mathbf{J}} - \mathbf{r}_{\mathbf{J}}\|_1 &\leq \|\hat{w}_{\mathbf{J}} - \mathbf{w}_{\mathbf{J}}\|_\infty \sum_{G\in\mathcal{G}_{\mathbf{J}}} \left(\frac{\left\|d^G_{\mathbf{J}}\right\|_2^2}{\|d^G \circ w\|_2} + \frac{5\varphi}{\nu} \frac{\|\mathbf{w}\|_\infty \left\|d^G_{\mathbf{J}} \circ d^G_{\mathbf{J}}\right\|_1}{\|d^G \circ w\|_2} \right) \\ &\leq \frac{3\sqrt{\varphi}\,\|\hat{w}_{\mathbf{J}} - \mathbf{w}_{\mathbf{J}}\|_\infty}{\nu} \left(1 + \frac{5\varphi\|\mathbf{w}\|_\infty}{\nu}\right) \sum_{G\in\mathcal{G}_{\mathbf{J}}} \left\|d^G_{\mathbf{J}}\right\|_2.\end{aligned}$$

Introducing $\alpha = \frac{18\varphi^{3/2}\|\mathbf{w}\|_\infty}{\nu^2} \sum_{G\in\mathcal{G}_{\mathbf{J}}} \left\|d^G_{\mathbf{J}}\right\|_2$, we thus have proved

$$\|\hat{r}_{\mathbf{J}} - \mathbf{r}_{\mathbf{J}}\|_1 \leq \alpha\,\|\hat{w}_{\mathbf{J}} - \mathbf{w}_{\mathbf{J}}\|_\infty. \tag{14}$$

By writing the Schur complement of Q on the block matrices $Q_{\mathbf{J}^c\mathbf{J}^c}$ and $Q_{\mathbf{JJ}}$, the positiveness of Q implies that the diagonal terms $\operatorname{diag}(Q_{\mathbf{J}^c\mathbf{J}}Q_{\mathbf{JJ}}^{-1}Q_{\mathbf{J}\mathbf{J}^c})$ are less than one, which results in $\|Q_{\mathbf{J}^c\mathbf{J}}Q_{\mathbf{JJ}}^{-1/2}\|_{\infty,2} \leq 1$. We then have

$$\begin{aligned}\left\|Q_{\mathbf{J}^c\mathbf{J}}Q_{\mathbf{JJ}}^{-1}(\hat{r}_{\mathbf{J}} - \mathbf{r}_{\mathbf{J}})\right\|_\infty &= \left\|Q_{\mathbf{J}^c\mathbf{J}}Q_{\mathbf{JJ}}^{-1/2}Q_{\mathbf{JJ}}^{-1/2}(\hat{r}_{\mathbf{J}} - \mathbf{r}_{\mathbf{J}})\right\|_\infty \\ &\leq \|Q_{\mathbf{J}^c\mathbf{J}}Q_{\mathbf{JJ}}^{-1/2}\|_{\infty,2}\|Q_{\mathbf{JJ}}^{-1/2}\|_2\,\|\hat{r}_{\mathbf{J}} - \mathbf{r}_{\mathbf{J}}\|_2 \\ &\leq \kappa^{-1/2}\,\|\hat{r}_{\mathbf{J}} - \mathbf{r}_{\mathbf{J}}\|_1 \\ &\leq \kappa^{-3/2}\alpha|\mathbf{J}|^{1/2}\left(\mu A(\mathbf{J}) + \|q_{\mathbf{J}}\|_\infty\right),\end{aligned}$$

where the last line comes from Equation (10) and (14). We get

$$(\Omega^c_{\mathbf{J}})^*[Q_{\mathbf{J}^c\mathbf{J}}Q_{\mathbf{JJ}}^{-1}(\hat{r}_{\mathbf{J}}-\mathbf{r}_{\mathbf{J}})] \leq \frac{\alpha|\mathbf{J}|^{1/2}}{\kappa^{3/2}a(\mathbf{J}^c)}\left(\mu A(\mathbf{J})+\|q_{\mathbf{J}}\|_\infty\right).$$

Thus, if the following inequalities are verified

$$\frac{\alpha|\mathbf{J}|^{1/2}A(\mathbf{J})}{\kappa^{3/2}a(\mathbf{J}^c)}\mu \leq \frac{\tau}{4},$$

$$\frac{\alpha|\mathbf{J}|^{1/2}}{\kappa^{3/2}a(\mathbf{J}^c)}\|q_{\mathbf{J}}\|_\infty \leq \frac{\tau}{4}, \tag{15}$$

$$(\Omega^c_{\mathbf{J}})^*[q_{\mathbf{J}^c|\mathbf{J}}] \leq \frac{\mu\tau}{2}, \tag{16}$$

we obtain

$$\begin{aligned}(\Omega^c_{\mathbf{J}})^*[\nabla L(\hat{w})_{\mathbf{J}^c}] &\leq (\Omega^c_{\mathbf{J}})^*[-q_{\mathbf{J}^c|\mathbf{J}}-\mu Q_{\mathbf{J}^c\mathbf{J}}Q_{\mathbf{JJ}}^{-1}\mathbf{r}_{\mathbf{J}}]\\ &\leq (\Omega^c_{\mathbf{J}})^*[-q_{\mathbf{J}^c|\mathbf{J}}]+\mu(1-\tau)+\mu\tau/2 \leq \mu,\end{aligned}$$

that is, $\mathbf{J}$ is exactly selected.

Combined with earlier constraints, this leads to the first part of the desired proposition.

We now need to make sure that the conditions (11), (15) and (16) hold with high probability. To this end, we upperbound, using Gaussian concentration inequalities, two tail-probabilities. First, $q_{\mathbf{J}^c|\mathbf{J}}$ is a centered Gaussian random vector with covariance matrix

$$\begin{aligned}\mathbb{E}\left[q_{\mathbf{J}^c|\mathbf{J}}q_{\mathbf{J}^c|\mathbf{J}}^\top\right] &= \mathbb{E}\left[q_{\mathbf{J}^c}q_{\mathbf{J}^c}^\top - q_{\mathbf{J}^c}q_{\mathbf{J}}^\top Q_{\mathbf{JJ}}^{-1}Q_{\mathbf{JJ}^c} - Q_{\mathbf{J}^c\mathbf{J}}Q_{\mathbf{JJ}}^{-1}q_{\mathbf{J}}q_{\mathbf{J}^c}^\top + Q_{\mathbf{J}^c\mathbf{J}}Q_{\mathbf{JJ}}^{-1}q_{\mathbf{J}}q_{\mathbf{J}}^\top Q_{\mathbf{JJ}}^{-1}Q_{\mathbf{JJ}^c}\right]\\ &= \frac{\sigma^2}{n}Q_{\mathbf{J}^c\mathbf{J}^c|\mathbf{J}},\end{aligned}$$

where $Q_{\mathbf{J}^c\mathbf{J}^c|\mathbf{J}} = Q_{\mathbf{J}^c\mathbf{J}^c} - Q_{\mathbf{J}^c\mathbf{J}}Q_{\mathbf{JJ}}^{-1}Q_{\mathbf{JJ}^c}$. In particular, $(\Omega^c_{\mathbf{J}})^*[q_{\mathbf{J}^c|\mathbf{J}}]$ has the same distribution as $\psi(W)$, with $\psi : u \mapsto (\Omega^c_{\mathbf{J}})^*(\sigma n^{-1/2}Q^{1/2}_{\mathbf{J}^c\mathbf{J}^c|\mathbf{J}}u)$ and W a centered Gaussian random variable with unit covariance matrix.

Since for any u we have $u^\top Q_{\mathbf{J}^c\mathbf{J}^c|\mathbf{J}}u \leq u^\top Q_{\mathbf{J}^c\mathbf{J}^c}u \leq \left\|Q^{1/2}\right\|_2^2\|u\|_2^2$, by using Sudakov-Fernique inequality (Adler, 1990, Theorem 2.9), we get:

$$\begin{aligned}\mathbb{E}[(\Omega^c_{\mathbf{J}})^*[q_{\mathbf{J}^c|\mathbf{J}}] = \mathbb{E}\sup_{\Omega^c_{\mathbf{J}}(u)\leq 1} u^\top q_{\mathbf{J}^c|\mathbf{J}} &\leq \sigma n^{-1/2}\|Q\|_2^{1/2}\mathbb{E}\sup_{\Omega^c_{\mathbf{J}}(u)\leq 1} u^\top W\\ &\leq \sigma n^{-1/2}\|Q\|_2^{1/2}\mathbb{E}[(\Omega^c_{\mathbf{J}})^*(W)].\end{aligned}$$

In addition, we have

$$|\psi(u)-\psi(v)| \leq \psi(u-v) \leq \sigma n^{-1/2}a(\mathbf{J}^c)^{-1}\left\|Q^{1/2}_{\mathbf{J}^c\mathbf{J}^c|\mathbf{J}}(u-v)\right\|_\infty.$$

On the other hand, since Q has unit diagonal and $Q_{\mathbf{J}^c\mathbf{J}}Q_{\mathbf{JJ}}^{-1}Q_{\mathbf{JJ}^c}$ has diagonal terms less than one, $Q_{\mathbf{J}^c\mathbf{J}^c|\mathbf{J}}$ also has diagonal terms less than one, which implies that $\|Q^{1/2}_{\mathbf{J}^c\mathbf{J}^c|\mathbf{J}}\|_{\infty,2} \leq 1$. Hence ψ is a Lipschitz function with Lipschitz constant upper bounded by $\sigma n^{-1/2}a(\mathbf{J}^c)^{-1}$. Thus by concentration

of Lipschitz functions of multivariate standard random variables (Massart, 2003, Theorem 3.4), we have for $t > 0$:

$$\mathbb{P}\Big[(\Omega_{\mathbf{J}}^c)^*[q_{\mathbf{J}^c|\mathbf{J}}] \geq t + \sigma n^{-1/2}\|Q\|_2^{1/2}\mathbb{E}\left[(\Omega_{\mathbf{J}}^c)^*(W)\right]\Big] \leq \exp\left(-\frac{nt^2 a(\mathbf{J}^c)^2}{2\sigma^2}\right).$$

Applied for $t = \mu\tau/2 \geq 2\sigma n^{-1/2}\|Q\|_2^{1/2}\mathbb{E}\left[(\Omega_{\mathbf{J}}^c)^*(W)\right]$, we get (because $(u-1)^2 \geq u^2/4$ for $u \geq 2$):

$$\mathbb{P}\big[(\Omega_{\mathbf{J}}^c)^*[q_{\mathbf{J}^c|\mathbf{J}}] \geq t\big] \leq \exp\left(-\frac{n\mu^2\tau^2 a(\mathbf{J}^c)^2}{32\sigma^2}\right).$$

It finally remains to control the term $\mathbb{P}(\|q_{\mathbf{J}}\|_\infty \geq \xi)$, with

$$\xi = \frac{\kappa\nu}{3}\min\left\{1, \frac{3\tau\kappa^{1/2}a(\mathbf{J}^c)}{4\alpha\nu}\right\}.$$

We can apply classical inequalities for standard random variables (Massart, 2003, Theorem 3.4) that directly lead to

$$\mathbb{P}(\|q_{\mathbf{J}}\|_\infty \geq \xi) \leq 2|\mathbf{J}|\exp\left(-\frac{n\xi^2}{2\sigma^2}\right).$$

To conclude, Theorem 7 holds with

$$\begin{aligned}
C_1(\mathcal{G},\mathbf{J}) &= \frac{a(\mathbf{J}^c)^2}{16}, \\
C_2(\mathcal{G},\mathbf{J}) &= \left(\frac{\kappa\nu}{3}\min\left\{1, \frac{\tau\kappa^{1/2}a(\mathbf{J}^c)\nu}{24\varphi^{3/2}\|\mathbf{w}\|_\infty \sum_{G\in\mathcal{G}_{\mathbf{J}}}\left\|d_{\mathbf{J}}^G\right\|_2}\right\}\right)^2, \\
C_3(\mathcal{G},\mathbf{J}) &= 4\|Q\|_2^{1/2}\mathbb{E}\left[(\Omega_{\mathbf{J}}^c)^*(W)\right],
\end{aligned}$$

and

$$C_4(\mathcal{G},\mathbf{J}) = \frac{\kappa\nu}{3A(\mathbf{J})}\min\left\{1, \frac{\tau\kappa^{1/2}a(\mathbf{J}^c)\nu}{24\varphi^{3/2}\,\|\mathbf{w}\|_\infty \sum_{G\in\mathcal{G}_{\mathbf{J}}}\left\|d_{\mathbf{J}}^G\right\|_2}\right\},$$

where we recall the definitions: W a centered Gaussian random variable with unit covariance matrix, $\overline{\mathbf{J}} = \{j \in \mathbf{J} : \mathbf{w}_j \neq 0\}$, $\nu = \min\{|\mathbf{w}_j|;\ j \in \overline{\mathbf{J}}\}$,

$$\varphi = \sup_{\substack{u\in\mathbb{R}^p : \overline{\mathbf{J}} \subset \{k\in\mathbf{J} : u_k \neq 0\} \subset \mathbf{J} \\ G\in\mathcal{G}_{\mathbf{J}}}} \frac{\|d^G \circ d^G \circ u\|_1}{\|d_{\overline{\mathbf{J}}}^G \circ d_{\overline{\mathbf{J}}}^G \circ u_{\overline{\mathbf{J}}}\|_1},$$

$\kappa = \lambda_{\min}(Q_{\mathbf{JJ}}) > 0$ and $\tau > 0$ such that $(\Omega_{\mathbf{J}}^c)^*[Q_{\mathbf{J}^c\mathbf{J}}Q_{\mathbf{JJ}}^{-1}\mathbf{r}] < 1-\tau$.

Appendix H. A First Order Approach to Solve Problems (2) and (3)

Both regularized minimization problems in Equation (2) and Equation (3) (that just differ in the squaring of Ω) can be solved by using generic toolboxes for second-order cone programming (SOCP) (Boyd and Vandenberghe, 2004). We propose here a first order approach that takes up

ideas from Micchelli and Pontil (2006) and Rakotomamonjy et al. (2008) and that is based on the following variational equalities: for $x \in \mathbb{R}^p$, we have

$$\|x\|_1^2 = \min_{\substack{z \in \mathbb{R}_+^p, \\ \sum_{j=1}^p z_j \leq 1}} \sum_{j=1}^p \frac{x_j^2}{z_j},$$

whose minimum is uniquely attained for $z_j = |x_j| / \|x\|_1$. Similarly, we have

$$2\|x\|_1 = \min_{z \in \mathbb{R}_+^p} \sum_{j=1}^p \frac{x_j^2}{z_j} + \|z\|_1,$$

whose minimum is uniquely obtained for $z_j = |x_j|$. Thus, we can equivalently rewrite Equation (2) as

$$\min_{\substack{w \in \mathbb{R}^p, \\ (\eta^G)_{G \in \mathcal{G}} \in \mathbb{R}_+^{|\mathcal{G}|}}} \frac{1}{n} \sum_{i=1}^n \ell\big(y_i, w^\top x_i\big) + \frac{\mu}{2} \sum_{j=1}^p w_j^2 \zeta_j^{-1} + \frac{\mu}{2} \big\|(\eta^G)_{G \in \mathcal{G}}\big\|_1, \tag{17}$$

with $\zeta_j = \big(\sum_{G \ni j} (d_j^G)^2 (\eta^G)^{-1}\big)^{-1}$. In the same vein, Equation (3) is equivalent to

$$\min_{\substack{w \in \mathbb{R}^p, \\ (\eta^G)_{G \in \mathcal{G}} \in \mathbb{R}_+^{|\mathcal{G}|}, \\ \sum_{G \in \mathcal{G}} \eta^G \leq 1}} \frac{1}{n} \sum_{i=1}^n \ell\big(y_i, w^\top x_i\big) + \frac{\lambda}{2} \sum_{j=1}^p w_j^2 \zeta_j^{-1}, \tag{18}$$

where ζ_j is defined as above. The reformulations Equation (17) and Equation (18) are *jointly* convex in $\{w, (\eta^G)_{G \in \mathcal{G}}\}$ and lend themselves well to a simple alternating optimization scheme between w (for instance, w can be computed in closed-form when the square loss is used) and $(\eta^G)_{G \in \mathcal{G}}$ (whose optimal value is always a closed-form solution). If the variables $(\eta^G)_{G \in \mathcal{G}} \in \mathbb{R}_+^{|\mathcal{G}|}$ are bounded away from zero by a smoothing parameter, the convergence of this scheme is guaranteed by standard results about block coordinate descent procedures (Bertsekas, 1999).

This first order approach is computationally appealing since it allows *warm-restart*, which can dramatically speed up the computation over regularization paths. Moreover, it does not make any assumptions on the nature of the family of groups $\mathcal{G}$.

Appendix I. Technical Lemmas

In this last section of the appendix, we give several technical lemmas. We consider $I \subseteq \{1, \ldots, p\}$ and $\mathcal{G}_I = \{G \in \mathcal{G};\ G \cap I \neq \varnothing\} \subseteq \mathcal{G}$, that is, the set of active groups when the variables I are selected.

We begin with a dual formulation of Ω^* obtained by conic duality (Boyd and Vandenberghe, 2004):

Lemma 9 *Let $u_I \in \mathbb{R}^{|I|}$. We have*

$$\begin{aligned} (\Omega_I)^*[u_I] \quad &= \quad \min_{(\xi_I^G)_{G \in \mathcal{G}_I}} \max_{G \in \mathcal{G}_I} \|\xi_I^G\|_2 \\ &\quad s.t. \quad u_j + \sum_{G \in \mathcal{G}_I, G \ni j} d_j^G \xi_j^G = 0 \text{ and } \xi_j^G = 0 \text{ if } j \notin G. \end{aligned}$$

Proof By definition of $(\Omega_I)^*[u_I]$, we have

$$(\Omega_I)^*[u_I] = \max_{\Omega_I(v_I)\leq 1} u_I^\top v_I.$$

By introducing the primal variables $(\alpha_G)_{G\in\mathcal{G}_I} \in \mathbb{R}^{|\mathcal{G}_I|}$, we can rewrite the previous maximization problem as

$$(\Omega_I)^*[u_I] = \max_{\sum_{G\in\mathcal{G}_I}\alpha_G\leq 1} u_I^\top v_I, \quad \text{s.t.} \quad \forall\, G\in\mathcal{G}_I,\ \|d_I^G \circ u_{G\cap I}\|_2 \leq \alpha_G,$$

which is a second-order cone program (SOCP) with $|\mathcal{G}_I|$ second-order cone constraints. This primal problem is convex and satisfies Slater's conditions for generalized conic inequalities, which implies that strong duality holds (Boyd and Vandenberghe, 2004). We now consider the Lagrangian $\mathcal{L}$ defined as

$$\mathcal{L}(v_I,\alpha_G,\gamma,\tau_G,\xi_I^G) = u_I^\top v_I + \gamma(1-\sum_{G\in\mathcal{G}_I}\alpha_G) + \sum_{G\in\mathcal{G}_I}\begin{pmatrix}\alpha_G\\ d_I^G\circ u_{G\cap I}\end{pmatrix}^\top\begin{pmatrix}\tau_G\\ \xi_I^G\end{pmatrix},$$

with the dual variables $\{\gamma,(\tau_G)_{G\in\mathcal{G}_I},(\xi_I^G)_{G\in\mathcal{G}_I}\} \in \mathbb{R}_+\times\mathbb{R}^{|\mathcal{G}_I|}\times\mathbb{R}^{|I|\times|\mathcal{G}_I|}$ such that for all $G\in\mathcal{G}_I$, $\xi_j^G = 0$ if $j\notin G$ and $\|\xi_I^G\|_2 \leq \tau_G$. The dual function is obtained by taking the derivatives of $\mathcal{L}$ with respect to the primal variables v_I and $(\alpha_G)_{G\in\mathcal{G}_I}$ and equating them to zero, which leads to

$$\begin{array}{lll}\forall j\in I, & u_j + \sum_{G\in\mathcal{G}_I, G\ni j} d_j^G\xi_j^G & = 0\\ \forall G\in\mathcal{G}_I, & \gamma - \tau_G & = 0.\end{array}$$

After simplifying the Lagrangian, the dual problem then reduces to

$$\min_{\gamma,(\xi_I^G)_{G\in\mathcal{G}_I}} \gamma \quad \text{s.t.} \begin{cases}\forall j\in I, u_j + \sum_{G\in\mathcal{G}_I,G\ni j} d_j^G\xi_j^G = 0 \text{ and } \xi_j^G = 0 \text{ if } j\notin G,\\ \forall G\in\mathcal{G}_I, \|\xi_I^G\|_2\leq\gamma,\end{cases}$$

which is equivalent to the displayed result. ■

Since we cannot compute in closed-form the solution of the previous optimization problem, we focus on a different *but closely related* problem, that is, when we replace the objective $\max_{G\in\mathcal{G}_I}\|\xi_I^G\|_2$ by $\max_{G\in\mathcal{G}_I}\|\xi_I^G\|_\infty$, to obtain a *meaningful* feasible point:

Lemma 10 *Let $u_I\in\mathbb{R}^{|I|}$. The following problem*

$$\begin{array}{ll}\min_{(\xi_I^G)_{G\in\mathcal{G}_I}} & \max_{G\in\mathcal{G}_I}\|\xi_I^G\|_\infty\\ s.t. & u_j + \sum_{G\in\mathcal{G}_I,G\ni j} d_j^G\xi_j^G = 0 \text{ and } \xi_j^G = 0 \text{ if } j\notin G,\end{array}$$

is minimized for $(\xi_j^G)^* = -\dfrac{u_j}{\sum_{H\in j,H\in\mathcal{G}_I} d_j^H}$.

Proof We proceed by contradiction. Let us assume there exists $(\xi_I^G)_{G\in\mathcal{G}_I}$ such that

$$\begin{aligned}\max_{G\in\mathcal{G}_I}\|\xi_I^G\|_\infty &< \max_{G\in\mathcal{G}_I}\|(\xi_I^G)^*\|_\infty\\ &= \max_{G\in\mathcal{G}_I}\max_{j\in G}\frac{|u_j|}{\sum_{H\in j,H\in\mathcal{G}_I} d_j^H}\\ &= \frac{|u_{j_0}|}{\sum_{H\in j_0,H\in\mathcal{G}_I} d_{j_0}^H},\end{aligned}$$

where we denote by j_0 an argmax of the latter maximization. We notably have for all $G \ni j_0$:

$$|\xi^G_{j_0}| < \frac{|u_{j_0}|}{\sum_{H \in j_0, H \in \mathcal{G}_I} d^H_{j_0}}.$$

By multiplying both sides by $d^G_{j_0}$ and by summing over $G \ni j_0$, we get

$$|u_{j_0}| = |\sum_{G \in \mathcal{G}_I, G \ni j_0} d^G_{j_0} \xi^G_{j_0}| \leq \sum_{G \ni j_0} d^G_{j_0} |\xi^G_{j_0}| < |u_{j_0}|,$$

which leads to a contradiction. ■

We now give an upperbound on Ω^* based on Lemma 9 and Lemma 10:

Lemma 11 *Let $u_I \in \mathbb{R}^{|I|}$. We have*

$$(\Omega_I)^*[u_I] \leq \max_{G \in \mathcal{G}_I} \left\{ \sum_{j \in G} \left\{ \frac{u_j}{\sum_{H \in j, H \in \mathcal{G}_I} d^H_j} \right\}^2 \right\}^{\frac{1}{2}}.$$

Proof We simply plug the minimizer obtained in Lemma 10 into the problem of Lemma 9. ■

We now derive a lemma to control the difference of the gradient of Ω_J evaluated in two points:

Lemma 12 *Let u_J, v_J be two nonzero vectors in $\mathbb{R}^{|J|}$. Let us consider the mapping $w_J \mapsto r(w_J) = \sum_{G \in \mathcal{G}_J} \frac{d^G_J \circ d^G_J \circ w_J}{\|d^G_J \circ w_J\|_2} \in \mathbb{R}^{|J|}$. There exists $z_J = t_0 u_J + (1-t_0) v_J$ for some $t_0 \in (0,1)$ such that*

$$\|r(u_J) - r(v_J)\|_1 \leq \|u_J - v_J\|_\infty \left(\sum_{G \in \mathcal{G}_J} \frac{\|d^G_J\|_2^2}{\|d^G_J \circ z_J\|_2} + \sum_{G \in \mathcal{G}_J} \frac{\|d^G_J \circ d^G_J \circ z_J\|_1^2}{\|d^G_J \circ z_J\|_2^3} \right).$$

Proof For $j,k \in J$, we have

$$\frac{\partial r_j}{\partial w_k}(w_J) = \sum_{G \in \mathcal{G}_J} \frac{(d^G_j)^2}{\|d^G_J \circ w_J\|_2} \mathbb{I}_{j=k} - \sum_{G \in \mathcal{G}_J} \frac{(d^G_j)^2 w_j}{\|d^G_J \circ w_J\|_2^3} (d^G_k)^2 w_k,$$

with $\mathbb{I}_{j=k} = 1$ if $j = k$ and 0 otherwise. We then consider $t \in [0,1] \mapsto h_j(t) = r_j(t u_J + (1-t) v_J)$. The mapping h_j being continuously differentiable, we can apply the mean-value theorem: there exists $t_0 \in (0,1)$ such that

$$h_j(1) - h_j(0) = \frac{\partial h_j(t)}{\partial t}(t_0).$$

We then have

$$\begin{aligned} |r_j(u_J) - r_j(v_J)| &\leq \sum_{k \in J} \left| \frac{\partial r_j}{\partial w_k}(z) \right| |u_k - v_k| \\ &\leq \|u_J - v_J\|_\infty \left(\sum_{G \in \mathcal{G}_J} \frac{(d^G_j)^2}{\|d^G_J \circ z_J\|_2} + \sum_{k \in J} \sum_{G \in \mathcal{G}_J} \frac{(d^G_j)^2 |z_j|}{\|d^G_J \circ z_J\|_2^3} (d^G_k)^2 |z_k| \right), \end{aligned}$$

which leads to

$$\|r(u_J) - r(v_J)\|_1 \leq \|u_J - v_J\|_\infty \left(\sum_{G \in \mathcal{G}_J} \frac{\|d_J^G\|_2^2}{\|d_J^G \circ z_J\|_2} + \sum_{G \in \mathcal{G}_J} \frac{\|d_J^G \circ d_J^G \circ z_J\|_1^2}{\|d_J^G \circ z_J\|_2^3} \right).$$

■

Given an active set $J \subseteq \{1,\ldots,p\}$ and a direct parent $K \in \Pi_{\mathcal{P}}(J)$ of J in the DAG of nonzero patterns, we have the following result:

Lemma 13 *For all $G \in \mathcal{G}_K \backslash \mathcal{G}_J$, we have $K \backslash J \subseteq G$.*

Proof We proceed by contradiction. We assume there exists $G_0 \in \mathcal{G}_K \backslash \mathcal{G}_J$ such that $K \backslash J \nsubseteq G_0$. Given that $K \in \mathcal{P}$, there exists $\mathcal{G}' \subseteq \mathcal{G}$ verifying $K = \bigcap_{G \in \mathcal{G}'} G^c$. Note that $G_0 \notin \mathcal{G}'$ since by definition $G_0 \cap K \neq \varnothing$.

We can now build the pattern $\tilde{K} = \bigcap_{G \in \mathcal{G}' \cup \{G_0\}} G^c = K \cap G_0^c$ that belongs to $\mathcal{P}$. Moreover, $\tilde{K} = K \cap G_0^c \subset K$ since we assumed $G_0^c \cap K \neq \varnothing$. In addition, we have that $J \subset K$ and $J \subset G_0^c$ because $K \in \Pi_{\mathcal{P}}(J)$ and $G_0 \in \mathcal{G}_K \backslash \mathcal{G}_J$. This results in $J \subset \tilde{K} \subset K$, which is impossible by definition of K. ■

We give below an important Lemma to characterize the solutions of Problem (2).

Lemma 14 *The vector $\hat{w} \in \mathbb{R}^p$ is a solution of*

$$\min_{w \in \mathbb{R}^p} L(w) + \mu \Omega(w)$$

if and only if

$$\begin{cases} \nabla L(\hat{w})_{\hat{J}} + \mu \hat{r}_{\hat{J}} = 0 \\ (\Omega^c_{\hat{J}})^* [\nabla L(\hat{w})_{\hat{J}^c}] \leq \mu, \end{cases}$$

with $\hat{J}$ the hull of $\{j \in \{1,\ldots,p\}, \hat{w}_j \neq 0\}$ and the vector $\hat{r} \in \mathbb{R}^p$ defined as

$$\hat{r} = \sum_{G \in \mathcal{G}_{\hat{J}}} \frac{d^G \circ d^G \circ \hat{w}}{\|d^G \circ \hat{w}\|_2}.$$

In addition, the solution $\hat{w}$ satisfies

$$\Omega^*[\nabla L(\hat{w})] \leq \mu.$$

Proof The problem

$$\min_{w \in \mathbb{R}^p} L(w) + \mu \Omega(w) = \min_{w \in \mathbb{R}^p} F(w)$$

being convex, the directional derivative optimality condition are necessary and sufficient (Borwein and Lewis, 2006, Propositions 2.1.1-2.1.2). Therefore, the vector $\hat{w}$ is a solution of the previous problem if and only if for all directions $u \in \mathbb{R}^p$, we have

$$\lim_{\substack{\varepsilon \to 0 \\ \varepsilon > 0}} \frac{F(\hat{w} + \varepsilon u) - F(\hat{w})}{\varepsilon} \geq 0.$$

Some algebra leads to the following equivalent formulation

$$\forall u \in \mathbb{R}^p,\ u^\top \nabla L(\hat{w}) + \mu u_{\hat{J}}^\top \hat{r}_{\hat{J}} + \mu (\Omega^c_{\hat{J}})[u_{\hat{J}^c}] \geq 0. \tag{19}$$

The first part of the lemma then comes from the projections on $\hat{J}$ and $\hat{J}^c$.

An application of the Cauchy-Schwartz inequality on $u_{\hat{J}}^\top \hat{r}_{\hat{J}}$ gives for all $u \in \mathbb{R}^p$

$$u_{\hat{J}}^\top \hat{r}_{\hat{J}} \leq (\Omega_{\hat{J}})[u_{\hat{J}}].$$

Combined with Equation (19), we get $\forall u \in \mathbb{R}^p,\ u^\top \nabla L(\hat{w}) + \mu \Omega(u) \geq 0$, hence the second part of the lemma. ■

We end up with a lemma regarding the dual norm of the sum of two *disjoint* norms (see Rockafellar, 1970):

Lemma 15 *Let A and B be a partition of $\{1,\dots,p\}$, that is, $A \cap B = \varnothing$ and $A \cup B = \{1,\dots,p\}$. We consider two norms $u_A \in \mathbb{R}^{|A|} \mapsto \|u_A\|_A$ and $u_B \in \mathbb{R}^{|B|} \mapsto \|u_B\|_B$, with dual norms $\|v_A\|_A^*$ and $\|v_B\|_B^*$. We have*

$$\max_{\|u_A\|_A + \|u_B\|_B \leq 1} u^\top v = \max\left\{\|v_A\|_A^*,\ \|v_B\|_B^*\right\}.$$

References

R. J. Adler. *An Introduction to Continuity, Extrema, and Related Topics for General Gaussian Processes*. IMS, 1990.

A. Argyriou, T. Evgeniou, and M. Pontil. Convex multi-task feature learning. *Machine Learning*, 73(3):243–272, 2008.

F. Bach. Exploring large feature spaces with hierarchical multiple kernel learning. In *Advances in Neural Information Processing Systems*, 2008a.

F. Bach. Consistency of the group lasso and multiple kernel learning. *Journal of Machine Learning Research*, 9:1179–1225, 2008b.

F. Bach. Bolasso: model consistent Lasso estimation through the bootstrap. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2008c.

F. Bach. Self-concordant analysis for logistic regression. *Electronic Journal of Statistics*, 4:384–414, 2009.

F. Bach, R. Jenatton, J. Mairal, and G. Obozinski. Convex optimization with sparsity-inducing norms. In *Optimization for Machine Learning*. MIT press, 2011. To appear.

R. G. Baraniuk, V. Cevher, M. F. Duarte, and C. Hegde. Model-based compressive sensing. *IEEE Transactions on Information Theory*, 56:1982–2001, 2010.

A. Beck and M. Teboulle. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM Journal on Imaging Sciences*, 2(1):183–202, 2009.

D. P. Bertsekas. *Nonlinear Programming*. Athena Scientific, 1999.

P. Bickel, Y. Ritov, and A. Tsybakov. Simultaneous analysis of Lasso and Dantzig selector. *Annals of Statistics*, 37(4):1705–1732, 2009.

J. M. Borwein and A. S. Lewis. *Convex Analysis and Nonlinear Optimization: Theory and Examples*. Springer, 2006.

S. P. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004.

P. J. Cameron. *Combinatorics: Topics, Techniques, Algorithms*. Cambridge University Press, 1994.

V. Cevher, M. F. Duarte, C. Hegde, and R. G. Baraniuk. Sparse signal recovery using markov random fields. In *Advances in Neural Information Processing Systems*, 2008.

F. R. K. Chung. *Spectral Graph Theory*. American Mathematical Society, 1997.

P. L. Combettes and J.-C. Pesquet. Proximal splitting methods in signal processing. In *Fixed-Point Algorithms for Inverse Problems in Science and Engineering*. Springer, 2010.

N. Dalal, B. Triggs, and C. Schmid. Human detection using oriented histograms of flow and appearance. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2006.

J. P. Doignon and J. C. Falmagne. *Knowledge Spaces*. Springer-Verlag, 1998.

C. Dossal. A necessary and sufficient condition for exact recovery by l1 minimization. Technical report, HAL-00164738:1, 2007.

B. Efron, T. Hastie, I. Johnstone, and R. Tibshirani. Least angle regression. *Annals of Statistics*, 32 (2):407–451, 2004.

W. Fu and K. Knight. Asymptotics for Lasso-type estimators. *Annals of Statistics*, 28(5):1356–1378, 2000.

J. J. Fuchs. Recovery of exact sparse representations in the presence of bounded noise. *IEEE Transactions on Information Theory*, 51(10):3601–3608, 2005.

A. Gramfort and M. Kowalski. Improving M/EEG source localization with an inter-condition sparse prior. In *IEEE International Symposium on Biomedical Imaging*, 2009.

H. Harzallah, F. Jurie, and C. Schmid. Combining efficient object localization and image classification. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2009.

L. He and L. Carin. Exploiting structure in wavelet-based Bayesian compressive sensing. *IEEE Transactions on Signal Processing*, 57:3488–3497, 2009.

J. Huang and T. Zhang. The benefit of group sparsity. *Annals of Statistics*, 38(4):1978–2004, 2010.

J. Huang, T. Zhang, and D. Metaxas. Learning with structured sparsity. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2009.

L. Jacob, G. Obozinski, and J.-P. Vert. Group Lasso with overlaps and graph Lasso. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2009.

R. Jenatton, J. Mairal, G. Obozinski, and F. Bach. Proximal methods for sparse hierarchical dictionary learning. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2010a.

R. Jenatton, G. Obozinski, and F. Bach. Structured sparse principal component analysis. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2010b.

R. Jenatton, A. Gramfort, V. Michel, G. Obozinski, F. Bach, and B. Thirion. Multi-scale mining of fMRI data with hierarchical structured sparsity. In *International Workshop on Pattern Recognition in Neuroimaging (PRNI)*, 2011a.

R. Jenatton, J. Mairal, G. Obozinski, and F. Bach. Proximal methods for hierarchical sparse coding. *Journal of Machine Learning Research*, 12:2297–2334, 2011b.

K. Kavukcuoglu, M. A. Ranzato, R. Fergus, and Y. Le-Cun. Learning invariant features through topographic filter maps. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2009.

S. Kim and E. P. Xing. Tree-guided group Lasso for multi-task regression with structured sparsity. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2010.

H. Lee, A. Battle, R. Raina, and A. Y. Ng. Efficient sparse coding algorithms. In *Advances in Neural Information Processing Systems*, 2007.

E. Levina, A. Rothman, and J. Zhu. Sparse estimation of large covariance matrices via a nested Lasso penalty. *Annals of Applied Statistics*, 2(1):245–263, 2008.

J. Mairal, F. Bach, J. Ponce, and G. Sapiro. Online learning for matrix factorization and sparse coding. *Journal of Machine Learning Research*, 11(1):19–60, 2010a.

J. Mairal, R. Jenatton, G. Obozinski, and F. Bach. Network flow algorithms for structured sparsity. In *Advances in Neural Information Processing Systems*, 2010b.

J. Mairal, R. Jenatton, G. Obozinski, and F. Bach. Convex and network flow optimization for structured sparsity. Technical report, Preprint arXiv:1104.1872, 2011. To appear in Journal Machine Learning Research.

A. F. T. Martins, N. A. Smith, P. M. Q. Aguiar, and M. A. T. Figueiredo. Structured sparsity in structured prediction. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, 2011.

P. Massart. *Concentration Inequalities and Model Selection: Ecole d'été de Probabilités de Saint-Flour 23*. Springer, 2003.

N. Meinshausen and P. Bühlmann. Stability selection. *Journal of the Royal Statistical Society. Series B*, 72(4):417–473, 2010.

C. A. Micchelli and M. Pontil. Learning the kernel function via regularization. *Journal of Machine Learning Research*, 6(2):1099, 2006.

Y. Nardi and A. Rinaldo. On the asymptotic properties of the group Lasso estimator for linear models. *Electronic Journal of Statistics*, 2:605–633, 2008.

S. Negahban, P. Ravikumar, M. J. Wainwright, and B. Yu. A unified framework for high-dimensional analysis of M-estimators with decomposable regularizers. In *Advances in Neural Information Processing Systems*, 2009.

Y. Nesterov. Gradient methods for minimizing composite objective function. Technical report, Center for Operations Research and Econometrics (CORE), Catholic University of Louvain, 2007.

G. Obozinski, B. Taskar, and M. I. Jordan. Joint covariate selection and joint subspace selection for multiple classification problems. *Statistics and Computing*, pages 1–22, 2009.

M. R. Osborne, B. Presnell, and B. A. Turlach. On the lasso and its dual. *Journal of Computational and Graphical Statistics*, 9:319–337, 2000.

A. Rakotomamonjy, F. Bach, S. Canu, and Y. Grandvalet. SimpleMKL. *Journal of Machine Learning Research*, 9:2491–2521, 2008.

F. Rapaport, E. Barillot, and J.-P. Vert. Classification of arrayCGH data using fused SVM. *Bioinformatics*, 24(13):i375–i382, Jul 2008.

R. T. Rockafellar. *Convex Analysis*. Princeton University Press, 1970.

S. Rosset, J. Zhu, and T. Hastie. Boosting as a regularized path to a maximum margin classifier. *Journal of Machine Learning Research*, 5:941–973, 2004.

V. Roth and B. Fischer. The group-Lasso for generalized linear models: uniqueness of solutions and efficient algorithms. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2008.

M. Schmidt and K. Murphy. Convex structure learning in log-linear models: Beyond pairwise potentials. In *Proceedings of the International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2010.

P. Soille. *Morphological Image Analysis: Principles and Applications*. Springer, 2003.

R. Tibshirani. Regression shrinkage and selection via the Lasso. *Journal of the Royal Statistical Society. Series B*, pages 267–288, 1996.

K. C. Toh, M. J. Todd, and R. H. Tütüncü. SDPT3–a MATLAB software package for semidefinite programming, version 1.3. *Optimization Methods and Software*, 11(1):545–581, 1999.

R. H. Tütüncü, K. C. Toh, and M. J. Todd. Solving semidefinite-quadratic-linear programs using SDPT3. *Mathematical Programming*, 95(2):189–217, 2003.

G. Varoquaux, R. Jenatton, A. Gramfort, G. Obozinski, B. Thirion, and F. Bach. Sparse structured dictionary learning for brain resting-state activity modeling. In *NIPS Workshop on Practical Applications of Sparse Modeling: Open Issues and New Directions*, 2010.

M. J. Wainwright. Sharp thresholds for noisy and high-dimensional recovery of sparsity using ℓ_1-constrained quadratic programming. *IEEE Transactions on Information Theory*, 55:2183–2202, 2009.

Z. J. Xiang, Y. T. Xi, U. Hasson, and P. J. Ramadge. Boosting with spatial regularization. In *Advances in Neural Information Processing Systems*, 2009.

G. X. Yuan, K. W. Chang, C. J. Hsieh, and C. J. Lin. Comparison of optimization methods and software for large-scale l1-regularized linear classification. *Journal of Machine Learning Research*, 11:3183–3234, 2010.

M. Yuan and Y. Lin. Model selection and estimation in regression with grouped variables. *Journal of the Royal Statistical Society. Series B*, 68(1):49–67, 2006.

T. Zhang. Some sharp performance bounds for least squares regression with l1 regularization. *Annals of Statistics*, 37(5A):2109–2144, 2009.

P. Zhao and B. Yu. On model selection consistency of Lasso. *Journal of Machine Learning Research*, 7:2541–2563, 2006.

P. Zhao, G. Rocha, and B. Yu. The composite absolute penalties family for grouped and hierarchical variable selection. *Annals of Statistics*, 37(6A):3468–3497, 2009.

H. Zou. The adaptive Lasso and its oracle properties. *Journal of the American Statistical Association*, 101(476):1418–1429, 2006.

H. Zou and T. Hastie. Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society. Series B*, 67(2):301–320, 2005.

Journal of Machine Learning Research 12 (2011) 2825-2830 Submitted 3/11; Revised 8/11; Published 10/11

Scikit-learn: Machine Learning in Python

Fabian Pedregosa FABIAN.PEDREGOSA@INRIA.FR
Gaël Varoquaux GAEL.VAROQUAUX@NORMALESUP.ORG
Alexandre Gramfort ALEXANDRE.GRAMFORT@INRIA.FR
Vincent Michel VINCENT.MICHEL@LOGILAB.FR
Bertrand Thirion BERTRAND.THIRION@INRIA.FR
Parietal, INRIA Saclay
Neurospin, Bât 145, CEA Saclay
91191 Gif sur Yvette – France

Olivier Grisel OLIVIER.GRISEL@ENSTA.FR
Nuxeo
20 rue Soleillet
75 020 Paris – France

Mathieu Blondel MBLONDEL@AI.CS.KOBE-U.AC.JP
Kobe University
1-1 Rokkodai, Nada
Kobe 657-8501 – Japan

Peter Prettenhofer PETER.PRETTENHOFER@GMAIL.COM
Bauhaus-Universität Weimar
Bauhausstr. 11
99421 Weimar – Germany

Ron Weiss RONWEISS@GMAIL.COM
Google Inc
76 Ninth Avenue
New York, NY 10011 – USA

Vincent Dubourg VINCENT.DUBOURG@GMAIL.COM
Clermont Université, IFMA, EA 3867, LaMI
BP 10448, 63000 Clermont-Ferrand – France

Jake Vanderplas VANDERPLAS@ASTRO.WASHINGTON.EDU
Astronomy Department
University of Washington, Box 351580
Seattle, WA 98195 – USA

Alexandre Passos ALEXANDRE.TP@GMAIL.COM
IESL Lab
UMass Amherst
Amherst MA 01002 – USA

David Cournapeau COURNAPE@GMAIL.COM
Enthought
21 J.J. Thompson Avenue
Cambridge, CB3 0FA – UK

Matthieu Brucher MATTHIEU.BRUCHER@GMAIL.COM
Total SA, CSTJF
avenue Larribau
64000 Pau – France

Matthieu Perrot MATTHIEU.PERROT@CEA.FR
Édouard Duchesnay EDOUARD.DUCHESNAY@CEA.FR
LNAO
Neurospin, Bât 145, CEA Saclay
91191 Gif sur Yvette – France

Editor: Mikio Braun

Abstract

Scikit-learn is a Python module integrating a wide range of state-of-the-art machine learning algorithms for medium-scale supervised and unsupervised problems. This package focuses on bringing machine learning to non-specialists using a general-purpose high-level language. Emphasis is put on ease of use, performance, documentation, and API consistency. It has minimal dependencies and is distributed under the simplified BSD license, encouraging its use in both academic and commercial settings. Source code, binaries, and documentation can be downloaded from `http://scikit-learn.sourceforge.net`.

Keywords: Python, supervised learning, unsupervised learning, model selection

1. Introduction

The Python programming language is establishing itself as one of the most popular languages for scientific computing. Thanks to its high-level interactive nature and its maturing ecosystem of scientific libraries, it is an appealing choice for algorithmic development and exploratory data analysis (Dubois, 2007; Milmann and Avaizis, 2011). Yet, as a general-purpose language, it is increasingly used not only in academic settings but also in industry.

Scikit-learn harnesses this rich environment to provide state-of-the-art implementations of many well known machine learning algorithms, while maintaining an easy-to-use interface tightly integrated with the Python language. This answers the growing need for statistical data analysis by non-specialists in the software and web industries, as well as in fields outside of computer-science, such as biology or physics. *Scikit-learn* differs from other machine learning toolboxes in Python for various reasons: *i)* it is distributed under the BSD license *ii)* it incorporates compiled code for efficiency, unlike MDP (Zito et al., 2008) and pybrain (Schaul et al., 2010), *iii)* it depends only on numpy and scipy to facilitate easy distribution, unlike pymvpa (Hanke et al., 2009) that has optional dependencies such as R and shogun, and *iv)* it focuses on imperative programming, unlike pybrain which uses a data-flow framework. While the package is mostly written in Python, it incorporates the C++ libraries LibSVM (Chang and Lin, 2001) and LibLinear (Fan et al., 2008) that provide reference implementations of SVMs and generalized linear models with compatible licenses. Binary packages are available on a rich set of platforms including Windows and any POSIX platforms.

Furthermore, thanks to its liberal license, it has been widely distributed as part of major free software distributions such as Ubuntu, Debian, Mandriva, NetBSD and Macports and in commercial distributions such as the "Enthought Python Distribution".

2. Project Vision

Code quality. Rather than providing as many features as possible, the project's goal has been to provide solid implementations. Code quality is ensured with unit tests—as of release 0.8, test coverage is 81%—and the use of static analysis tools such as `pyflakes` and `pep8`. Finally, we strive to use consistent naming for the functions and parameters used throughout a strict adherence to the Python coding guidelines and numpy style documentation.

BSD licensing. Most of the Python ecosystem is licensed with non-copyleft licenses. While such policy is beneficial for adoption of these tools by commercial projects, it does impose some restrictions: we are unable to use some existing scientific code, such as the GSL.

Bare-bone design and API. To lower the barrier of entry, we avoid framework code and keep the number of different objects to a minimum, relying on numpy arrays for data containers.

Community-driven development. We base our development on collaborative tools such as git, github and public mailing lists. External contributions are welcome and encouraged.

Documentation. *Scikit-learn* provides a ∼300 page user guide including narrative documentation, class references, a tutorial, installation instructions, as well as more than 60 examples, some featuring real-world applications. We try to minimize the use of machine-learning jargon, while maintaining precision with regards to the algorithms employed.

3. Underlying Technologies

Numpy: the base data structure used for data and model parameters. Input data is presented as numpy arrays, thus integrating seamlessly with other scientific Python libraries. Numpy's view-based memory model limits copies, even when binding with compiled code (Van der Walt et al., 2011). It also provides basic arithmetic operations.

Scipy: efficient algorithms for linear algebra, sparse matrix representation, special functions and basic statistical functions. *Scipy* has bindings for many Fortran-based standard numerical packages, such as LAPACK. This is important for ease of installation and portability, as providing libraries around Fortran code can prove challenging on various platforms.

Cython: a language for combining C in Python. Cython makes it easy to reach the performance of compiled languages with Python-like syntax and high-level operations. It is also used to bind compiled libraries, eliminating the boilerplate code of Python/C extensions.

4. Code Design

Objects specified by interface, not by inheritance. To facilitate the use of external objects with *scikit-learn*, inheritance is not enforced; instead, code conventions provide a consistent interface. The central object is an `estimator`, that implements a `fit` method, accepting as arguments an input data array and, optionally, an array of labels for supervised problems. Supervised estimators, such as SVM classifiers, can implement a `predict` method. Some estimators, that we call `transformers`, for example, PCA, implement a `transform` method, returning modified input data. Estimators

	scikit-learn	mlpy	pybrain	pymvpa	mdp	shogun
Support Vector Classification	**5.2**	9.47	17.5	11.52	40.48	5.63
Lasso (LARS)	**1.17**	105.3	-	37.35	-	-
Elastic Net	**0.52**	73.7	-	1.44	-	-
k-Nearest Neighbors	0.57	1.41	-	**0.56**	0.58	1.36
PCA (9 components)	**0.18**	-	-	8.93	0.47	0.33
k-Means (9 clusters)	1.34	0.79	$\star$	-	35.75	**0.68**
License	BSD	GPL	BSD	BSD	BSD	GPL

-: Not implemented. $\star$: Does not converge within 1 hour.

Table 1: Time in seconds on the Madelon data set for various machine learning libraries exposed in Python: MLPy (Albanese et al., 2008), PyBrain (Schaul et al., 2010), pymvpa (Hanke et al., 2009), MDP (Zito et al., 2008) and Shogun (Sonnenburg et al., 2010). For more benchmarks see `http://github.com/scikit-learn`.

may also provide a `score` method, which is an increasing evaluation of goodness of fit: a log-likelihood, or a negated loss function. The other important object is the *cross-validation iterator*, which provides pairs of train and test indices to split input data, for example K-fold, leave one out, or stratified cross-validation.

Model selection. *Scikit-learn* can evaluate an estimator's performance or select parameters using cross-validation, optionally distributing the computation to several cores. This is accomplished by wrapping an estimator in a `GridSearchCV` object, where the "CV" stands for "cross-validated". During the call to `fit`, it selects the parameters on a specified parameter grid, maximizing a score (the `score` method of the underlying estimator). `predict`, `score`, or `transform` are then delegated to the tuned estimator. This object can therefore be used transparently as any other estimator. Cross validation can be made more efficient for certain estimators by exploiting specific properties, such as warm restarts or regularization paths (Friedman et al., 2010). This is supported through special objects, such as the `LassoCV`. Finally, a `Pipeline` object can combine several `transformers` and an estimator to create a combined estimator to, for example, apply dimension reduction before fitting. It behaves as a standard estimator, and `GridSearchCV` therefore tune the parameters of all steps.

5. High-level yet Efficient: Some Trade Offs

While *scikit-learn* focuses on ease of use, and is mostly written in a high level language, care has been taken to maximize computational efficiency. In Table 1, we compare computation time for a few algorithms implemented in the major machine learning toolkits accessible in Python. We use the Madelon data set (Guyon et al., 2004), 4400 instances and 500 attributes, The data set is quite large, but small enough for most algorithms to run.

SVM. While all of the packages compared call libsvm in the background, the performance of *scikit-learn* can be explained by two factors. First, our bindings avoid memory copies and have up to 40% less overhead than the original libsvm Python bindings. Second, we patch libsvm to improve efficiency on dense data, use a smaller memory footprint, and better use memory alignment and pipelining capabilities of modern processors. This patched version also provides unique features, such as setting weights for individual samples.

LARS. Iteratively refining the residuals instead of recomputing them gives performance gains of 2–10 times over the reference R implementation (Hastie and Efron, 2004). *Pymvpa* uses this implementation via the Rpy R bindings and pays a heavy price to memory copies.
Elastic Net. We benchmarked the *scikit-learn* coordinate descent implementations of Elastic Net. It achieves the same order of performance as the highly optimized Fortran version *glmnet* (Friedman et al., 2010) on medium-scale problems, but performance on very large problems is limited since we do not use the KKT conditions to define an active set.
kNN. The k-nearest neighbors classifier implementation constructs a ball tree (Omohundro, 1989) of the samples, but uses a more efficient brute force search in large dimensions.
PCA. For medium to large data sets, *scikit-learn* provides an implementation of a truncated PCA based on random projections (Rokhlin et al., 2009).
k-means. *scikit-learn*'s k-means algorithm is implemented in pure Python. Its performance is limited by the fact that numpy's array operations take multiple passes over data.

6. Conclusion

Scikit-learn exposes a wide variety of machine learning algorithms, both supervised and unsupervised, using a consistent, task-oriented interface, thus enabling easy comparison of methods for a given application. Since it relies on the scientific Python ecosystem, it can easily be integrated into applications outside the traditional range of statistical data analysis. Importantly, the algorithms, implemented in a high-level language, can be used as building blocks for approaches specific to a use case, for example, in medical imaging (Michel et al., 2011). Future work includes *online* learning, to scale to large data sets.

References

D. Albanese, G. Merler, S.and Jurman, and R. Visintainer. MLPy: high-performance python package for predictive modeling. In *NIPS, MLOSS Workshop*, 2008.

C.C. Chang and C.J. Lin. LIBSVM: a library for support vector machines. `http://www.csie.ntu.edu.tw/cjlin/libsvm`, 2001.

P.F. Dubois, editor. *Python: Batteries Included*, volume 9 of *Computing in Science & Engineering*. IEEE/AIP, May 2007.

R.E. Fan, K.W. Chang, C.J. Hsieh, X.R. Wang, and C.J. Lin. LIBLINEAR: a library for large linear classification. *The Journal of Machine Learning Research*, 9:1871–1874, 2008.

J. Friedman, T. Hastie, and R. Tibshirani. Regularization paths for generalized linear models via coordinate descent. *Journal of Statistical Software*, 33(1):1, 2010.

I Guyon, S. R. Gunn, A. Ben-Hur, and G. Dror. Result analysis of the NIPS 2003 feature selection challenge, 2004.

M. Hanke, Y.O. Halchenko, P.B. Sederberg, S.J. Hanson, J.V. Haxby, and S. Pollmann. PyMVPA: A Python toolbox for multivariate pattern analysis of fMRI data. *Neuroinformatics*, 7(1):37–53, 2009.

T. Hastie and B. Efron. Least Angle Regression, Lasso and Forward Stagewise. `http://cran.r-project.org/web/packages/lars/lars.pdf`, 2004.

V. Michel, A. Gramfort, G. Varoquaux, E. Eger, C. Keribin, and B. Thirion. A supervised clustering approach for fMRI-based inference of brain states. *Patt Rec*, page epub ahead of print, April 2011. doi: 10.1016/j.patcog.2011.04.006.

K.J. Milmann and M. Avaizis, editors. *Scientific Python*, volume 11 of *Computing in Science & Engineering*. IEEE/AIP, March 2011.

S.M. Omohundro. Five balltree construction algorithms. ICSI Technical Report TR-89-063, 1989.

V. Rokhlin, A. Szlam, and M. Tygert. A randomized algorithm for principal component analysis. *SIAM Journal on Matrix Analysis and Applications*, 31(3):1100–1124, 2009.

T. Schaul, J. Bayer, D. Wierstra, Y. Sun, M. Felder, F. Sehnke, T. Rückstieß, and J. Schmidhuber. PyBrain. *The Journal of Machine Learning Research*, 11:743–746, 2010.

S. Sonnenburg, G. Rätsch, S. Henschel, C. Widmer, J. Behr, A. Zien, F. de Bona, A. Binder, C. Gehl, and V. Franc. The SHOGUN machine learning toolbox. *Journal of Machine Learning Research*, 11:1799–1802, 2010.

S. Van der Walt, S.C Colbert, and G. Varoquaux. The NumPy array: A structure for efficient numerical computation. *Computing in Science and Engineering*, 11, 2011.

T. Zito, N. Wilbert, L. Wiskott, and P. Berkes. Modular toolkit for data processing (MDP): A Python data processing framework. *Frontiers in Neuroinformatics*, 2, 2008.

Journal of Machine Learning Research 12 (2011) 2831-2855 Submitted 2/11; Revised 7/11; Published 10/11

Neyman-Pearson Classification, Convexity and Stochastic Constraints

Philippe Rigollet RIGOLLET@PRINCETON.EDU
Xin Tong XTONG@PRINCETON.EDU
Department of Operations Research and Financial Engineering
Princeton University
Princeton, NJ 08544, USA

Editor: Gábor Lugosi

Abstract

Motivated by problems of anomaly detection, this paper implements the Neyman-Pearson paradigm to deal with asymmetric errors in binary classification with a convex loss φ. Given a finite collection of classifiers, we combine them and obtain a new classifier that satisfies simultaneously the two following properties with high probability: (i) its φ-type I error is below a pre-specified level and (ii), it has φ-type II error close to the minimum possible. The proposed classifier is obtained by minimizing an empirical convex objective with an empirical convex constraint. The novelty of the method is that the classifier output by this computationally feasible program is shown to satisfy the original constraint on type I error. New techniques to handle such problems are developed and they have consequences on chance constrained programming. We also evaluate the price to pay in terms of type II error for being conservative on type I error.

Keywords: binary classification, Neyman-Pearson paradigm, anomaly detection, empirical constraint, empirical risk minimization, chance constrained optimization

1. Introduction

The Neyman-Pearson (NP) paradigm in statistical learning extends the objective of classical binary classification in that, while the latter focuses on minimizing classification error that is a weighted sum of type I and type II errors, the former minimizes type II error with an upper bound α on type I error. With slight abuse of language, in verbal discussion we do not distinguish type I/II error from probability of type I/II error.

For learning with the NP paradigm, it is essential to avoid one kind of error at the expense of the other. As an illustration, consider the following problem in medical diagnosis: failing to detect a malignant tumor has far more severe consequences than flagging a benign tumor. So it makes sense to put priority on controlling the false negative rate. Other scenarios include spam filtering, machine monitoring, target recognition, etc.

In the learning context, as true errors are inaccessible, we cannot enforce almost surely the desired upper bound for type I error. The best we can hope is that a data dependent classifier has type I error bounded with high probability. Ideally, a good classification rule $\hat{f}$ in NP context should satisfy two properties. The first is that type I error of the classifier $\hat{f}$ is bounded from above by a pre-specified level with pre-specified high probability; the second is that $\hat{f}$ has good performance bounds for excess type II error. As will be illustrated, it is unlikely that these two goals can be fulfilled simultaneously. Following the original spirit of NP paradigm, we put priority on type I error and insist on the pre-specified upper bound α. Our proposed learning procedure meets the

conservative attitude on type I error, and has good performance bound measured by the excess φ-type II error. We also discuss the general consequence of being conservative in NP learning.

The paper is organized as follows. In Section 2, the classical setup for binary classification is reviewed and the main notation is introduced. A parallel between binary classification and statistical hypothesis testing is drawn in Section 3 with emphasis on the NP paradigm in both frameworks. The main propositions and theorems are stated in Section 4 while proofs and technical results are relegated to Appendix A. Finally, Section 5 illustrates an application of our results to chance constrained optimization.

In the rest of the paper, we denote by x_j the j-th coordinate of a vector $x \in \mathbb{R}^d$.

2. Binary Classification

In this section, we review the classical setup of binary classification together with the convexification procedure that we employ throughout the paper. Moreover, we introduce the Neyman-Pearson paradigm in this setup.

2.1 Classification Risk and Classifiers

Let (X,Y) be a random couple where $X \in \mathcal{X} \subset \mathbb{R}^d$ is a vector of covariates and $Y \in \{-1,1\}$ is a label that indicates to which class X belongs. A *classifier* h is a mapping $h : \mathcal{X} \to [-1,1]$ whose sign returns the predicted class given X. An error occurs when $-h(X)Y \geq 0$ and it is therefore natural to define the classification loss by $\mathbb{1}(-h(X)Y \geq 0)$, where $\mathbb{1}(\cdot)$ denotes the indicator function.

The expectation of the classification loss with respect to the joint distribution of (X,Y) is called *(classification) risk* and is defined by

$$R(h) = \mathbb{P}\left(-h(X)Y \geq 0\right).$$

Clearly the indicator function is not convex, and for computational convenience, a common practice is to replace it by a convex surrogate (see, e.g., Bartlett et al., 2006, and references therein).

To this end, we rewrite the risk function as

$$R(h) = \mathbb{E}[\varphi(-h(X)Y)],$$

where $\varphi(z) = \mathbb{1}\left(z \geq 0\right)$. Convex relaxation can be achieved by simply replacing the indicator function by a convex surrogate.

Definition 1 *A function $\varphi : [-1,1] \to \mathbb{R}^+$ is called a* convex surrogate *if it is non-decreasing, continuous and convex and if $\varphi(0) = 1$.*

Commonly used examples of convex surrogates are the hinge loss $\varphi(x) = (1+x)_+$, the logit loss $\varphi(x) = \log_2(1+e^x)$ and the exponential loss $\varphi(x) = e^x$.

For a given choice of φ, define the φ-risk

$$R_\varphi(h) = \mathbb{E}[\varphi(-Yh(X))]\,.$$

Hereafter, we assume that φ is fixed and refer to R_φ as the risk when there is no confusion. In our subsequent analysis, this convex relaxation will also be the ground to analyze a stochastic convex

optimization problem subject to stochastic constraints. A general treatment of such problems can be found in Section 5.

Because of overfitting, it is unreasonable to look for mappings minimizing empirical risk over all classifiers. Indeed, one could have a small empirical risk but a large true risk. Hence, we resort to regularization. There are in general two ways to proceed. The first is to restrict the candidate classifiers to a specific class $\mathcal{H}$, and the second is to change the objective function by, for example, adding a penalty term. The two approaches can be combined, and sometimes they are obviously equivalent.

In this paper, we pursue the first idea by defining the class of candidate classifiers as follows. Let $h_1,\dots,h_M, M \geq 2$ be a given collection of classifiers. In our setup, we allow M to be large. In particular, our results remain asymptotically meaningful as long as $M = o(e^n)$. Such classifiers are usually called base classifiers and can be constructed in a very naive manner. Typical examples include decision stumps or small trees. While the h_j's may have no satisfactory classifying power individually, for over two decades, boosting type of algorithms have successfully exploited the idea that a suitable weighted majority vote among these classifiers may result in low classification risk (Schapire, 1990). Consequently, we restrict our search for classifiers to the set of functions consisting of convex combinations of the h_j's:

$$\mathcal{H}^{\text{conv}} = \{\mathsf{h}_\lambda = \sum_{j=1}^{M} \lambda_j h_j, \lambda \in \Lambda\},$$

where Λ denotes the flat simplex of $\mathbb{R}^M$ and is defined by $\Lambda = \{\lambda \in \mathbb{R}^M : \lambda_j \geq 0, \sum_{j=1}^M \lambda_j = 1\}$. In effect, classification rules given by the sign of $h \in \mathcal{H}^{\text{conv}}$ are exactly the set of rules produced by the weighted majority votes among the base classifiers $h_1,\dots,h_M$.

By restricting our search to classifiers in $\mathcal{H}^{\text{conv}}$, the best attainable φ-risk is called *oracle risk* and is abusively denoted by $R_\varphi(\mathcal{H}^{\text{conv}})$. As a result, we have $R_\varphi(h) \geq R_\varphi(\mathcal{H}^{\text{conv}})$ for any $h \in \mathcal{H}^{\text{conv}}$ and a natural measure of performance for a classifier $h \in \mathcal{H}^{\text{conv}}$ is given by its excess risk defined by $R_\varphi(h) - R_\varphi(\mathcal{H}^{\text{conv}})$.

The excess risk of a data driven classifier h_n is a random quantity and we are interested in bounding it with high probability. Formally, the statistical goal of binary classification is to construct a classifier h_n such that the oracle inequality

$$R_\varphi(h_n) \leq R_\varphi(h_{\mathcal{H}^{\text{conv}}}) + \Delta_n(\mathcal{H}^{\text{conv}}, \delta)$$

holds with probability $1-\delta$, where $\Delta_n(\cdot,\cdot)$ should be as small as possible.

In the scope of this paper, we focus on candidate classifiers in the class $\mathcal{H}^{\text{conv}}$. Some of the following results such as Theorem 3 can be extended to more general classes of classifiers with known complexity such as classes with bounded VC-dimension, as for example in Cannon et al. (2002). However, our main argument for bounding φ-type II error (defined in next subsection) relies on Proposition 4 which, in turn, depends heavily on the convexity of the problem, and it is not clear how it can be extended to more general classes of classifiers.

2.2 The Neyman-Pearson Paradigm

We make the convention that when $h(X) \geq 0$ the predicted class is $+1$, and -1 otherwise. Under this convention, the risk function in classical binary classification can be expressed as a convex combination of type I error $R^-(h) = \mathbb{P}(-Yh(X) \geq 0 | Y = -1)$ and type II error

$R^+(h) = \mathbb{P}(-Yh(X) > 0 | Y = 1)$:

$$R(h) = \mathbb{P}(Y = -1)R^-(h) + \mathbb{P}(Y = 1)R^+(h).$$

While the goal of classical binary classification is $\min_{h\in\mathcal{H}} R(h)$, where $\mathcal{H}$ is the set of candidate classifiers, the NP classification targets on

$$\min_{\substack{h\in\mathcal{H} \\ R^-(h)\le\alpha}} R^+(h)\,.$$

More generally, we can define the φ-type I and φ-type II errors respectively by

$$R^-_\varphi(h) = \mathbb{E}\left[\varphi(-Yh(X))|Y=-1\right] \qquad \text{and} \qquad R^+_\varphi(h) = \mathbb{E}\left[\varphi(-Yh(X))|Y=1\right].$$

Our main theorems concern about $R^-_\varphi(\cdot)$ and $R^+_\varphi(\cdot)$, but we will come back and address how convexification and conservativeness affect $R^-(\cdot)$ and $R^+(\cdot)$.

Following the NP paradigm, for a given class $\mathcal{H}$ of classifiers, we seek to solve the constrained minimization problem:

$$\min_{\substack{h\in\mathcal{H} \\ R^-_\varphi(h)\le\alpha}} R^+_\varphi(h), \tag{1}$$

where $\alpha \in (0,1)$, the significance level, is a constant specified by the user.

NP classification is closely related to the NP approach to statistical hypothesis testing. We now recall a few key concepts about the latter. Many classical works have addressed the theory of statistical hypothesis testing, in particular Lehmann and Romano (2005) provides a thorough treatment of the subject.

Statistical hypothesis testing bears strong resemblance with binary classification if we assume the following model. Let P^- and P^+ be two probability distributions on $\mathcal{X} \subset \mathbb{R}^d$. Let $p \in (0,1)$ and assume that Y is a random variable defined by

$$Y = \begin{cases} 1 & \text{with probability } p\,, \\ -1 & \text{with probability } 1-p\,. \end{cases}$$

Assume further that the conditional distribution of X given Y is given by P^Y. Given such a model, the goal of statistical hypothesis testing is to determine whether X was generated from P^- or P^+. To that end, we construct a test $\phi : \mathcal{X} \to [0,1]$ and the conclusion of the test based on ϕ is that X is generated from P^+ with probability $\phi(X)$ and from P^- with probability $1-\phi(X)$. Note that randomness here comes from an exogenous randomization process such as flipping a biased coin. Two kinds of errors arise: type I error occurs when rejecting P^- when it is true, and type II error occurs when accepting P^- when it is false. The Neyman-Pearson paradigm in hypothesis testing amounts to choosing ϕ that solves the following constrained optimization problem

$$\begin{aligned} \text{maximize} \quad & \mathbb{E}[\phi(X)|Y=1]\,, \\ \text{subject to} \quad & \mathbb{E}[\phi(X)|Y=-1] \le \alpha\,, \end{aligned}$$

where $\alpha \in (0,1)$ is the significance level of the test. In other words, we specify a significance level α on type I error, and minimize type II error. We call a solution to this problem *a most powerful test* of level α. The Neyman-Pearson Lemma gives mild sufficient conditions for the existence of such a test.

Theorem 2 (Neyman-Pearson Lemma) *Let P^- and P^+ be probability distributions possessing densities p^- and p^+ respectively with respect to some measure μ. Let $f_k(x) = \mathbb{1}(L(x) \geq k)$, where the likelihood ratio $L(x) = p^+(x)/p^-(x)$ and k is such that $P^-(L(X) > k) \leq \alpha$ and $P^-(L(X) \geq k) \geq \alpha$. Then,*

- *f_k is a level $\alpha = \mathbb{E}[\varphi_k(X)|Y = -1]$ most powerful test.*
- *For a given level α, the most powerful test of level α is defined by*

$$\phi(X) = \begin{cases} 1 & \text{if} \quad L(X) > k \\ 0 & \text{if} \quad L(X) < k \\ \frac{\alpha - P^-(L(X)>k)}{P^-(L(X)=k)} & \text{if} \quad L(X) = k\,. \end{cases}$$

Notice that in the learning framework, ϕ cannot be computed since it requires the knowledge of the likelihood ratio and of the distributions P^- and P^+. Therefore, it remains merely a theoretical proposition. Nevertheless, the result motivates the NP paradigm pursued here.

3. Neyman-Pearson Classification Via Convex Optimization

Recall that in NP classification with a loss function φ, the goal is to solve the problem (1). This cannot be done directly as conditional distributions P^- and P^+, and hence R^-_φ and R^+_φ, are unknown. In statistical applications, information about these distributions is available through two i.i.d. samples $X^-_1, \ldots, X^-_{n^-}, n^- \geq 1$ and $X^+_1, \ldots, X^+_{n^+}, n^+ \geq 1$, where $X^-_i \sim P^-, i = 1, \ldots, n^-$ and $X^+_i \sim P^+, i = 1, \ldots, n^+$. We do not assume that the two samples $(X^-_1, \ldots, X^-_{n^-})$ and $(X^+_1, \ldots, X^+_{n^+})$ are mutually independent. Presently the sample sizes n^- and n^+ are assumed to be deterministic and will appear in the subsequent finite sample bounds. A different sampling scheme, where these quantities are random, is investigated in Section 4.3.

3.1 Conservativeness on Type I Error

While the binary classification problem has been extensively studied, theoretical proposition on how to implement the NP paradigm remains scarce. To the best of our knowledge, Cannon et al. (2002) initiated the theoretical treatment of the NP classification paradigm and an early empirical study can be found in Casasent and Chen (2003). The framework of Cannon et al. (2002) is the following. Fix a constant $\varepsilon_0 > 0$ and let $\mathcal{H}$ be a given set of classifiers with finite VC dimension. They study a procedure that consists of solving the following relaxed empirical optimization problem

$$\min_{\substack{h \in \mathcal{H} \\ \hat{R}^-(h) \leq \alpha + \varepsilon_0/2}} \hat{R}^+(h), \tag{2}$$

where

$$\hat{R}^-(h) = \frac{1}{n^-} \sum_{i=1}^{n^-} \mathbb{1}(h(X^-_i) \geq 0)\,, \quad \text{and} \quad \hat{R}^+(h) = \frac{1}{n^+} \sum_{i=1}^{n^+} \mathbb{1}(h(X^-_i) \leq 0)$$

denote the empirical type I and empirical type II errors respectively. Let $\hat{h}$ be a solution to (2). Denote by h^* a solution to the original Neyman-Pearson optimization problem:

$$h^* \in \operatorname*{argmin}_{\substack{h \in \mathcal{H} \\ R^-(h) \leq \alpha}} R^+(h)\,, \tag{3}$$

The main result of Cannon et al. (2002) states that, simultaneously with high probability, the type II error $R^+(\hat{h})$ is bounded from above by $R^+(h^*)+\varepsilon_1$, for some $\varepsilon_1 > 0$ and the type I error of $\hat{h}$ is bounded from above by $\alpha+\varepsilon_0$. In a later paper, Cannon et al. (2003) considers problem (2) for a data-dependent family of classifiers $\mathcal{H}$, and bound estimation errors accordingly. Several results for traditional statistical learning such as PAC bounds or oracle inequalities have been studied in Scott (2005) and Scott and Nowak (2005) in the same framework as the one laid down by Cannon et al. (2002). A noteworthy departure from this setup is Scott (2007) where sensible performance measures for NP classification that go beyond analyzing separately two kinds of errors are introduced. Furthermore, Blanchard et al. (2010) develops a general solution to semi-supervised novelty detection by reducing it to NP classification. Recently, Han et al. (2008) transposed several results of Cannon et al. (2002) and Scott and Nowak (2005) to NP classification with convex loss.

The present work departs from previous literature in our treatment of type I error. In fact, the classifiers in all the papers mentioned above take a compromise on the pre-determined upper bound on type I error, that is, they ensure that $\mathbb{P}(R^-(\hat{h}) > \alpha+\varepsilon_0)$ is small, for some $\varepsilon_0 > 0$. However, it is our primary interest to make sure that $R^-(\hat{h}) \le \alpha$ with high probability, following the original principle of the Neyman-Pearson paradigm that type I error should be controlled by a pre-specified level α. As we follow an empirical risk minimization procedure, to control $\mathbb{P}(R^-(\hat{h}) > \alpha)$, it is necessary to have $\hat{h}$ be a solution to some program with a strengthened constraint on empirical type I error. If our concern is only on type I error, we can just do so. However, we also want to evaluate the excess type II error. Our conservative attitude on type I error faces new technical challenges which we summarize here. In the approach of Cannon et al. (2002) and of Scott and Nowak (2005), the relaxed constraint on the type I error is constructed such that the constraint $\hat{R}^-(h) \le \alpha+\varepsilon_0/2$ on type I error in (2) is satisfied by h^* (defined in (3)) with high probability, and that this classifier accommodates the excess type II error well. As a result, the control of type II error mainly follows as a standard exercise to control suprema of empirical processes. This is not the case here; we have to develop methods to control the optimum value of an optimization problem under a stochastic constraint. Such methods have consequences not only in NP classification but also on chance constraint programming as explained in Section 5.

3.2 Convexified NP Classifier

Concerned about computational feasibility, our proposed classifier is the solution to a convex program, which is an empirical form NP classification problem (1) where the distribution of the observations is unknown. In view of the arguments presented in the previous subsection, we cannot simply replace the unknown risk functions by their empirical counterparts. The treatment of the convex constraint should be done carefully and we proceed as follows.

For any classifier h and a given convex surrogate φ, define $\hat{R}^-_\varphi$ and $\hat{R}^+_\varphi$ to be the empirical counterparts of R^-_φ and R^+_φ respectively by

$$\hat{R}^-_\varphi(h) = \frac{1}{n^-}\sum_{i=1}^{n^-} \varphi(h(X_i^-))\,, \quad \text{and} \quad \hat{R}^+_\varphi(h) = \frac{1}{n^+}\sum_{i=1}^{n^+} \varphi(-h(X_i^+))\,.$$

Moreover, for any $a > 0$, let $\mathcal{H}^{\varphi,a} = \{h \in \mathcal{H}^{\text{conv}} : R^-_\varphi(h) \le a\}$ be the set of classifiers in $\mathcal{H}^{\text{conv}}$ whose convexified type I errors are bounded from above by a, and let $\mathcal{H}^{\varphi,a}_{n^-} = \{h \in \mathcal{H}^{\text{conv}} : \hat{R}^-_\varphi(h) \le a\}$ be the set of classifiers in $\mathcal{H}^{\text{conv}}$ whose empirical convexified type I errors are bounded by a. To make our analysis meaningful, we assume that $\mathcal{H}^{\varphi,\alpha} \neq \emptyset$.

We are now in a position to construct a classifier in $\mathcal{H}^{\text{conv}}$ according to the Neyman-Pearson paradigm. For any $\tau > 0$ such that $\tau \leq \alpha\sqrt{n^-}$, define the convexified NP classifier $\tilde{h}^\tau$ as any classifier that solves the following optimization problem

$$\min_{\substack{h \in \mathcal{H}^{\text{conv}} \\ \hat{R}_\varphi^-(h) \leq \alpha - \tau/\sqrt{n^-}}} \hat{R}_\varphi^+(h). \tag{4}$$

Note that this problem consists of minimizing a convex function subject to a convex constraint and can therefore be solved by standard algorithms (see, e.g., Boyd and Vandenberghe, 2004, and references therein).

In the next section, we present a series of results on type I and type II errors of classifiers that are more general than $\tilde{h}^\tau$.

4. Performance Bounds

In this section, we will first evaluate our proposed classifier $\tilde{h}^\tau$ against φ I/II errors. These benchmarks are necessary because $\tilde{h}^\tau$ is constructed based on them. Moreover, in view of the decision theory framework, such errors are just expected loss with a general loss function φ, which are interesting to investigate. As the true type I and type II errors are usually the main concern in statistical learning, we will also address the effect of convexification in terms of the excess type II error. Interestingly, given that we want to be conservative on type I error, neither working on φ errors nor working on true errors leads to a most desirable type II error. The price to pay for being conservative will be characterized explicitly.

4.1 Control of Type I Error

First, we identify classifiers h such that $R_\varphi^-(h) \leq \alpha$ with high probability. This is done by enforcing its empirical counterpart $\hat{R}_\varphi^-(h)$ be bounded from above by the quantity

$$\alpha_\kappa = \alpha - \kappa/\sqrt{n^-},$$

for a proper choice of positive constant κ.

Theorem 3 *Fix constants $\delta, \alpha \in (0,1), L > 0$ and let $\varphi : [-1,1] \to \mathbb{R}^+$ be a given L-Lipschitz convex surrogate. Define*

$$\kappa = 4\sqrt{2}L\sqrt{\log\left(\frac{2M}{\delta}\right)}.$$

Then for any (random) classifier $h \in \mathcal{H}^{\text{conv}}$ that satisfies $\hat{R}_\varphi^-(h) \leq \alpha_\kappa$, we have

$$R^-(h) \leq R_\varphi^-(h) \leq \alpha.$$

with probability at least $1-\delta$. Equivalently

$$\mathbb{P}\left[\mathcal{H}_{n^-}^{\varphi,\alpha_\kappa} \subset \mathcal{H}^{\varphi,\alpha}\right] \geq 1 - \delta. \tag{5}$$

4.2 Simultaneous Control of the Two Errors

Theorem 3 guarantees that any classifier that satisfies the strengthened constraint on the empirical φ-type I error will have φ-type I error and true type I error bounded from above by α. We now check that the constraint is not too strong so that the φ-type II error is overly deteriorated. Indeed, an extremely small α_κ would certainly ensure a good control of type I error but would deteriorate significantly the best achievable φ-type II error. Below, we show not only that this is not the case for our approach but also that the convexified NP classifier $\tilde{h}^\tau$ defined in Section 3.2 with $\tau = \alpha_\kappa$ suffers only a small degradation of its φ-type II error compared to the best achievable. Analogous to classical binary classification, a desirable result is that with high probability,

$$R_\varphi^+(\tilde{h}^\kappa) - \min_{h \in \mathcal{H}^{\varphi,\alpha}} R_\varphi^+(h) \le \tilde{\Delta}_n(\mathcal{F}), \tag{6}$$

where $\tilde{\Delta}_n(\mathcal{F})$ goes to 0 as $n = n^- + n^+ \to \infty$.

The following proposition is pivotal to our argument.

Proposition 4 *Fix constant $\alpha \in (0,1)$ and let $\varphi : [-1,1] \to \mathbb{R}^+$ be a given continuous convex surrogate. Assume further that there exists $\nu_0 > 0$ such that the set of classifiers $\mathcal{H}^{\varphi,\alpha-\nu_0}$ is nonempty. Then, for any $\nu \in (0,\nu_0)$,*

$$\min_{h \in \mathcal{H}^{\varphi,\alpha-\nu}} R_\varphi^+(h) - \min_{h \in \mathcal{H}^{\varphi,\alpha}} R_\varphi^+(h) \le \varphi(1) \frac{\nu}{\nu_0 - \nu}\,.$$

This proposition ensures that if the convex surrogate φ is continuous, strengthening the constraint on type I error (φ-type I error) does not increase too much the best achievable φ-type II error. We should mention that the proof does not use the Lipschitz property of φ, but only that it is uniformly bounded by $\varphi(1)$ on $[-1,1]$. This proposition has direct consequences on chance constrained programming as discussed in Section 5.

The next theorem shows that the NP classifier $\tilde{h}^\kappa$ defined in Section 3.2 is a good candidate to perform classification with the Neyman-Pearson paradigm. It relies on the following assumption which is necessary to verify the condition of Proposition 4.

Assumption 1 *There exists a positive constant $\varepsilon < 1$ such that the set of classifiers $\mathcal{H}^{\varphi,\varepsilon\alpha}$ is nonempty.*

Note that this assumption can be tested using (5) for large enough n^-. Indeed, it follows from this inequality that with probability $1-\delta$,

$$\mathcal{H}_{n^-}^{\varphi,\varepsilon\alpha-\kappa/\sqrt{n^-}} \subset \mathcal{H}^{\varphi,\varepsilon\alpha-\kappa/\sqrt{n^-}+\kappa/\sqrt{n^-}} = \mathcal{H}^{\varphi,\varepsilon\alpha}\,.$$

Thus, it is sufficient to check if $\mathcal{H}_{n^-}^{\varphi,\varepsilon\alpha-\kappa/\sqrt{n^-}}$ is nonempty for some $\varepsilon > 0$. Before stating our main theorem, we need the following definition. Under Assumption 1, let ε denote the smallest ε such that $\mathcal{H}^{\varphi,\varepsilon\alpha} \neq \emptyset$ and let n_0 be the smallest integer such that

$$n_0 \ge \left(\frac{4\kappa}{(1-\varepsilon)\alpha}\right)^2. \tag{7}$$

Theorem 5 *Let* φ, κ, δ *and* α *be the same as in Theorem 3, and* $\tilde{h}^\kappa$ *denote any solution to* (4). *Moreover, let Assumption 1 hold and assume that* $n^- \geq n_0$ *where* n_0 *is defined in* (7). *Then, the following hold with probability* $1-2\delta$,

$$R^-(\tilde{h}^\kappa) \leq R^-_\varphi(\tilde{h}^\kappa) \leq \alpha \tag{8}$$

and

$$R^+_\varphi(\tilde{h}^\kappa) - \min_{h\in\mathcal{H}^{\varphi,\alpha}} R^+_\varphi(h) \leq \frac{4\varphi(1)\kappa}{(1-\varepsilon)\alpha\sqrt{n^-}} + \frac{2\kappa}{\sqrt{n^+}}\,. \tag{9}$$

In particular, there exists a constant $C>0$ *depending on* α, $\varphi(1)$ *and* ε, *such that* (9) *yields*

$$R^+_\varphi(\tilde{h}^\kappa) - \min_{h\in\mathcal{H}^{\varphi,\alpha}} R^+_\varphi(h) \leq C\left(\sqrt{\frac{\log(2M/\delta)}{n^-}} + \sqrt{\frac{\log(2M/\delta)}{n^+}}\right).$$

Note here that Theorem 4.2 is not exactly of the type (6). The right hand side of (9) goes to zero if both n^- and n^+ go to infinity. Inequality (9) conveys a message that accuracy of the estimate depends on information from both classes of labeled data. This concern motivates us to consider a different sampling scheme.

4.3 A Different Sampling Scheme

In this subsection (only), we consider a model for observations that is more standard in statistical learning theory (see, e.g., Devroye et al., 1996; Boucheron et al., 2005).

Let $(X_1,Y_1),\ldots,(X_n,Y_n)$ be n independent copies of the random couple $(X,Y)\in\mathcal{X}\times\{-1,1\}$. Denote by P_X the marginal distribution of X and by $\eta(x)=\mathbb{E}[Y|X=x]$ the regression function of Y onto X. Denote by p the probability of positive label and observe that

$$p = \mathbb{P}[Y=1] = \mathbb{E}\left(\mathbb{P}[Y=1|X]\right) = \frac{1+\mathbb{E}[\eta(X)]}{2}\,.$$

In what follows, we assume that $P_X(\eta(X)=-1)\vee P_X(\eta(X)=1)<1$ so that $p\in(0,1)$.

Let $N^-=\mathrm{card}\{Y_i : Y_i=-1\}$ be the random number of instances labeled -1 and $N^+=n-N^-=\mathrm{card}\{Y_i : Y_i=1\}$. In this setup, the NP classifier is defined as in Section 3.2 where n^- and n^+ are replaced by N^- and N^+ respectively. To distinguish this classifier from $\tilde{h}^\tau$ previously defined, we denote the NP classifier obtained with this sampling scheme by $\tilde{h}^\tau_n$.

Let the event $\mathcal{F}$ be defined by

$$\mathcal{F} = \{R^-_\varphi(\tilde{h}^\kappa_n)\leq\alpha\}\cap\{R^+_\varphi(\tilde{h}^\kappa_n) - \min_{h\in\mathcal{H}^{\varphi,\alpha}} R^+_\varphi(h) \leq \frac{4\varphi(1)\kappa}{(1-\varepsilon)\alpha\sqrt{N^-}} + \frac{2\kappa}{\sqrt{N^+}}\}.$$

Denote $\mathcal{B}_{n^-}=\{Y_1=\cdots=Y_{n^-}=-1, Y_{n^-+1}=\cdots=Y_n=1\}$. Although the event $\mathcal{B}_{n^-}$ is different from the event $\{N^-=n^-\}$, symmetry leads to the following key observation:

$$\mathbb{P}(\mathcal{F}|N^-=n^-) = \mathbb{P}(\mathcal{F}|\mathcal{B}_{n^-}).$$

Therefore, under the conditions of Theorem 5, we find that for $n^-\geq n_0$ the event $\mathcal{F}$ satisfies

$$\mathbb{P}(\mathcal{F}|N^-=n^-)\geq 1-2\delta\,. \tag{10}$$

We obtain the following corollary of Theorem 5.

Corollary 6 *Let* φ, κ, δ *and* α *be the same as in Theorem 3, and* $\tilde{h}_n^\kappa$ *be the NP classifier obtained with the current sampling scheme. Then under Assumption 1, if* $n > 2n_0/(1-p)$, *where* n_0 *is defined in* (7), *we have with probability* $(1-2\delta)(1-e^{-\frac{n(1-p)^2}{2}})$,

$$R^-(\tilde{h}_n^\kappa) \le R_\varphi^-(\tilde{h}_n^\kappa) \le \alpha \tag{11}$$

and

$$R_\varphi^+(\tilde{h}_n^\kappa) - \min_{h \in \mathcal{H}^{\varphi,\alpha}} R_\varphi^+(h) \le \frac{4\varphi(1)\kappa}{(1-\varepsilon)\alpha\sqrt{N^-}} + \frac{2\kappa}{\sqrt{N^+}}\,. \tag{12}$$

Moreover, with probability $1-2\delta-e^{-\frac{n(1-p)^2}{2}}-e^{-\frac{np^2}{2}}$, *we have simultaneously* (11) *and*

$$R_\varphi^+(\tilde{h}_n^\kappa) - \min_{h \in \mathcal{H}^{\varphi,\alpha}} R_\varphi^+(h) \le \frac{4\sqrt{2}\varphi(1)\kappa}{(1-\varepsilon)\alpha\sqrt{n(1-p)}} + \frac{2\sqrt{2}\kappa}{\sqrt{np}}\,. \tag{13}$$

4.4 Price to Pay For Being Conservative

We have shown that the the computational feasible classifier $\tilde{h}^\kappa$ satisfies oracle inequalities which take the optimal φ-type II errors as the benchmark. In this subsection, the excess type II error will be measured, and we will characterize the price to pay by being conservative on type I error.

Much like its counterparts in classical binary classification, the next strikingly simple relation addresses the consequence of convexification in the NP paradigm.

Theorem 7 *Let* $\tilde{h}$ *be any classifier, then*

$$R^+(\tilde{h}) - \min_{R^-(h)\le\alpha} R^+(h) \le R_\varphi^+(\tilde{h}) - \inf_{R^-(h)\le\alpha} R_\varphi^+(h)\,.$$

This theorem applies to any classifier; in particular, it holds for our proposed $\tilde{h}^\kappa$. As the proof of Theorem 7 indicates, $\min_{R^-(h)\le\alpha} R^+(h) = \inf_{R^-(h)\le\alpha} R_\varphi^+(h)$. So the bound in the theorem can be very tight, depending on the nature of $\tilde{h}$.

Now relax the range of base classifiers $h_1, \ldots, h_M$ to be $[-B,B]$. Denote by $\mathcal{H}_B^{\varphi,\alpha}$ the set of convex combinations of the base classifiers that have φ-type I error bounded from above by α.

Therefore, we have the following observation:

$$R^+(\tilde{h}^\kappa) - \min_{R^-(h)\le\alpha} R^+(h) \le T_1 + T_2 + T_3\,,$$

where

$$\begin{aligned}
T_1 &= R_\varphi^+(\tilde{h}^\kappa) - \min_{h \in \mathcal{H}_B^{\varphi,\alpha}} R_\varphi^+(h)\,,\\
T_2 &= \min_{h \in \mathcal{H}_B^{\varphi,\alpha}} R_\varphi^+(h) - \inf_{\substack{R^-(h)\le\alpha\\ -B\le h\le B}} R_\varphi^+(h)\,,\\
T_3 &= \inf_{\substack{R^-(h)\le\alpha\\ -B\le h\le B}} R_\varphi^+(h) - \inf_{R^-(h)\le\alpha} R_\varphi^+(h)\,.
\end{aligned}$$

With the new set of base classifiers taking ranges in $[-B,B]$, Theorem 5 holds if we replace κ by $\kappa_B = 4\sqrt{2}L_B B\sqrt{\log(2M/\delta)}$, where L_B is the Lipschitz constant of φ on $[-B,B]$. Therefore, the

convergence rate of T_1 is explicitly controlled. We can see that with a fixed sample size, choosing a set of base classifiers with smaller range will result in a tighter bound for the excess φ-type II error. However, if one concerns more about the true type II error, choosing a smaller B should not be a better option, because only signs matters for true type I and II errors. This intuition is reflected in the term T_3. When B increases, T_3 decreases. More specifically, it can be shown that

$$T_3 = \big(P^+(\mathcal{X}^+)\varphi(-B) + P^+(\mathcal{X}^-)\varphi(0)\big) - P^+(\mathcal{X}^-) = P^+(\mathcal{X}^+)\varphi(-B)\,,$$

where $\mathcal{X}^+ \subset \mathcal{X}$ is the part of feature space mapped to label $+1$ by the optimal NP classifier that solves $\min_{R^-(h)\le\alpha} R^+(h)$, and $\mathcal{X}^-$ is the part that mapped to label -1; this is what NP Lemma says when there is no need for randomization. Therefore, T_3 diminishes towards 0 as B increases, and the trade-off between T_1 and T_3 is very clear. When $\varphi(x) = (1+x)_+$ is the hinge loss, the best trade-off occurs at $B \in (0,1)$. When $B(\ge 1)$ goes to infinity, $T_3 = 0$ stays the same while the upper bound of T_1 blows up.

Note that $\mathcal{H}_B^{\varphi,\alpha} \subset \{h : R^-(h) \le \alpha, -B \le h \le B\}$, so T_2 reflects the price to pay for being conservative on type I error. It also reflect the bias for choosing a specific candidate pool of classifiers, that is, convex combinations of base classifiers. As long as the base classifiers are rich enough, the latter bias should be small. However in our belief, the price to pay for being conservative is unavoidable. Even if we do not resort to convexification, getting the best insurance on type I error still demands a high premium on type II error.

The same attitude is shared in the seminal paper Cannon et al. (2002), where it was claimed without justification that if we use $\alpha' < \alpha$ for the empirical program, "it seems unlikely that we can control the estimation error $R^+(\hat{h}) - R^+(h^*)$ in a distribution independent way". The following proposition confirms this opinion in a certain sense.

Fix $\alpha \in (0,1), n^- \ge 1, n^+ \ge 1$ and $\alpha' < \alpha$. Let $\hat{h}(\alpha')$ be the classifier defined as any solution of the following optimization problem:

$$\min_{\substack{h\in\mathcal{H}\\ \hat{R}^-(h)\le\alpha'}} \hat{R}^+(h)\,.$$

The following negative result holds not only for this estimator but also for the oracle $h^*(\alpha')$ defined as the solution of

$$\min_{\substack{h\in\mathcal{H}\\ R^-(h)\le\alpha'}} R^+(h)\,.$$

Note that $h^*(\alpha')$ is not a classifier but only a pseudo-classifier since it depends on the unknown distribution of the data.

Proposition 8 *There exist base classifiers h_1, h_2 and a probability distribution for (X,Y) for which, regardless of the sample sizes n^- and n^+, any pseudo-classifier $h_{\tilde{\lambda}} = \tilde{\lambda}h_1 + (1-\tilde{\lambda})h_2$, $0 \le \tilde{\lambda} \le 1$, such that $R^-(h_{\tilde{\lambda}}) < \alpha$, it holds*

$$R^+(h_{\tilde{\lambda}}) - \min_{R^-(h_\lambda)\le\alpha, \lambda\in[0,1]} R^+(h_\lambda) \ge \alpha\,.$$

In particular, the excess type II risk of $h^*(\alpha - \varepsilon_{n^-}), \varepsilon_{n^-} > 0$ does not converge to zero as sample sizes increase even if $\varepsilon_{n^-} \to 0$. Moreover, when $\alpha \le 1/2$ for any (pseudo-)classifier $h_{\tilde{\lambda}}$ ($0 \le \tilde{\lambda} \le 1$) such that $\hat{R}^-(h_{\tilde{\lambda}}) < \alpha$, it holds

$$R^+(h_{\tilde{\lambda}}) - \min_{R^-(h_\lambda)\le\alpha, \lambda\in[0,1]} R^+(h_\lambda) \ge \alpha\,.$$

with probability at least $\alpha \wedge 1/4$. *In other words, if we let* $\mathcal{A} = \{h_\lambda : \hat{R}^-(h_\lambda) < \alpha, \lambda \in [0,1]\}$, *and* $\mathcal{B} = \{h_\lambda : R^+(h_\lambda) - \min_{R^-(h_\lambda) \leq \alpha, \lambda \in [0,1]} R^+(h_\lambda) \geq \alpha, \lambda \in [0,1]\}$, *then* $\mathbb{P}(\mathcal{A} \subset \mathcal{B}) \geq \alpha \wedge 1/4$. *In particular, the excess type II risk of* $\hat{h}(\alpha - \varepsilon_{n^-})$, $\varepsilon_{n^-} > 0$ *does not converge to zero with positive probability, as sample sizes increase even if* $\varepsilon_{n^-} \to 0$.

The proof of this result is postponed to Appendix A. The fact that the oracle $h^*(\alpha - \varepsilon_{n^-})$ satisfies the lower bound indicates that the problem comes from using a strengthened constraint. Note that the condition $\alpha \leq 1/2$ is purely technical and can be removed. Nevertheless, it is always the case in practice that $\alpha \leq 1/2$. When the number of base classifiers is great then two, we believe that similar counterexamples can be still constructed, though the technicality will be move involved.

In view of this negative result and our previous discussion, we have to accept the price to pay for being conservative on type I error, and our classifier $\tilde{h}^\kappa$ is no exception. As such conservativeness follows from the original spirit of the Neyman-Pearson paradigm, we need to pay whatever we have to pay. The positive sides are that our proposed procedure is computationally feasible, and it attains good rates under a different (but still meaningful) criterion.

5. Chance Constrained Optimization

Implementing the Neyman-Pearson paradigm for the convexified binary classification bears strong connections with chance constrained optimization. A recent account of such problems can be found in Ben-Tal et al. (2009, Chapter 2) and we refer to this book for references and applications. A chance constrained optimization problem is of the following form:

$$\min_{\lambda \in \Lambda} f(\lambda) \quad \text{s.t.} \quad \mathbb{P}\{F(\lambda, \xi) \leq 0\} \geq 1 - \alpha, \tag{14}$$

where $\xi \in \Xi$ is a random vector, $\Lambda \subset \mathbb{R}^M$ is convex, α is a small positive number and f is a deterministic real valued convex function. Problem (14) can be viewed as a relaxation of robust optimization. Indeed, for the latter, the goal is to solve the problem

$$\min_{\lambda \in \Lambda} f(\lambda) \quad \text{s.t.} \quad \sup_{\xi \in \Xi} F(\lambda, \xi) \leq 0\,, \tag{15}$$

and this essentially corresponds to (14) for the case $\alpha = 0$. For simplicity, we take F to be scalar valued but extensions to vector valued functions and conic orders are considered in Ben-Tal et al. (2009, Chapter 10). Moreover, it is standard to assume that $F(\cdot, \xi)$ is convex almost surely.

Problem (14) may not be convex because the chance constraint $\{\lambda \in \Lambda : \mathbb{P}\{F(\lambda, \xi) \leq 0\} \geq 1 - \alpha\}$ is not convex in general and thus may not be tractable. To solve this problem, Prékopa (1995) and Lagoa et al. (2005) have derived sufficient conditions on the distribution of ξ for the chance constraint to be convex. On the other hand, Calafiore and Campi (2006) initiated a different treatment of the problem where no assumption on the distribution of ξ is made, in line with the spirit of statistical learning. In that paper, they introduced the so-called *scenario approach* based on a sample $\xi_1, \ldots, \xi_n$ of independent copies of ξ. The scenario approach consists of solving

$$\min_{\lambda \in \Lambda} f(\lambda) \quad \text{s.t.} \quad F(\lambda, \xi_i) \leq 0, i = 1, \ldots, n. \tag{16}$$

Calafiore and Campi (2006) showed that under certain conditions, if the sample size n is bigger than some $n(\alpha, \delta)$, then with probability $1 - \delta$, the optimal solution $\hat{\lambda}^{sc}$ of (16) is feasible for (14). The

authors did not address the control of the term $f(\hat{\lambda}^{sc}) - f^*$ where f^* denotes the optimal objective value in (14). However, in view of Proposition 8, it is very unlikely that this term can be controlled well.

In an attempt to overcome this limitation, a new *analytical* approach was introduced by Nemirovski and Shapiro (2006). It amounts to solving the following convex optimization problem

$$\min_{\lambda\in\Lambda, t\in\mathbb{R}^s} f(\lambda) \quad \text{s.t.} \quad G(\lambda,t) \le 0, \tag{17}$$

in which t is some additional instrumental variable and where $G(\cdot,t)$ is convex. The problem (17) provides a conservative convex approximation to (14), in the sense that every x feasible for (17) is also feasible for (14). Nemirovski and Shapiro (2006) considered a particular class of conservative convex approximation where the key step is to replace $\mathbb{P}\{F(\lambda,\xi) \ge 0\}$ by $\mathbb{E}\varphi(F(\lambda,\xi))$ in (14), where φ a nonnegative, nondecreasing, convex function that takes value 1 at 0. Nemirovski and Shapiro (2006) discussed several choices of φ including hinge and exponential losses, with a focus on the latter that they name *Bernstein Approximation*.

The idea of a conservative convex approximation is also what we employ in our paper. Recall that P^- the conditional distribution of X given $Y = -1$. In a parallel form of (14), we cast our target problem as

$$\min_{\lambda\in\Lambda} R^+(h_\lambda) \quad \text{s.t.} \quad P^-\{h_\lambda(X) \le 0\} \ge 1-\alpha, \tag{18}$$

where Λ is the flat simplex of $\mathbb{R}^M$.

Problem (18) differs from (14) in that $R^+(h_\lambda)$ is not a convex function of λ. Replacing $R^+(h_\lambda)$ by $R^+_\varphi(h_\lambda)$ turns (18) into a standard chance constrained optimization problem:

$$\min_{\lambda\in\Lambda} R^+_\varphi(h_\lambda) \quad \text{s.t.} \quad P^-\{h_\lambda(X) \le 0\} \ge 1-\alpha. \tag{19}$$

However, there are two important differences in our setting, so that we cannot use directly Scenario Approach or Bernstein Approximation or other analytical approaches to (14). First, $R^+_\varphi(f_\lambda)$ is an *unknown* function of λ. Second, we assume minimum knowledge about P^-. On the other hand, chance constrained optimization techniques in previous literature assume knowledge about the distribution of the random vector ξ. For example, Nemirovski and Shapiro (2006) require that the moment generating function of the random vector ξ is efficiently computable to study the Bernstein Approximation.

Given a finite sample, it is not feasible to construct a strictly conservative approximation to the constraint in (19). On the other hand, it is possible to ensure that if we learned $\hat{h}$ from the sample, this constraint is satisfied with high probability $1-\delta$, that is, the classifier is approximately feasible for (19). In retrospect, our approach to (19) is an innovative hybrid between the analytical approach based on convex surrogates and the scenario approach.

We do have structural assumptions on the problem. Let g_j, $j \in \{1,\dots,M\}$ be arbitrary functions that take values in $[-1,1]$ and $F(\lambda,\xi) = \sum_{j=1}^N \lambda_j g_j(\xi)$. Consider a convexified version of (14):

$$\min_{\lambda\in\Lambda} f(\lambda) \quad \text{s.t.} \quad \mathbb{E}[\varphi(F(\lambda,\xi))] \le \alpha, \tag{20}$$

where φ is a L-Lipschitz convex surrogate, $L > 0$. Suppose that we observe a sample $(\xi_1,\dots,\xi_n)$ that are independent copies of ξ. We propose to approximately solve the above problem by

$$\min_{\lambda\in\Lambda} f(\lambda) \quad \text{s.t.} \quad \sum_{i=1}^n \varphi(F(\lambda,\xi_i)) \le n\alpha - \kappa\sqrt{n},$$

for some $\kappa >$ to be defined. Denote by $\tilde{\lambda}$ any solution to this problem and by f_φ^* the value of the objective at the optimum in (20). The following theorem summarizes our contribution to chance constrained optimization.

Theorem 9 *Fix constants* $\delta, \alpha \in (0,1/2), L > 0$ *and let* $\varphi : [-1,1] \to \mathbb{R}^+$ *be a given L-Lipschitz convex surrogate. Define*

$$\kappa = 4\sqrt{2}L\sqrt{\log\left(\frac{2M}{\delta}\right)}.$$

Then, the following hold with probability at least $1-2\delta$

(i) $\tilde{\lambda}$ *is feasible for* (14).

(ii) If there exists $\varepsilon \in (0,1)$ *such that the constraint* $\mathbb{E}[\varphi(F(\lambda,\xi))] \le \varepsilon\alpha$ *is feasible for some* $\lambda \in \Lambda$*, then for*

$$n \ge \left(\frac{4\kappa}{(1-\varepsilon)\alpha}\right)^2,$$

we have

$$f(\tilde{\lambda}) - f_\varphi^* \le \frac{4\varphi(1)\kappa}{(1-\varepsilon)\alpha\sqrt{n}}.$$

In particular, as M and n go to infinity with all other quantities kept fixed, we obtain

$$f(\tilde{\lambda}) - f_\varphi^* = O\left(\sqrt{\frac{\log M}{n}}\right).$$

The proof essentially follows that of Theorem 5 and we omit it. The limitations of Theorem 9 include rigid structural assumptions on the function F and on the set Λ. While the latter can be easily relaxed using more sophisticated empirical process theory, the former is inherent to our analysis.

Acknowledgments

Philippe Rigollet is supported by the National Science Foundation (DMS-0906424 & DMS-1053987).

Appendix A. Proof of the Main Results

We gather in this appendix the proofs of the main results of the paper.

A.1 Proof of Theorem 3

We begin with the following lemma, which is extensively used in the sequel. Its proof relies on standard arguments to bound suprema of empirical processes. Recall that $\{h_1, \dots, h_M\}$ is family of M classifiers such that $h_j : \mathcal{X} \to [-1,1]$ and that for any λ in the simplex $\Lambda \subset R^M$, h_λ denotes the convex combination defined by

$$\mathsf{h}_\lambda = \sum_{j=1}^N \lambda_j h_j.$$

The following standard notation in empirical process theory will be used. Let $X_1,\dots,X_n \in \mathcal{X}$ be n i.i.d random variables with marginal distribution P. Then for any measurable function $f : \mathcal{X} \to \mathbb{R}$, we write

$$P_n(f) = \frac{1}{n}\sum_{i=1}^{n} f(X_i) \qquad \text{and} \qquad P(f) = \mathbb{E}f(X) = \int f \mathrm{d}P.$$

Moreover, the Rademacher average of f is defined as

$$R_n(f) = \frac{1}{n}\sum_{i=1}^{n} \varepsilon_i f(X_i)\,,$$

where $\varepsilon_1,\dots,\varepsilon_n$ are i.i.d. Rademacher random variables such that $\mathbb{P}(\varepsilon_i = 1) = \mathbb{P}(\varepsilon_i = -1) = 1/2$ for $i = 1,\dots,n$.

Lemma 10 *Fix $L > 0, \delta \in (0,1)$. Let $X_1,\dots,X_n$ be n i.i.d random variables on $\mathcal{X}$ with marginal distribution P. Moreover, let $\varphi : [-1,1] \to \mathbb{R}$ an L-Lipschitz function. Then, with probability at least $1-\delta$, it holds*

$$\sup_{\lambda\in\Lambda} |(P_n - P)(\varphi \circ h_\lambda)| \le \frac{4\sqrt{2}L}{\sqrt{n}} \sqrt{\log\left(\frac{2M}{\delta}\right)}\,.$$

Proof Define $\varphi(\cdot) \doteq \varphi(\cdot) - \varphi(0)$, so that φ is an L-Lipschitz function that satisfies $\varphi(0) = 0$. Moreover, for any $\lambda \in \Lambda$, it holds

$$(P_n - P)(\varphi \circ \mathsf{h}_\lambda) = (P_n - P)(\varphi \circ \mathsf{h}_\lambda)\,.$$

Let $\Phi : \mathbb{R} \to \mathbb{R}_+$ be a given convex increasing function. Applying successively the symmetrization and the contraction inequalities (see, e.g., Koltchinskii, 2011, Chapter 2), we find

$$\mathbb{E}\Phi\left(\sup_{\lambda\in\Lambda} |(P_n - P)(\varphi \circ \mathsf{h}_\lambda)|\right) \le \mathbb{E}\Phi\left(2\sup_{\lambda\in\Lambda} |R_n(\varphi \circ \mathsf{h}_\lambda)|\right) \le \mathbb{E}\Phi\left(4L\sup_{\lambda\in\Lambda} |R_n(\mathsf{h}_\lambda)|\right).$$

Observe now that $\lambda \mapsto |R_n(\mathsf{h}_\lambda)|$ is a convex function and Theorem 32.2 in Rockafellar (1997) entails that

$$\sup_{\lambda\in\Lambda} |R_n(\mathsf{h}_\lambda)| = \max_{1\le j\le M} \left|R_n(h_j)\right|.$$

We now use a Chernoff bound to control this quantity. To that end, fix $s,t > 0$, and observe that

$$\begin{aligned}
\mathbb{P}\left(\sup_{\lambda\in\Lambda} |(P_n - P)(\varphi \circ \mathsf{h}_\lambda)| > t\right) &\le \frac{1}{\Phi(st)}\mathbb{E}\Phi\left(s\sup_{\lambda\in\Lambda} |(P_n - P)(\varphi \circ \mathsf{h}_\lambda)|\right) \\
&\le \frac{1}{\Phi(st)}\mathbb{E}\Phi\left(4Ls\max_{1\le j\le M} \left|R_n(h_j)\right|\right). \qquad (21)
\end{aligned}$$

Moreover, since Φ is increasing,

$$\begin{aligned}
\mathbb{E}\Phi\left(4Ls\max_{1\le j\le M} \left|R_n(h_j)\right|\right) &= \mathbb{E}\max_{1\le j\le M} \Phi\left(4Ls\left|R_n(h_j)\right|\right) \\
&\le \sum_{j=1}^{M} \mathbb{E}\left[\Phi\left(4LsR_n(h_j)\right) \vee \Phi\left(-4LsR_n(h_j)\right)\right] \\
&\le 2\sum_{j=1}^{M} \mathbb{E}\Phi\left(4LsR_n(h_j)\right). \qquad (22)
\end{aligned}$$

Now choose $\Phi(\cdot) = \exp(\cdot)$, then

$$\mathbb{E}\Phi\left(4LsR_n(h_j)\right) = \prod_{i=1}^{n} \mathbb{E}\cosh\left(\frac{4Lsh_j(X_i)}{n}\right) \leq \exp\left(\frac{8L^2s^2}{n}\right),$$

where cosh is the hyperbolic cosine function and where in the inequality, we used the fact that $|h_j(X_i)| \leq 1$ for any i, j and $\cosh(x) \leq \exp(x^2/2)$. Together with (21) and (22), it yields

$$\mathbb{P}\left(\sup_{\lambda\in\Lambda} |(P_n - P)(\varphi \circ \mathsf{h}_\lambda)| > t\right) \leq 2M \inf_{s>0} \exp\left(\frac{8L^2s^2}{n} - st\right) \leq 2M \exp\left(-\frac{nt^2}{32L^2}\right).$$

Choosing

$$t = \frac{4\sqrt{2}L}{\sqrt{n}} \sqrt{\log\left(\frac{2M}{\delta}\right)},$$

completes the proof of the Lemma. ■

We now proceed to the proof of Theorem 3. Note first that from the properties of φ, $R^-(h) \leq R^-_\varphi(h)$. Next, we have for any data-dependent classifier $h \in \mathcal{H}^{\text{conv}}$ such that $\hat{R}^-_\varphi(h) \leq \alpha_\kappa$:

$$R^-_\varphi(h) \leq \hat{R}^-_\varphi(h) + \sup_{h\in\mathcal{H}^{\text{conv}}} \left|\hat{R}^-_\varphi(h) - R^-_\varphi(h)\right| \leq \alpha - \frac{\kappa}{\sqrt{n^-}} + \sup_{h\in\mathcal{H}^{\text{conv}}} \left|\hat{R}^-_\varphi(h) - R^-_\varphi(h)\right|.$$

Lemma 10 implies that, with probability $1-\delta$

$$\sup_{h\in\mathcal{H}^{\text{conv}}} \left|\hat{R}^-_\varphi(h) - R^-_\varphi(h)\right| = \sup_{\lambda\in\Lambda} \left|(P^-_{n^-} - P^-)(\varphi \circ \mathsf{h}_\lambda)\right| \leq \frac{\kappa}{\sqrt{n^-}}.$$

The previous two displays imply that $R^-_\varphi(h) \leq \alpha$ with probability $1-\delta$, which completes the proof of Theorem 3.

A.2 Proof of Proposition 4

The proof of this proposition builds upon the following lemma.

Lemma 11 *Let* $\gamma(\alpha) = \inf_{\mathsf{h}_\lambda \in \mathcal{H}^{\varphi,\alpha}} R^+_\varphi(\mathsf{h}_\lambda)$, *then* γ *is a non-increasing convex function on* $[0,1]$.

Proof First, it is clear that γ is a non-increasing function of α because for $\alpha' > \alpha$, $\{\mathsf{h}_\lambda \in \mathcal{H}^{\text{conv}} : R^-_\varphi(\mathsf{h}_\lambda) \leq \alpha\} \subset \{\mathsf{h}_\lambda \in \mathcal{H}^{\text{conv}} : R^-_\varphi(\mathsf{h}_\lambda) \leq \alpha'\}$.

We now show that γ is convex. To that end, observe first that since φ is continuous on $[-1,1]$, the set $\{\lambda \in \Lambda : \mathsf{h}_\lambda \in \mathcal{H}^{\varphi,\alpha}\}$ is compact. Moreover, the function $\lambda \mapsto R^+_\varphi(\mathsf{h}_\lambda)$ is convex. Therefore, there exists $\lambda^* \in \Lambda$ such that

$$\gamma(\alpha) = \inf_{\mathsf{h}_\lambda \in \mathcal{H}^{\varphi,\alpha}} R^+_\varphi(\mathsf{h}_\lambda) = \min_{\mathsf{h}_\lambda \in \mathcal{H}^{\varphi,\alpha}} R^+_\varphi(\mathsf{h}_\lambda) = R^+_\varphi(\mathsf{h}_{\lambda^*}).$$

Now, fix $\alpha_1, \alpha_2 \in [0,1]$. From the above considerations, there exist $\lambda_1, \lambda_2 \in \Lambda$ such that $\gamma(\alpha_1) = R^+_\varphi(\mathsf{h}_{\lambda_1})$ and $\gamma(\alpha_2) = R^+_\varphi(\mathsf{h}_{\lambda_2})$. For any $\theta \in (0,1)$, define the convex combinations $\alpha_\theta = \theta\alpha_1 + (1-\theta)\alpha_2$ and $\lambda_\theta = \theta\lambda_1 + (1-\theta)\lambda_2$. Since $\lambda \mapsto R^-_\varphi(\mathsf{h}_\lambda)$ is convex, it holds

$$R^-_\varphi(\mathsf{h}_{\lambda_\theta}) \leq \theta R^-_\varphi(\mathsf{h}_{\lambda_1}) + (1-\theta) R^-_\varphi(\mathsf{h}_{\lambda_2}) \leq \theta\alpha_1 + (1-\theta)\alpha_2 = \alpha_\theta,$$

so that $\mathsf{h}_{\lambda_\theta} \in \mathcal{H}^{\varphi,\alpha_\theta}$. Hence, $\gamma(\alpha_\theta) \le R^+_\varphi(\mathsf{h}_{\lambda_\theta})$. Together with the convexity of φ, it yields

$$\gamma(\theta\alpha_1 + (1-\theta)\alpha_2) \le R^+_\varphi(\mathsf{h}_{\lambda_\theta}) \le \theta R^+_\varphi(\mathsf{h}_{\lambda_1}) + (1-\theta)R^+_\varphi(\mathsf{h}_{\lambda_2}) = \theta\gamma(\alpha_1) + (1-\theta)\gamma(\alpha_2)\,.$$

■

We now complete the proof of Proposition 4. For any $x \in [0,1]$, let $\gamma(x) = \inf_{h\in\mathcal{H}^{\varphi,x}} R^+_\varphi(h)$ and observe that the statement of the proposition is equivalent to

$$\gamma(\alpha-\nu) - \gamma(\alpha) \le \varphi(1)\frac{\nu}{\nu_0-\nu}\,, \quad 0 < \nu < \nu_0\,.$$

Lemma 11 together with the assumption that $\mathcal{H}^{\varphi,\alpha-\nu_0} \neq \emptyset$ imply that γ is a non-increasing convex real-valued function on $[\alpha-\nu_0, 1]$ so that

$$\gamma(\alpha-\nu) - \gamma(\alpha) \le \nu \sup_{g\in\partial\gamma(\alpha-\nu)} |g|\,,$$

where $\partial\gamma(\alpha-\nu)$ denotes the sub-differential of γ at $\alpha-\nu$. Moreover, since γ is a non-increasing convex function on $[\alpha-\nu_0, \alpha-\nu]$, it holds

$$\gamma(\alpha-\nu_0) - \gamma(\alpha-\nu) \ge (\nu-\nu_0) \sup_{g\in\partial\gamma(\alpha-\nu)} |g|\,.$$

The previous two displays yield

$$\gamma(\alpha-\nu) - \gamma(\alpha) \le \nu\frac{\gamma(\alpha-\nu_0)-\gamma(\alpha-\nu)}{\nu-\nu_0} \le \nu\frac{\varphi(1)}{\nu-\nu_0}\,.$$

A.3 Proof of Theorem 5

Define the events $\mathcal{E}^-$ and $\mathcal{E}^+$ by

$$\mathcal{E}^- = \bigcap_{h\in\mathcal{H}^{\mathrm{conv}}} \{|\hat{R}^-_\varphi(h) - R^-_\varphi(h)| \le \frac{\kappa}{\sqrt{n^-}}\}\,,$$
$$\mathcal{E}^+ = \bigcap_{h\in\mathcal{H}^{\mathrm{conv}}} \{|\hat{R}^+_\varphi(h) - R^+_\varphi(h)| \le \frac{\kappa}{\sqrt{n^+}}\}\,.$$

Lemma 10 implies

$$\mathbb{P}(\mathcal{E}^-) \wedge \mathbb{P}(\mathcal{E}^+) \ge 1-\delta\,. \tag{23}$$

Note first that Theorem 3 implies that (8) holds with probability $1-\delta$. Observe now that the l.h.s of (9) can be decomposed as

$$R^+_\varphi(\tilde{h}^\kappa) - \min_{h\in\mathcal{H}^{\varphi,\alpha}} R^+_\varphi(h) = A_1 + A_1 + A_3\,,$$

where

$$A_1 = \left(R^+_\varphi(\tilde{h}^\kappa) - \hat{R}^+_\varphi(\tilde{h}^\kappa)\right) + \left(\hat{R}^+_\varphi(\tilde{h}^\kappa) - \min_{h\in\mathcal{H}^{\varphi,\alpha_\kappa}_{n^-}} R^+_\varphi(h)\right)$$
$$A_2 = \min_{h\in\mathcal{H}^{\varphi,\alpha_\kappa}_{n^-}} R^+_\varphi(h) - \min_{h\in\mathcal{H}^{\varphi,\alpha_{2\kappa}}} R^+_\varphi(h)$$
$$A_3 = \min_{h\in\mathcal{H}^{\varphi,\alpha_{2\kappa}}} R^+_\varphi(h) - \min_{h\in\mathcal{H}^{\varphi,\alpha}} R^+_\varphi(h).$$

To bound A_1 from above, observe that

$$A_1 \le 2 \sup_{h \in \mathcal{H}_{n^-}^{\varphi,\alpha_\kappa}} |\hat{R}_\varphi^+(h) - R_\varphi^+(h)| \le 2 \sup_{h \in \mathcal{H}^{\mathrm{conv}}} |\hat{R}_\varphi^+(h) - R_\varphi^+(h)|.$$

Therefore, on the event $\mathcal{E}^+$ it holds

$$A_1 \le \frac{2\kappa}{\sqrt{n^+}}.$$

We now treat A_2. Note that $A_2 \le 0$ on the event $\mathcal{H}^{\varphi,\alpha_{2\kappa}} \subset \mathcal{H}_{n^-}^{\varphi,\alpha_\kappa}$. But this event contains $\mathcal{E}^-$ so that $A_2 \le 0$ on the event $\mathcal{E}^-$.

Finally, to control A_3, observe that under Assumption 1, Proposition 4 can be applied with $\nu = 2\kappa/\sqrt{n^-}$ and $\nu_0 = (1-\varepsilon)\alpha$. Indeed, the assumptions of the theorem imply that $\nu \le \nu_0/2$. It yields

$$A_3 \le \frac{4\varphi(1)\kappa}{(1-\varepsilon)\alpha\sqrt{n^-}}.$$

Combining the bounds on A_1, A_2 and A_3 obtained above, we find that (9) holds on the event $\mathcal{E}^- \cap \mathcal{E}^+$ that has probability at least $1-2\delta$ in view of (23).

The last statement of the theorem follows directly from the definition of κ.

A.4 Proof of Corollary 6

Now prove (12),

$$\begin{aligned}
\mathbb{P}(\mathcal{F}) &= \sum_{n^-=0}^{n} \mathbb{P}(\mathcal{F}\,|N^- = n^-)\mathbb{P}(N^- = n^-) \\
&\ge \sum_{n^-=n_0}^{n} \mathbb{P}(\mathcal{F}\,|N^- = n^-)\mathbb{P}(N^- = n^-) \\
&\ge (1-2\delta)\mathbb{P}(N^- \ge n_0),
\end{aligned}$$

where in the last inequality, we used (10). Applying now Lemma 12, we obtain

$$\mathbb{P}(N^- \ge n_0) \ge 1 - e^{-\frac{n(1-p)^2}{2}}.$$

Therefore,

$$\mathbb{P}(\mathcal{F}) \ge (1-2\delta)(1-e^{-\frac{n(1-p)^2}{2}}),$$

which completes the proof of (12).

The proof of (13) follows by observing that

$$\left\{ R_\varphi^+(\tilde{h}_n^\kappa) - \min_{h \in \mathcal{H}^{\varphi,\alpha}} R_\varphi^+(h) > \frac{4\sqrt{2}\varphi(1)\kappa}{(1-\varepsilon)\alpha\sqrt{n(1-p)}} + \frac{2\sqrt{2}\kappa}{\sqrt{np}} \right\} \subset (\mathcal{A}_1 \cap \mathcal{A}_2^c) \cup \mathcal{A}_2 \cup \mathcal{A}_3,$$

where

$$\begin{aligned}
\mathcal{A}_1 &= \left\{ R_\varphi^+(\tilde{h}_n^\kappa) - \min_{h \in \mathcal{H}^{\varphi,\alpha}} R_\varphi^+(h) > \frac{4\varphi(1)\kappa}{(1-\varepsilon)\alpha\sqrt{N^-}} + \frac{2\kappa}{\sqrt{N^+}} \right\} \subset \mathcal{F}^c, \\
\mathcal{A}_2 &= \{N^- < n(1-p)/2\}, \\
\mathcal{A}_3 &= \{N^+ < np/2\}.
\end{aligned}$$

Since $\mathcal{A}_2^c \subset \{N^- \geq n_0\}$, we find

$$\mathbb{P}(\mathcal{A}_1 \cap \mathcal{A}_2^c) \leq \sum_{n^- \geq n_0} \mathbb{P}(\mathcal{F}^c | N^- = n^-)\mathbb{P}(N^- = n^-) \leq 2\delta\,.$$

Next, using Lemma 12, we get

$$\mathbb{P}(\mathcal{A}_2) \leq e^{-\frac{n(1-p)^2}{2}} \qquad \text{and} \qquad \mathbb{P}(\mathcal{A}_3) \leq e^{-\frac{np^2}{2}}\,.$$

Hence, we find

$$\mathbb{P}\left\{ R_\varphi^+(\tilde{h}_n^\kappa) - \min_{h \in \mathcal{H}^{\varphi,\alpha}} R_\varphi^+(h) > \frac{4\sqrt{2}\varphi(1)\kappa}{(1-\varepsilon)\alpha\sqrt{n(1-p)}} + \frac{2\sqrt{2}\kappa}{\sqrt{np}} \right\} \leq 2\delta + e^{-\frac{n(1-p)^2}{2}} + e^{-\frac{np^2}{2}}\,,$$

which completes the proof of the corollary.

A.5 Proof of Theorem 7

First observe that for any $\hat{h}$, $R^+(\hat{h}) \leq R_\varphi^+(\hat{h})$. Then the result follows from the claim that

$$\min_{R^-(h) \leq \alpha} R^+(h) = \inf_{R^-(h) \leq \alpha} R_\varphi^+(h)\,.$$

It is clear $\min_{R^-(h)\leq\alpha} R^+(h) \leq \inf_{R^-(h)\leq\alpha} R_\varphi^+(h)$, it remains to prove the other direction. By the Neyman-Pearson Lemma, We can decompose the feature space $\mathcal{X}$ into a disjoint union of $\mathcal{X}^+$ and $\mathcal{X}^-$, and the optimal (pseudo) classifier that solves $\min_{R^-(h)\leq\alpha} R^+(h)$ assigns label $+1$ for any $x \in \mathcal{X}^+$, and -1 for any $x \in \mathcal{X}^-$. Note that if any two classifiers g_1 and g_2 have the same signs, that is, $\operatorname{sgn}(g_1) = \operatorname{sgn}(g_2)$, then $R^-(g_1) = R^-(g_2)$ and $R^+(g_1) = R^+(g_2)$. On the other hand, for φ-type I and II errors, values of classifiers do matter.

Let $h_{B,\varepsilon}(x) = B \cdot \mathbb{I}(x \in \mathcal{X}^+) + (-\varepsilon) \cdot \mathbb{I}(x \in \mathcal{X}^-)$. Then clearly for any $B, \varepsilon > 0$, $h_{B,\varepsilon}$ solves $\min_{R^-(h)\leq\alpha} R^+(h)$. Also, for any $B, \varepsilon > 0$,

$$\inf_{R^-(h) \leq \alpha} R_\varphi^+(h) \leq R_\varphi^+(h_{B,\varepsilon}) = P^+(\mathcal{X}^+)\varphi(-B) + P^+(\mathcal{X}^-)\varphi(\varepsilon)\,.$$

Taking the limit, we have

$$\lim_{B\to\infty,\varepsilon\to 0} R_\varphi^+(h_{B,\varepsilon}) = \lim_{B\to\infty,\varepsilon\to 0} P^+(\mathcal{X}^+)\varphi(-B) + P^+(\mathcal{X}^-)\varphi(\varepsilon) = P^+(\mathcal{X}^-) = R^+(h_{B,\varepsilon})\,.$$

Therefore, $\inf_{R^-(h)\leq\alpha} R_\varphi^+(h) \leq \min_{R^-(h)\leq\alpha} R^+(h)$, which completes the proof.

A.6 Proof of Proposition 8

Let the base classifiers be defined as

$$h_1(x) = -1 \quad \text{and} \quad h_2(x) = \mathbb{1}(x \leq \alpha) - \mathbb{1}(x > \alpha)\,, \quad \forall x \in [0,1]$$

For any $\lambda \in [0,1]$, denote the convex combination of h_1 and h_2 by $\mathsf{h}_\lambda = \lambda h_1 + (1-\lambda)h_2$, that is,

$$\mathsf{h}_\lambda(x) = (1-2\lambda)\,\mathbb{1}(x \leq \alpha) - \mathbb{1}(x > \alpha)\,.$$

Suppose the conditional distributions of X given $Y=1$ or $Y=-1$, denoted respectively by P^+ and P^-, are both uniform on $[0,1]$. Recall that $R^-(\mathsf{h}_\lambda) = P^-(\mathsf{h}_\lambda(X) \geq 0)$ and $R^+(\mathsf{h}_\lambda) = P^+(\mathsf{h}_\lambda(X) < 0)$. Then, we have

$$R^-(\mathsf{h}_\lambda) = P^-(\mathsf{h}_\lambda(X) \geq 0) = \alpha \mathbb{I}(\lambda \leq 1/2). \tag{24}$$

Therefore, for any $\tau \in [0,\alpha]$, we have

$$\{\lambda \in [0,1] : R^-(\mathsf{h}_\lambda) \leq \tau\} = \begin{cases} [0,1] & \text{if } \tau = \alpha, \\ (1/2,1] & \text{if } \tau < \alpha. \end{cases}$$

Observe now that

$$R^+(\mathsf{h}_\lambda) = P^+(\mathsf{h}_\lambda(X) < 0) = (1-\alpha)\mathbb{I}(\lambda < 1/2) + \mathbb{I}(\lambda \geq 1/2). \tag{25}$$

For any $\tau \in [0,\alpha]$, it yields

$$\inf_{\lambda \in [0,1]: R^-(\mathsf{h}_\lambda) \leq \tau} R^+(\mathsf{h}_\lambda) = \begin{cases} 1-\alpha & \text{if } \tau = \alpha, \\ 1 & \text{if } \tau < \alpha. \end{cases}$$

Consider now a classifier h_λ such that $R^-(\mathsf{h}_\lambda) \leq \tau$ for some $\tau < \alpha$. Then from (24), we see that must have $\lambda > 1/2$. Together with (25), this imples that $R^+(\mathsf{h}_\lambda) = 1$. It yields

$$R^+(\mathsf{h}_\lambda) - \min_{\lambda : R^-(\mathsf{h}_\lambda) \leq \alpha} R^+(\mathsf{h}_\lambda) = 1 - (1-\alpha) = \alpha.$$

This completes the first part of the proposition. Moreover, in the same manner as (24), it can be easily proved that

$$\hat{R}^-(\mathsf{h}_\lambda) = \frac{1}{n^-} \sum_{i=1}^{n^-} \mathbb{I}(\mathsf{h}_\lambda(X_i^-) \geq 0) = \alpha_{n^-} \mathbb{I}(\lambda \leq 1/2), \tag{26}$$

where

$$\alpha_{n^-} = \frac{1}{n^-} \sum_{i=1}^{n^-} \mathbb{I}(X_i^- \leq \alpha) \tag{27}$$

If a classifier $\hat{\mathsf{h}}_\lambda$ is such that $\hat{R}^-(\hat{\mathsf{h}}_\lambda) < \alpha_{n^-}$, then (26) implies that $\lambda > 1/2$. Using again (25), we find also that $R^+(\hat{\mathsf{h}}_\lambda) = 1$. It yields

$$R^+(\hat{\mathsf{h}}_\lambda) - \min_{\lambda : R^-(\mathsf{h}_\lambda) \leq \alpha} R^+(\mathsf{h}_\lambda) = 1 - (1-\alpha) = \alpha.$$

It remains to show that $\hat{R}^-(\hat{\mathsf{h}}_\lambda) < \alpha_{n^-}$ with positive probability for any classifier such that $\hat{R}^-(\hat{\mathsf{h}}_\lambda) \leq \tau$ for some $\tau < \alpha$. Note that a sufficient condition for a classifier $\hat{\mathsf{h}}_\lambda$ to satisfy this constraint is to have $\alpha \leq \alpha_{n^-}$. It is therefore sufficient to find a lower bound on the probability of the event $\mathcal{A} = \{\alpha_{n^-} \geq \alpha\}$. Such a lower bound is provided by Lemma 13, which guarantees that $\mathbb{P}(\mathcal{A}) \geq \alpha \wedge 1/4$.

Appendix B. Technical Lemmas on Binomial Distributions

The following lemmas are purely technical on the tails of Binomial distributions.

Lemma 12 *Let N be a binomial random variables with parameters $n \geq 1$ and $q \in (0,1)$. Then, for any $t > 0$ such that $t \leq nq/2$, it holds*

$$\mathbb{P}(N \geq t) \geq 1 - e^{-\frac{nq^2}{2}}.$$

Proof Note first that $n - N$ has binomial distribution with parameters $n \geq 1$ and $1-q$. Therefore, we can write $n - N = \sum_{i=1}^{n} Z_i$ where Z_i are i.i.d. Bernoulli random variables with parameter $1-q$. Thus, using Hoeffding's inequality, we find that for any $s \geq 0$,

$$\mathbb{P}(n - N - n(1-q) \geq s) \leq e^{-\frac{2s^2}{n}}.$$

Applying the above inequality with $s = n - n(1-q) - t \geq nq/2 \geq 0$ yields

$$\mathbb{P}(N \geq t) = \mathbb{P}(n - N - n(1-q) \leq n - n(1-q) - t) \geq 1 - e^{-\frac{nq^2}{2}}.$$

■

The next lemma provides a lower bound on the probability that a binomial distribution exceeds its expectation. Our result is uniform in the size of the binomial and it can be easily verified that it is sharp by considering sizes $n = 1$ and $n = 2$ and by looking at Figure 1. In particular, we do not resort to Gaussian approximation which improves upon the lower bounds that can be derived from the inequalities presented in Slud (1977).

Lemma 13 *Let N be a binomial random variable with parameters $n \geq 1$ and $0 < q \leq 1/2$. Then, it holds*

$$\mathbb{P}(N \geq nq) \geq q \wedge (1/4).$$

Proof We introduce the following local definition, which is limited to the scope of this proof. Fix $n \geq 1$ and for any $q \in (0,1)$, let P_q denote the distribution of a binomial random variable with parameters n and q. Note first that if $n = 1$, the result is trivial since

$$P_q(N \geq q) = \mathbb{P}(Z \geq q) = \mathbb{P}(Z = 1) = q,$$

where Z is a Bernoulli random variable with parameter q.

Assume that $n \geq 2$. Note that if $q \leq 1/n$, then $P_q(N \geq nq) \geq \mathbb{P}(Z = 1) = q$, where Z is a Bernoulli random variable with parameter q. Moreover, for any any integer k such that $k/n < q \leq (k+1)/n$, we have

$$P_q(N \geq nq) = P_q(N \geq k+1) \geq P_{\frac{k}{n}}(N \geq k+1). \tag{28}$$

The above inequality can be easily proved by taking the derivative over the interval $(k/n, (k+1)/n]$, of the function

$$q \mapsto \sum_{j=k+1}^{n} \binom{n}{j} q^j (1-q)^{n-j}.$$

We now show that

$$P_{\frac{k}{n}}(N \geq k+1) \geq P_{\frac{k-1}{n}}(N \geq k), \quad 2 \leq k \leq n/2. \tag{29}$$

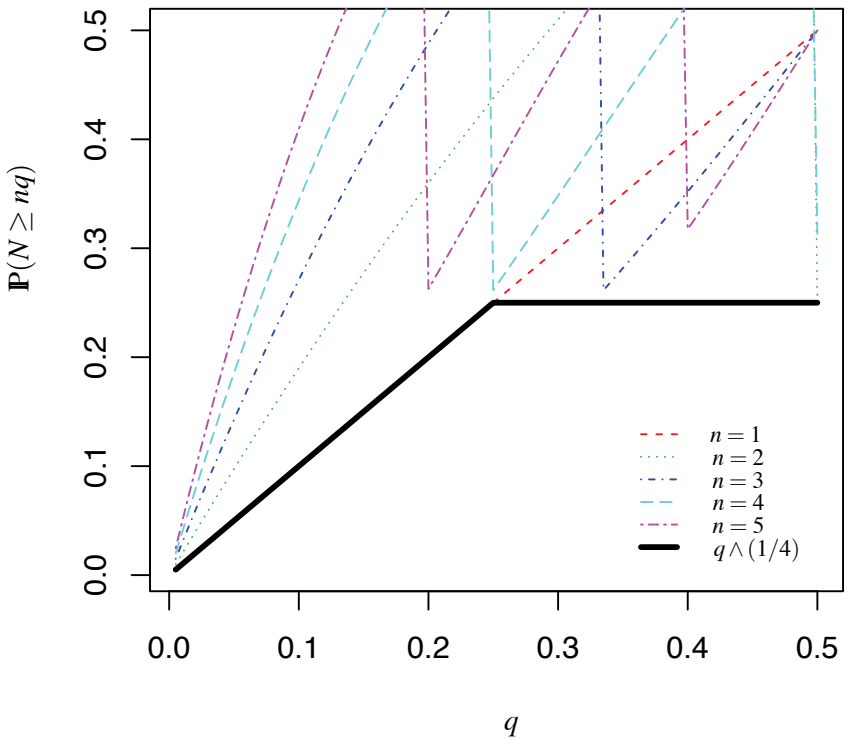


Figure 1: Tail probabilities $\mathbb{P}(N \geq nq)$ where N is a binomial random variable with parameters n and q.

Let $U_1, \ldots, U_n$ be n i.i.d. random variables uniformly distributed on the interval $[0,1]$ and denote by $U_{(k)}$ the corresponding kth order statistic such that $U_{(1)} \leq \ldots \leq U_{(n)}$. Following Feller (1971, Section 7.2), it is not hard to show that

$$P_{\frac{k}{n}}(N \geq k+1) = \mathbb{P}(U_{(k+1)} \leq \frac{k}{n}) = n\binom{n-1}{k}\int_0^{\frac{k}{n}} t^k(1-t)^{n-k-1}\mathrm{d}t\,,$$

and in the same manner,

$$P_{\frac{k-1}{n}}(N \geq k) = \mathbb{P}(U_{(k)} \leq \frac{k-1}{n}) = n\binom{n-1}{k-1}\int_0^{\frac{k-1}{n}} t^{k-1}(1-t)^{n-k}\mathrm{d}t\,.$$

Note that

$$\binom{n-1}{k-1} = \binom{n-1}{k}\frac{k}{n-k}\,,$$

so that (29) follows if we prove

$$k\int_0^{\frac{k-1}{n}} t^{k-1}(1-t)^{n-k}\mathrm{d}t \le (n-k)\int_0^{\frac{k}{n}} t^k(1-t)^{n-k-1}\mathrm{d}t\,. \tag{30}$$

We can establish the following chain of equivalent inequalities.

$$\begin{aligned}
& k\int_0^{\frac{k-1}{n}} t^{k-1}(1-t)^{n-k}\mathrm{d}t \le (n-k)\int_0^{\frac{k}{n}} t^k(1-t)^{n-k-1}\mathrm{d}t \\
\Leftrightarrow \quad & \int_0^{\frac{k}{n}} \frac{\mathrm{d}t^k}{\mathrm{d}t}(1-t)^{n-k}\mathrm{d}t \le -\int_0^{\frac{k}{n}} t^k\frac{\mathrm{d}(1-t)^{n-k}}{\mathrm{d}t}\mathrm{d}t + k\int_{\frac{k-1}{n}}^{\frac{k}{n}} t^{k-1}(1-t)^{n-k}\mathrm{d}t \\
\Leftrightarrow \quad & \int_0^{\frac{k}{n}} \frac{\mathrm{d}}{\mathrm{d}t}\left[t^k(1-t)^{n-k}\right]\mathrm{d}t \le k\int_{\frac{k-1}{n}}^{\frac{k}{n}} t^{k-1}(1-t)^{n-k}\mathrm{d}t \\
\Leftrightarrow \quad & \left(\frac{k}{n}\right)^k\left(1-\frac{k}{n}\right)^{n-k} \le k\int_{\frac{k-1}{n}}^{\frac{k}{n}} t^{k-1}(1-t)^{n-k}\mathrm{d}t
\end{aligned}$$

We now study the variations of the function $t \mapsto b(t) = t^{k-1}(1-t)^{n-k}$ on the interval $[(k-1)/n, k/n]$. Taking derivative, it is not hard to see that function b admits a unique local optimum, which is a maximum, at $t_0 = \frac{k-1}{n-1}$ and that $t_0 \in ((k-1)/n, k/n)$ because $k \le n$. Therefore, the function is increasing on $[(k-1)/n, t_0]$ and decreasing on $[t_0, k/n]$. It implies that

$$\int_{\frac{k-1}{n}}^{\frac{k}{n}} b(t)\mathrm{d}t \ge \frac{1}{n}\min\left[b(\frac{k-1}{n}), b(\frac{k}{n})\right].$$

Hence, the proof of (30) follows from the following two observations:

$$\left(\frac{k}{n}\right)^k\left(1-\frac{k}{n}\right)^{n-k} = \frac{k}{n}\left(\frac{k}{n}\right)^{k-1}\left(1-\frac{k}{n}\right)^{n-k} = \frac{k}{n}b(\frac{k}{n}),$$

and

$$\left(\frac{k}{n}\right)^k\left(1-\frac{k}{n}\right)^{n-k} \le \frac{k}{n}\left(\frac{k-1}{n}\right)^{k-1}\left(1-\frac{k-1}{n}\right)^{n-k} = \frac{k}{n}b(\frac{k-1}{n}).$$

While the first equality above is obvious, the second inequality can be obtained by an equivalent statement is

$$\begin{aligned}
& \left(\frac{k}{n}\right)^{k-1}\left(\frac{n-k}{n}\right)^{n-k} \le \left(\frac{k-1}{n}\right)^{k-1}\left(\frac{n-k+1}{n}\right)^{n-k} \\
\Leftrightarrow \quad & \left(\frac{k}{k-1}\right)^{k-1}\left(\frac{n-k}{n-k+1}\right)^{n-k} \le 1
\end{aligned}$$

Since the function $t \mapsto \left(\frac{t+1}{t}\right)^t$ is increasing on $[0,\infty)$, and $k \le n-k+1$, the result follows.

To conclude the proof of the Lemma, note that (28) and (29) imply that for any $q > 1/n$,

$$P_q(N \ge nq) \ge P_{\frac{1}{n}}(N \ge 2) = 1 - \left(\frac{n-1}{n}\right)^n - \left(\frac{n-1}{n}\right)^{n-1} \ge 1 - \left(\frac{1}{2}\right)^2 - \frac{1}{2} = \frac{1}{4},$$

where, in the last inequality, we used the fact that the function

$$t \mapsto 1 - \left(\frac{t-1}{t}\right)^t - \left(\frac{t-1}{t}\right)^{t-1}$$

is increasing on $[1, \infty)$. ■

References

P. L. Bartlett, M. I. Jordan, and J. D. McAuliffe. Convexity, classification, and risk bounds. *J. Amer. Statist. Assoc.*, 101(473):138–156, 2006.

A. Ben-Tal, L. El Ghaoui, and A. Nemirovski. *Robust Optimization*. Princeton Series in Applied Mathematics. Princeton University Press, Princeton, NJ, 2009.

G. Blanchard, G. Lee, and C. Scott. Semi-supervised novelty detection. *J. Mach. Learn. Res.*, 11: 2973–3009, 2010.

S. Boucheron, O. Bousquet, and G. Lugosi. Theory of classification: A survey of some recent advances. *ESAIM Probab. Stat.*, 9:323–375, 2005.

S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, Cambridge, 2004.

G. C. Calafiore and M. C. Campi. The scenario approach to robust control design. *IEEE Trans. Automat. Control*, 51(5):742–753, 2006.

A. Cannon, J. Howse, D. Hush, and C. Scovel. Learning with the Neyman-Pearson and min-max criteria. Technical Report LA-UR-02-2951, 2002.

A. Cannon, J. Howse, D. Hush, and C. Scovel. Simple classifiers. Technical Report LA-UR-03-0193, 2003.

D. Casasent and X. Chen. Radial basis function neural networks for nonlinear fisher discrimination and neyman-pearson classification. *Neural Networks*, 16(5-6):529 – 535, 2003.

L. Devroye, L. Györfi, and G. Lugosi. *A Probabilistic Theory of Pattern Recognition, Volume 31 of Applications of Mathematics (New York)*. Springer-Verlag, New York, 1996.

W. Feller. *An Introduction to Probability Theory and Its Applications. Vol. II*. Second edition. John Wiley & Sons Inc., New York, 1971.

M. Han, D. Chen, and Z. Sun. Analysis to Neyman-Pearson classification with convex loss function. *Anal. Theory Appl.*, 24(1):18–28, 2008.

V. Koltchinskii. *Oracle Inequalities in Empirical Risk Minimization and Sparse Recovery Problems*. École d'Été de Probabilités de Saint-Flour XXXVIII-2008. Lecture Notes in Mathematics 2033. Berlin: Springer. ix, 254 p. EUR 48.10 , 2011.

C. M. Lagoa, X. Li, and M. Sznaier. Probabilistically constrained linear programs and risk-adjusted controller design. *SIAM J. Optim.*, 15(3):938–951 (electronic), 2005.

E. L. Lehmann and J. P. Romano. *Testing Statistical Hypotheses*. Springer Texts in Statistics. Springer, New York, third edition, 2005.

A. Nemirovski and A. Shapiro. Convex approximations of chance constrained programs. *SIAM J. Optim.*, 17(4):969–996, 2006.

A. Prékopa. *Stochastic Programming, volume 324 of Mathematics and its Applications*. Kluwer Academic Publishers Group, Dordrecht, 1995.

R. T. Rockafellar. *Convex Analysis*. Princeton Landmarks in Mathematics. Princeton University Press, Princeton, NJ, 1997. Reprint of the 1970 original, Princeton Paperbacks.

R. Schapire. The strength of weak learnability. *Machine learning*, 5(2):197–227, 1990.

C. Scott. Comparison and design of Neyman-Pearson classifiers. Unpublished, 2005.

C. Scott. Performance measures for Neyman-Pearson classification. *IEEE Trans. Inform. Theory*, 53(8):2852–2863, 2007.

C. Scott and R. Nowak. A Neyman-Pearson approach to statistical learning. *IEEE Transactions on Information Theory*, 51(11):3806–3819, 2005.

E. V. Slud. Distribution inequalities for the binomial law. *Ann. Probability*, 5(3):404–412, 1977.

Journal of Machine Learning Research 12 (2011) 2857-2878 Submitted 4/11; Revised 9/11; Published 10/11

Efficient Learning with Partially Observed Attributes*

Nicolò Cesa-Bianchi NICOLO.CESA-BIANCHI@UNIMI.IT
DSI, Università degli Studi di Milano
via Comelico, 39
20135 Milano, Italy

Shai Shalev-Shwartz SHAIS@CS.HUJI.AC.IL
The Hebrew University
Givat Ram, Jerusalem 91904, Israel

Ohad Shamir OHADSH@MICROSOFT.COM
Microsoft Research
One Memorial Drive
Cambridge, MA 02142, USA

Editor: Russ Greiner

Abstract

We investigate three variants of budgeted learning, a setting in which the learner is allowed to access a limited number of attributes from training or test examples. In the "local budget" setting, where a constraint is imposed on the number of available attributes per training example, we design and analyze an efficient algorithm for learning linear predictors that actively samples the attributes of each training instance. Our analysis bounds the number of additional examples sufficient to compensate for the lack of full information on the training set. This result is complemented by a general lower bound for the easier "global budget" setting, where it is only the overall number of accessible training attributes that is being constrained. In the third, "prediction on a budget" setting, when the constraint is on the number of available attributes per test example, we show that there are cases in which there exists a linear predictor with zero error but it is statistically impossible to achieve arbitrary accuracy without full information on test examples. Finally, we run simple experiments on a digit recognition problem that reveal that our algorithm has a good performance against both partial information and full information baselines.

Keywords: budgeted learning, statistical learning, linear predictors, learning with partial information, learning theory

1. Introduction

Consider the problem of predicting whether a person has some disease based on medical tests. In principle, we may draw a sample of the population, perform a large number of medical tests on each person in the sample, and use this information to train a classifier. In many situations, however, this approach is unrealistic. First, patients participating in the experiment are generally not willing to go through a large number of medical tests. Second, each test has some associated cost, and we typically have a budget on the amount of money to spend for collecting the training information. This scenario, where there is a hard constraint on the number of training attributes the

*. A short version of this paper has been presented in ICML 2010.

learner has access to, is known as *budgeted learning*.[1] Note that the constraint on the number of training attributes may be local (no single participant is willing to undergo many tests) or global (the overall number of tests that can be performed is limited). In a different but related budgeted learning setting, the system may be facing a restriction on the number of attributes that can be viewed at test time. This may happen, for example, in a search engine, where a ranking of web pages must be generated for each incoming user query and there is no time to evaluate a large number of attributes to answer the query.

We may thus distinguish three basic budgeted learning settings:

- **Local Budget Constraint:** The learner has access to at most k attributes of each individual example, where k is a parameter of the problem. The learner has the freedom to actively choose *which* of the attributes is revealed, as long as at most k of them will be given.

- **Global Budget Constraint:** The total number of training attributes the learner is allowed to see is bounded by k. As in the local budget constraint setting, the learner has the freedom to actively choose which of the attributes is revealed. In contrast to the local budget constraint setting, the learner can choose to access more than k/m attributes from specific examples (where m is the overall number of examples) as long as the global number of attributes is bounded by k.

- **Prediction on a budget:** The learner receives the entire training set, however, at test time, the predictor can see at most k attributes of each instance and then must form a prediction. The predictor is allowed to actively choose which of the attributes is revealed.

In this paper we focus on budgeted linear regression, and prove negative and positive learning results in the three abovementioned settings. Our first result shows that, under a *global* budget constraint, no algorithm can learn a general d-dimensional linear predictor while observing less than $\Omega(d)$ attributes at training time. This is complemented by the following positive result: we show an efficient algorithm for learning under a given *local* budget constraint of $2k$ attributes per example, for any $k \geq 1$. The algorithm actively picks which attributes to observe in each example in a randomized way depending on past observed attributes, and constructs a "noisy" version of *all* attributes. Intuitively, we can still learn despite the error of this estimate because instead of receiving the exact value of each individual example in a small set it suffices to get noisy estimations of many examples. We show that the overall number of attributes our algorithm needs to learn a regressor is at most a factor of d bigger than that used by standard regression algorithms that view all the attributes of each example. Ignoring logarithmic factors, the same gap of d exists when the attribute bound of our algorithm is specialized to the choice of parameters that is used to prove the abovementioned $\Omega(d)$ lower bound under the global budget constraint.

In the prediction on a budget setting, we prove that in general it is not possible (even with an infinite amount of training examples) to build an active classifier that uses at most two attributes of each example at test time, and whose error will be smaller than a constant. This in contrast with the local budget setting, where it is possible to learn a consistent predictor by accessing at most two attributes of each example at training time.

1. See, for example, `webdocs.cs.ualberta.ca/~greiner/BudgetedLearning/`.

2. Related Work

The notion of budgeted learning is typically identified with the "global budget" and "prediction on a budget" settings—see, for example, Deng et al. (2007), Kapoor and Greiner (2005a,b) and Greiner et al. (2002) and references therein. The more restrictive "local budget" setting has been first proposed in Ben-David and Dichterman (1998) under the name of "learning with restricted focus of attention". Ben-David and Dichterman (1998) considered binary classification and showed learnability of several hypothesis classes in this model, like k-DNF and axis-aligned rectangles. However, to the best of our knowledge, no efficient algorithm for the class of linear predictors has been so far proposed.[2]

Our algorithm for the local budget setting actively chooses which attributes to observe for each example. Similarly to the heuristics of Deng et al. (2007), we borrow ideas from the adversarial multi-armed bandit problem (Auer et al., 2003; Cesa-Bianchi and Lugosi, 2006). However, our algorithm is guaranteed to be attribute efficient, comes with finite sample generalization bounds, and is provably competitive with algorithms which enjoy full access to the data. A related but different setting is multi-armed bandit on a global budget—see, for example, Guha and Munagala (2007) and Madani et al. (2004). There one learns the single best arm rather than the best linear combination of many attributes, as we do here. Similar protocols were also studied in the context of active learning (Cohn et al., 1994; Balcan et al., 2006; Hanneke, 2007, 2009; Beygelzimer et al., 2009), where the learner can ask for the target associated with specific examples.

Finally, our technique is reminiscent of methods used in the compressed learning framework (Calderbank et al., 2009; Zhou et al., 2009), where data is accessed via a small set of random linear measurements. Unlike compressed learning, where learners are both trained and evaluated in the compressed domain, our techniques are mainly designed for a scenario in which only the access to training data is restricted.

We note that a recent follow-up work (Hazan and Koren, 2011) present 1-norm and 2-norm based algorithms for our local budget setting, whose theoretical guarantees improve on those presented in this paper, and match our lower bound to within logarithmic factors.

3. Linear Regression

We consider linear regression problems where each example is an instance-target pair, $(x,y) \in \mathbb{R}^d \times \mathbb{R}$. We refer to x as a vector of attributes. Throughout the paper we assume that $\|x\|_\infty \leq 1$ and $|y| \leq B$. The goal of the learner is to find a linear predictor $x \mapsto \langle w,x \rangle$. In the rest of the paper, we use the term predictor to denote the vector $w \in \mathbb{R}^d$. The performance of a predictor w on an instance-target pair, $(x,y) \in \mathbb{R}^d \times \mathbb{R}$, is measured by a loss function $\ell(\langle w,x \rangle, y)$. For simplicity, we focus on the squared loss function, $\ell(a,b) = (a-b)^2$, and briefly mention other loss functions in Section 8. Following the standard framework of statistical learning (Haussler, 1992; Devroye et al., 1996; Vapnik, 1998), we model the environment as a joint distribution $\mathcal{D}$ over the set of instance-target pairs, $\mathbb{R}^d \times \mathbb{R}$. The goal of the learner is to find a predictor with low risk, defined as the expected loss

$$L_{\mathcal{D}}(w) \stackrel{\text{def}}{=} \mathop{\mathbb{E}}_{(x,y)\sim\mathcal{D}} \left[\ell(\langle w,x \rangle, y) \right] .$$

2. Ben-David and Dichterman (1998) do describe learnability results for similar classes but only under the restricted family of product distributions.

Since the distribution $\mathcal{D}$ is unknown, the learner relies on a training set of m examples $S = \big\{(x_1, y_1), \ldots, (x_m, y_m)\big\}$, which are assumed to be sampled i.i.d. from $\mathcal{D}$. We denote the training loss by

$$L_S(w) \stackrel{\text{def}}{=} \frac{1}{m} \sum_{i=1}^{m} (\langle w, x_i \rangle - y_i)^2 \ .$$

4. Impossibility Results

Our first result states that any budget learning algorithm (local or global) needs in general a budget of $\Omega(d)$ attributes for learning a d-dimensional linear predictor.

Theorem 1 *For any $d \geq 4$ and $\varepsilon \in \big(0, \frac{1}{16}\big)$, there exists a distribution $\mathcal{D}$ over $\{-1,+1\}^d \times \{-1,+1\}$ and a weight vector $w^\star \in \mathbb{R}^d$, with $\|w^\star\|_0 = 1$ and $\|w^\star\|_2 = \|w^\star\|_1 = 2\sqrt{\varepsilon}$, such that any learning algorithm must see at least*

$$k \geq \frac{1}{2} \left\lfloor \frac{d}{96\varepsilon} \right\rfloor$$

attributes in order to learn a linear predictor w such that $L_{\mathcal{D}}(w) - L_{\mathcal{D}}(w^\star) < \varepsilon$.

The proof is given in the Appendix. In Section 6 we prove that under the same assumptions as those of Theorem 1, it is possible to learn a predictor using a local budget of two attributes per example and using a total of $\widetilde{O}(d^2)$ training examples. Thus, ignoring logarithmic factors hidden in the $\widetilde{O}$ notation, we have a multiplicative gap of d between the lower bound and the upper bound.

Next, we consider the prediction on a budget setting. Greiner et al. (2002) studied this setting and showed positive results regarding (agnostic) PAC-learning of *k-active* predictors. A k-active predictor is restricted to use at most k attributes per test example x, where the choice of the i-th attribute of x may depend on the values of the $i-1$ attributes of x that have been already observed. Greiner et al. (2002) show that it is possible to learn a k-active predictor from training examples whose performance is slightly worse than that of the best k-active predictor. But, how good are the predictions of the best k-active predictor? We now show that even in simple cases in which there exists a linear predictor $w^\star$ with $L_{\mathcal{D}}(w^\star) = 0$, the risk of the best k-active predictor can be high. The following theorem indeed shows that if the only constraint on $w^\star$ is bounded ℓ_2 norm, then the risk can be as high as $1 - \frac{k}{d}$. We use the notation $L_{\mathcal{D}}(A)$ to denote the expected loss of the k-active predictor A on a test example.

Theorem 2 *There exists a weight vector $w^\star \in \mathbb{R}^d$ and a distribution $\mathcal{D}$ such that $\|w^\star\|_2 = 1$ and $L_{\mathcal{D}}(w^\star) = 0$, while any k-active predictor A must have $L_{\mathcal{D}}(A) \geq 1 - \frac{k}{d}$.*

Note that the risk of the constant prediction of zero is 1. Therefore, the theorem tells us that no active predictor can get an improvement over the naive predictor of more than $\frac{k}{d}$.

Proof For any $d > k$ let $w^\star = \big(1/\sqrt{d}, \ldots, 1/\sqrt{d}\big)$. Let $x \in \{\pm 1\}^d$ be distributed uniformly at random and y is determined deterministically to be $\langle w^\star, x \rangle$. Then, $L_{\mathcal{D}}(w^\star) = 0$ and $\|w^\star\|_2 = 1$. Without loss of generality, suppose the k-active predictor asks for the first k attributes of a test example and forms its prediction to be $\hat{y}$. Since the generation of attributes is independent, we have that the value of $x_{k+1}, \ldots, x_d$ does not depend neither on $x_1, \ldots, x_k$ nor on $\hat{y}$. Using this and the fact that $\mathbb{E}[x_j] = 0$ for

all j we therefore obtain

$$\begin{aligned}
\mathbb{E}\left[(\hat{y}-\langle w^\star,x\rangle)^2\right] &= \mathbb{E}\left[\left(\hat{y}-\sum_{i=1}^{k} w_i^\star x_i - \sum_{j=k+1}^{d} w_j^\star x_j\right)^2\right] \\
&= \mathbb{E}\left[\left(\hat{y}-\sum_{i=1}^{k} w_i^\star x_i\right)^2\right] + \sum_{j=k+1}^{d} (w_j^\star)^2\,\mathbb{E}[x_j^2] \\
&\quad + 2\hat{y}\sum_{j=k+1}^{d} w_j^\star\,\mathbb{E}[x_j] - 2\sum_{i=1}^{k}\sum_{j=k+1}^{d} w_i^\star w_j^\star\,\mathbb{E}[x_i]\,\mathbb{E}[x_j] \\
&= \mathbb{E}\left[\left(\hat{y}-\sum_{i=1}^{k} w_i^\star x_i\right)^2\right] + \sum_{i>k}(w_i^\star)^2\,\mathbb{E}[x_i^2] + 0 \\
&\geq 0 + \frac{d-k}{d} = 1-\frac{k}{d}
\end{aligned}$$

which concludes our proof. ■

It is well known that a low 1-norm of $w^\star$ encourages sparsity of the learned predictor, which naturally helps in designing active predictors. The following theorem shows that even if we restrict $w^\star$ to have $\|w^\star\|_1 = 1$, $L_{\mathcal{D}}(w^\star) = 0$, and $\|w^\star\|_0 > k$, we still have that the risk of the best k-active predictor can be non-vanishing.

Theorem 3 *There exists a weight vector $w^\star \in \mathbb{R}^d$ and a distribution $\mathcal{D}$ such that $\|w^\star\|_1 = 1$, $L_{\mathcal{D}}(w^\star) = 0$, and $\|w^\star\|_0 = ck$ (for $c > 1$) such that any k-active predictor A must have $L_{\mathcal{D}}(A) \geq \left(1-\frac{1}{c}\right)\frac{1}{ck}$.*

For example, if in the theorem above we choose $c = 2$, then $\|w^\star\|_0 = 2k$ and $L_{\mathcal{D}}(A) \geq \frac{1}{4k}$. If we choose instead $c = \frac{k+1}{k}$, then $\|w^\star\|_0 = k+1$ and $L_{\mathcal{D}}(A) \geq \frac{1}{(k+1)^2}$. Note that if $\|w^\star\|_0 \leq k$ there is a trivial way to predict on a budget of k attributes by always querying the attributes corresponding to the non-zero elements of $w^\star$.

Proof Let

$$w^\star = \big(\underbrace{\tfrac{1}{ck},\ldots,\tfrac{1}{ck}}_{ck \text{ components}},0,\ldots,0\big)$$

and, similarly to the proof of Theorem2, let $x \in \{\pm 1\}^d$ be distributed uniformly at random and let y be determined deterministically to be $\langle w^\star,x\rangle$. Then, $L_{\mathcal{D}}(w^\star) = 0$, $\|w^\star\|_1 = 1$, and $\|w^\star\|_0 = ck$. Without loss of generality, suppose the k-active predictor asks for the first $k < ck$ attributes of a test example and form its prediction to be $\hat{y}$. Again similarly to the proof of Theorem2, since the generation of attributes is independent, we have that the value of $x_{k+1},\ldots,x_d$ does not depend on

$x_1,\ldots,x_k$, and on $\hat{y}$. Therefore,

$$\begin{aligned}\mathbb{E}\left[(\hat{y}-\langle w^\star,x\rangle)^2\right] &= \mathbb{E}\left[\left(\hat{y}-\sum_{i=1}^{k} w_i^\star x_i\right)^2\right]+\sum_{i>k}(w_i^\star)^2\,\mathbb{E}[x_i^2]\\ &\geq 0+\frac{ck-k}{(ck)^2}\\ &= \frac{c-1}{c^2k}=\left(1-\frac{1}{c}\right)\frac{1}{ck}\end{aligned}$$

which concludes our proof. ■

These negative results highlight an interesting phenomenon: in Section 6 we show that one can learn an arbitrarily accurate predictor w with a local budget of $k=2$. However, here we show that even if we know the optimal $w^\star$, we might not be able to accurately predict a new partially observed example unless k is very large. Therefore, at least in the worst-case sense, learning on a budget is much easier than predicting on a budget.

5. Local Budget Constraint: A Baseline Algorithm

In this section we describe a straightforward adaptation of Lasso (Tibshirani, 1996) to the local budget setting. This adaptation is based on a direct nonadaptive estimate of the loss function. In Section 6 we describe a more effective approach, which combines a stochastic gradient descent algorithm called Pegasos (Shalev-Shwartz et al., 2007) with the adaptive sampling of attributes to estimate the gradient of the loss at each step.

A popular approach for learning a linear regressor is to minimize the empirical loss on the training set plus a regularization term, which often takes the form of a norm of the predictor w. For example, in ridge regression the regularization term is $\|w\|_2^2$ and in Lasso the regularization term is $\|w\|_1$. Instead of regularization, we can include a constraint of the form $\|w\|_1 \leq B$ or $\|w\|_2 \leq B$. Modulo an appropriate choice of the parameters, the regularization form is equivalent to the constraint form. In the constraint form, the predictor is a solution to the following optimization problem

$$\begin{aligned}\min_{w\in\mathbb{R}^d} \quad & \frac{1}{|S|}\sum_{(x,y)\in S}\left(\langle w,x\rangle-y\right)^2\\ \text{s.t.} \quad & \|w\|_p \leq B\end{aligned} \tag{1}$$

where $S=\{(x_1,y_1),\ldots,(x_m,y_m)\}$ is a training set of m examples, B is the regularization parameter, and p is 1 for Lasso and 2 for ridge regression.

We start with a standard risk bound for constrained predictors.

Lemma 4 *Let $\mathcal{D}$ be a distribution on pairs $(x,y)\in\mathbb{R}^d\times\mathbb{R}$ such that $\|x\|_\infty\leq 1$ and $|y|\leq B$ holds with probability one. Then there exists a constant $c>0$ such that*

$$\max_{w:\,\|w\|_1\leq B}\left|L_S(w)-L_{\mathcal{D}}(w)\right| = cB^2\sqrt{\frac{1}{m}\ln\frac{d}{\delta}}\,.$$

holds with probability at least $1-\delta$ with respect to the random draw of the training set S of size m from $\mathcal{D}$.

Proof We apply the following Rademacher bound (Kakade et al., 2008)

$$\left|L_S(w) - L_{\mathcal{D}}(w)\right| \leq L_{\max} B \sqrt{\frac{2}{m} \ln 2d} + \ell_{\max} \sqrt{\frac{1}{2m} \ln \frac{2}{\delta}}$$

that holds with probability at least $1-\delta$ for all $w \in \mathbb{R}^d$ such that $\|w\|_1 \leq B$, where $L_{\max}$ bounds the Lipschitz constant for the square loss from above, and $\ell_{\max}$ bounds the square loss from above. The result then follows by observing that $\left|(a-y)^2 - (b-y)^2\right| \leq |a-b|\,|a+b-2y|$. Hence, $L_{\max} \leq \max_{a,b,y} |a+b-2y| = 4B$ where both a and b are of the form $\langle w,x\rangle$, and we used the fact $\left|\langle w,x\rangle\right| \leq B$ (recall that $\|x\|_\infty \leq 1$) together with the assumption $|y| \leq B$. Similarly, under the same assumptions, $\ell_{\max} = \max_{a,y}(a-y)^2 = 4B^2$. ■

This immediately leads to the following risk bound for Lasso.

Corollary 5 *If $\widehat{w}$ is a minimizer of (1) with $p=1$, then there exists a constant $c>0$ such that, under the same assumptions as Lemma 4,*

$$L_{\mathcal{D}}(\widehat{w}) \leq \min_{w:\,\|w\|_1 \leq B} L_D(w) + c\,B^2 \sqrt{\frac{1}{m} \ln \frac{d}{\delta}} \tag{2}$$

holds with probability at least $1-\delta$ over the random draw of the training set S of size m from $\mathcal{D}$.

To adapt Lasso to the partial information case, we first rewrite the squared loss as follows:

$$\left(\langle w,x\rangle - y\right)^2 = w^\top x x^\top w - 2y\,x^\top w + y^2$$

where w, x are column vectors and $w^\top, x^\top$ are their corresponding transpose (i.e., row vectors). Next, we estimate the matrix $xx^\top$ and the vector x using the partial information we have, and then we solve the optimization problem given in (1) with the estimated values of $xx^\top$ and x. To estimate the vector x we can pick an index i uniformly at random from $[d] = \{1,\ldots,d\}$ and define the estimation to be a vector v such that

$$v_r = \begin{cases} d\,x_r & \text{if } r = i \\ 0 & \text{else} \end{cases}. \tag{3}$$

It is easy to verify that v is an unbiased estimate of x, namely, $\mathbb{E}[v] = x$ where expectation is with respect to the choice of the index i. To estimate the matrix $xx^\top$ we could pick two indices i, j independently and uniformly at random from $[d]$, and define the estimation to be a matrix with all zeros except $d^2 x_i x_j$ in the (i,j) entry. However, this yields a non-symmetric matrix which will make our optimization problem with the estimated matrix non-convex. To overcome this obstacle, we symmetrize the matrix by adding its transpose and dividing by 2. This sampling process can be easily generalized to the case where $k > 1$ attributes can be seen. The resulting baseline procedure[3] is given in Algorithm 1.

The following theorem shows that similar to Lasso, the Baseline algorithm is competitive with the optimal linear predictor with a bounded 1-norm.

3. We note that an even simpler approach is to arbitrarily assume that the correlation matrix is the identity matrix and then the solution to the loss minimization problem is simply the averaged vector, $w = \sum_{(x,y)\in S} y x$. In that case, we can simply replace x by its estimated vector as defined in (3). While this naive approach can work on very simple classification tasks, it will perform poorly on realistic data sets, in which the correlation matrix is not likely to be identity. Indeed, in our experiments with the MNIST data set, we found out that this approach performed poorly relatively to the algorithms proposed in this paper.

ALGORITHM: Baseline(S,k)
INPUT: Training set S of size m, local budget $k \geq 2$ (with k even)
INITIALIZE: $A = 0 \in \mathbb{R}^{d\times d}$; $v = 0 \in \mathbb{R}^d$; $y = 0$

for each $(x,y) \in S$
  $v = 0 \in \mathbb{R}^d$; $A = 0 \in \mathbb{R}^{d\times d}$
  Choose a set C of k entries from $[d]$, uniformly without replacement
  for each $c \in C$
    $v_c = v_c + \frac{d}{k} x_c$
  Randomly split C into two sets I, J of size $k/2$ each
  for each $(i,j) \in I \times J$
    $A_{i,j} = A_{i,j} + 2\left(\frac{d}{k}\right)^2 x_i x_j \quad ; \quad A_{j,i} = A_{j,i} + 2\left(\frac{d}{k}\right)^2 x_i x_j$
  end
  $A = A + \frac{A}{m} \quad ; \quad v = v + 2y\frac{v}{m} \quad ; \quad y = y + \frac{y^2}{m}$
end
Let $\tilde{L}_S(w) = w^\top A w + w^\top v + y$
OUTPUT: $\widehat{w} = \operatorname*{argmin}_{w:\|w\|_1 \leq B} \tilde{L}_S(w)$

Figure 1: An adaptation of Lasso to the local budget setting, where the learner can view at most k attributes of each training example. The predictive performance of this algorithm is analyzed in Theorem 6.

Theorem 6 *Let $\mathcal{D}$ be a distribution on pairs $(x,y) \in \mathbb{R}^d \times \mathbb{R}$ such that $\|x\|_\infty \leq 1$ and $|y| \leq B$ with probability one. Let $\hat{w}$ be the output of Baseline(S,k), where $|S| = m$. Then there exists a constant $c > 0$ such that*

$$L_{\mathcal{D}}(\hat{w}) \leq \min_{w:\|w\|_1 \leq B} L_D(w) + c\left(\frac{dB}{k}\right)^2 \sqrt{\frac{1}{m}\ln\frac{d}{\delta}}$$

holds with probability of at least $1-\delta$ over the random draw of the training set S from $\mathcal{D}$ and the algorithm's own randomization.

The above theorem tells us that for a sufficiently large training set we can find a very good predictor. Put another way, a large number of examples can compensate for the lack of full information on each individual example. In particular, to overcome the extra factor $(d/k)^2$ in the bound, which does not appear in the full information bound given in (2), we need to increase m by a factor of $(d/k)^4$. In the next subsection, we describe a better, adaptive procedure for the partial information case.

In view of proving Theorem 6, we first show that sampling k elements without replacements and then averaging the result has the same expectation as sampling just once.

Lemma 7 *Let C be a set of n elements and let $f : C \to \mathbb{R}$ be an arbitrary function. Let $\mathcal{C}_k = \{C' \subset C : |C'| = k\}$ and let U be the uniform distribution over $\mathcal{C}_k$. Then*

$$\mathop{\mathbb{E}}_{C' \sim U}\left[\frac{1}{k}\sum_{c' \in C'} f(c')\right] = \frac{1}{n}\sum_{c \in C} f(c)\ .$$

Proof We have

$$\begin{aligned}
\mathop{\mathbb{E}}_{C' \sim U(}\left[\frac{1}{k}\sum_{c' \in C'} f(c')\right] &= \frac{1}{\binom{n}{k}}\sum_{C' \in \mathcal{C}_k}\frac{1}{k}\sum_{c' \in C'} f(c') \\
&= \frac{1}{k\binom{n}{k}}\sum_{c \in C} f(c)\left|\{C' \in \mathcal{C}_k : c' \in C'\}\right| \\
&= \frac{\binom{n-1}{k-1}}{k\binom{n}{k}}\sum_{c \in C} f(c) \\
&= \frac{1}{n}\sum_{c \in C} f(c)
\end{aligned}$$

and this concludes the proof. ■

We now show that the estimation matrix constructed by the Baseline algorithm is likely to be close to the true correlation matrix over the training set.

Lemma 8 *Let A_t be the matrix constructed at iteration t of the Baseline algorithm and note that $A = \frac{1}{m}\sum_{t=1}^{m} A_t$. Let $X = \frac{1}{m}\sum_{t=1}^{m} x_t x_t^\top$. Then, with probability of at least $1-\delta$ over the algorithm's own randomness we have that*

$$\left|A_{r,s} - X_{r,s}\right| \leq \left(\frac{d}{k}\right)^2 \sqrt{\frac{8}{m}\ln\left(\frac{2d^2}{\delta}\right)} \qquad r,s = 1,\ldots,d.$$

Proof Based on Lemma 7, it is easy to verify that $\mathbb{E}[A_t] = x_t^\top x_t$. Additionally, since we sample without replacements, each element of A_t is in $\left[-2\left(\frac{d}{k}\right)^2, 2\left(\frac{d}{k}\right)^2\right]$ because we assume $\|x_t\|_\infty \leq 1$. Therefore, we can apply Hoeffding's inequality on each element of A and obtain that

$$\mathbb{P}\left[\left|A_{r,s} - X_{r,s}\right| > \varepsilon\right] \leq 2\exp\left(-\frac{m\varepsilon^2}{8}\left(\frac{k}{d}\right)^4\right)\ .$$

Combining the above with the union bound we obtain that

$$\mathbb{P}\left[\exists(r,s)\ :\ \left|A_{r,s} - X_{r,s}\right| > \varepsilon\right] \leq 2d^2\exp\left(-\frac{m\varepsilon^2}{8}\left(\frac{k}{d}\right)^4\right)\ .$$

Setting the right-hand side of the above to δ and rearranging terms concludes the proof. ■

Next, we show that the estimate of the linear part of the objective function is also likely to be accurate.

Lemma 9 *Let v_t be the vector constructed at iteration t of the Baseline algorithm and note that $v = \frac{1}{m}\sum_{t=1}^{m} 2\,y_t\,v_t$. Let $x = \frac{1}{m}\sum_{t=1}^{m} 2\,y_t\,x_t$. Then, with probability at least $1-\delta$ over the algorithm's own randomness we have that*

$$\|v-x\|_\infty \le \frac{dB}{k}\sqrt{\frac{8}{m}\ln\left(\frac{2d}{\delta}\right)}\,.$$

Proof Based on Lemma 7, it is easy to verify that $\mathbb{E}[2\,y_t\,v_t] = 2\,y_t\,x_t$. Additionally, since we sample k elements without replacement, each element of v_t is in $\left[-\frac{d}{k},\frac{d}{k}\right]$ (because we assume $\|x_t\|_\infty \le 1$) and thus each element of $2y_tv_t$ is in $\left[-\frac{2dB}{k},\frac{2dB}{k}\right]$ (because we assume that $|y_t| \le B$). Therefore, we can apply Hoeffding's inequality on each element of v and obtain that

$$\mathbb{P}\Big[\big|v_r - x_r\big| > \varepsilon\Big] \le 2\exp\left(-\frac{m\varepsilon^2}{8}\left(\frac{k}{dB}\right)^2\right)\,.$$

Combining the above with the union bound we obtain that

$$\mathbb{P}\Big[\exists(r,s)\,:\,\big|A_{r,s}-X_{r,s}\big| > \varepsilon\Big] \le 2d\exp\left(-\frac{m\varepsilon^2}{8}\left(\frac{k}{dB}\right)^2\right)\,.$$

Setting the right-hand side of the above to δ and rearranging terms concludes proof. ■

Next, we show that the estimated training loss

$$\tilde{L}_S(w) = w^\top A w + w^\top v + y$$

computed by the Baseline algorithm is close to the true training loss.

Lemma 10 *With probability greater than $1-\delta$ over the Baseline algorithm's own randomization, for all w such that $\|w\|_1 \le B$,*

$$\big|\tilde{L}_S(w) - L_S(w)\big| \le \left(\frac{Bd}{k}\right)^2\sqrt{\frac{32}{m}\ln\left(\frac{2d^2}{\delta}\right)}\,.$$

Proof Using twice Hölder's inequality and Lemma 8 we get

$$\begin{aligned}\big|w^\top(A-X)w\big| &\le \|w\|_1\,\big\|(A-X)w\big\|_\infty \le \|w\|_1^2\max_{r,s=1,\ldots,d}\big|(A-X)_{r,s}\big| \\ &\le \left(\frac{Bd}{k}\right)^2\sqrt{\frac{8}{m}\ln\left(\frac{2d^2}{\delta}\right)}\,. \qquad (4)\end{aligned}$$

Similarly, using Hölder's inequality and Lemma 9 we also get

$$\big|w^\top(v-x)\big| \le \frac{B^2 d}{k}\sqrt{\frac{8}{m}\ln\left(\frac{2d}{\delta}\right)}\,. \qquad (5)$$

Using the triangle inequality, (4)–(5), and the union bound we finally obtain

$$\begin{aligned}\left|\tilde{L}_S(w) - L_S(w)\right| &= \left|w^\top A w + w^\top v + y - w^\top X w - w^\top x - y\right| \\ &\leq \left|w^\top (A - X) w\right| + \left|w^\top (v - x)\right| \\ &\leq \left(\frac{Bd}{k}\right)^2 \sqrt{\frac{8}{m} \ln\left(\frac{2d^2}{\delta}\right)} + \frac{B^2 d}{k} \sqrt{\frac{8}{m} \ln\left(\frac{2d}{\delta}\right)}\end{aligned}$$

which upon slight simplifications concludes the proof. ■

We are now ready to prove Theorem 6.

Proof (of Theorem 6) Lemma 4 states that with probability greater than $1-\delta$ over the random draw of a training set S of m examples, for all w such that $\|w\|_1 \leq B$, we have that

$$\left|L_S(w) - L_{\mathcal{D}}(w)\right| = c' B^2 \sqrt{\frac{1}{m} \ln \frac{d}{\delta}}$$

for some $c' > 0$. Combining the above with Lemma 10, we obtain that for some $c > 0$, with probability at least $1-\delta$ over both the random draw of the training set and the algorithm's own randomization,

$$\left|L_{\mathcal{D}}(w) - \tilde{L}_S(w)\right| \leq \left|L_{\mathcal{D}}(w) - L_S(w)\right| + \left|L_S(w) - \tilde{L}_S(w)\right| \leq c \left(\frac{dB}{k}\right)^2 \sqrt{\frac{1}{m} \ln \frac{d}{\delta}}$$

for all w such that $\|w\|_1 \leq B$. The proof of Theorem 6 follows since the Baseline algorithm minimizes $\tilde{L}_S(w)$. ■

6. Gradient-Based Attribute Efficient Regression

In this section, by avoiding the estimation of the matrix $xx^\top$, we significantly decrease the number of additional examples sufficient for learning with k attributes per training example. To do so, we do not try to estimate the loss function but rather to estimate the *gradient* $\nabla \ell(w) = 2\big(\langle w, x\rangle - y\big)x$, with respect to w, of the squared loss function $\ell(w) = \big(\langle w, x\rangle - y\big)^2$. Each vector w defines a probability distribution P over $[d]$ by letting $P(i) = |w_i| / \|w\|_1$. We can estimate the gradient using an even number $k \geq 2$ of attributes as follows. First, we randomly pick a subset $i_1, \ldots, i_{k/2}$ from $[d]$ according to the uniform distribution over the $k/2$-subsets in $[d]$. Based on this, we estimate the vector x via

$$v = \frac{2}{k} d \sum_{s=1}^{k/2} x_{i_s} e_{i_s} \tag{6}$$

where e_j is the j-th element of the canonical basis of $\mathbb{R}^d$. Second, we randomly pick $j_1, \ldots, j_{k/2}$ from $[d]$ without replacement according to the distribution defined by w. Based on this, we estimate the term $\langle w, x\rangle$ by

$$\hat{y} = \frac{2}{k} \|w\|_1 \sum_{s=1}^{k/2} \operatorname{sgn}\big(w_{j_s}\big) x_{j_s} e_{j_s} \,. \tag{7}$$

This allows us to obtain an unbiased estimate of the gradient, as stated by the following simple result.

Lemma 11 *Fix any $w,x \in \mathbb{R}^d$ and $y \in \mathbb{R}$ and let $\ell(w) = \big(\langle w,x\rangle - y\big)$ be the square loss. Then the estimate*

$$\widetilde{\nabla}\ell(w) = 2\big(\hat{y} - y\big)v \tag{8}$$

satisfies $\mathbb{E}\,\widetilde{\nabla}\ell(w) = 2\big(\langle w,x\rangle - y\big)x = \nabla\ell(w)$.

Proof Since $\mathbb{E}[d\,x_j\,e_j] = x$ for a random $j \in [d]$, Lemma 7 immediately implies that $\mathbb{E}[v] = x$. Moreover, it is easy to see that $\mathbb{E}\big[\|w\|_1 \operatorname{sgn}(w_i)\,x_i\,e_i\big] = \langle w,x\rangle$ when i is drawn with probability $P(i) = |w_i|\big/\|w\|_1$. Hence $\mathbb{E}[\hat{y}] = \langle w,x\rangle$. The proof is concluded by noting that $i_1,\ldots,i_{k/2}$ are drawn independently from $j_1,\ldots,j_{k/2}$. ■

The advantage of the above approach over the loss based approach we took before is that the magnitude of each element of the gradient estimate is order of $d\,\|w\|_1$. This is in contrast to what we had for the loss based approach, where the magnitude of each element of the matrix A was order of d^2. In many situations, the 1-norm of a good predictor is significantly smaller than d and in these cases the gradient based estimate is better than the loss based estimate. However, while in the previous approach our estimation did not depend on a specific w, now the estimation depends on w. We therefore need an iterative learning method in which at each iteration we use the gradient of the loss function on an individual example. Luckily, the stochastic gradient descent approach conveniently fits our needs.

Concretely, below we describe a variant of the Pegasos algorithm (Shalev-Shwartz et al., 2007) for learning linear regressors. Pegasos tries to minimize the regularized risk

$$\min_w \frac{\lambda}{2}\|w\|_2^2 \quad + \mathop{\mathbb{E}}_{(x,y)\sim\mathcal{D}}\Big[\big(\langle w,x\rangle - y\big)^2\Big]\,. \tag{9}$$

Of course, the distribution $\mathcal{D}$ is unknown, and therefore we cannot hope to solve the above problem exactly. Instead, Pegasos finds a sequence of weight vectors that (on average) converge to the solution of (9). We start with the all zeros vector $w = 0 \in \mathbb{R}^d$. Then, at each iteration Pegasos picks the next example in the training set (which is equivalent to sampling a fresh example according to $\mathcal{D}$) and calculates the gradient of the regularized loss

$$g(w) = \frac{\lambda}{2}\|w\|_2^2 + \big(\langle w,x\rangle - y\big)^2$$

for this example with respect to the current weight vector w. This gradient is simply $\nabla g(w) = \lambda w + \nabla\ell(w)$, where $\nabla\ell(w) = 2\big(\langle w,x\rangle - y\big)x$. Finally, Pegasos updates the predictor according to the gradient descent rule $w \leftarrow w - \frac{1}{\lambda t}\nabla g(w)$ where t is the current iteration number. This can be rewritten as $w \leftarrow \big(1 - \frac{1}{t}\big)w - \frac{1}{\lambda t}\nabla\ell(w)$.

To apply Pegasos in the partial information case we could simply replace the gradient vector $\nabla\ell(w)$ with its estimation given in (8). However, our analysis shows that it is desirable to maintain an estimation vector $\widetilde{\nabla}\ell(w)$ with small magnitude. Since the magnitude of $\widetilde{\nabla}\ell(w) = 2\big(\hat{y} - y\big)v$ is order of $d\,\|w\|_1$, we would like to ensure that $\|w\|_1$ is always smaller than some threshold B. We achieve this goal by adding an additional projection step at the end of each Pegasos's iteration. Formally, the update is performed in two steps as follows

$$w \leftarrow \left(1 - \frac{1}{t}\right)w - \frac{1}{\lambda t}2\big(\hat{y} - y\big)v \tag{10}$$

$$w \leftarrow \mathop{\operatorname{argmin}}_{u:\,\|u\|_1 \le B} \|u - w\|_2 \tag{11}$$

ALGORITHM: AER(S,k)
INPUT: Training set S of size m, local budget $k \geq 2$ (with k even)
PARAMETER: $\lambda > 0$
INITIALIZATION: $w = 0 \in \mathbb{R}^d$; $w = w$; $t = 1$

for each $(x,y) \in S$
  $v = 0 \in \mathbb{R}^d$; $\hat{y} = 0$
  Choose C uniformly at random from all subsets of $[d]$ of size $\frac{k}{2}$
  for each $j \in C$
    $v_j = v_j + \frac{2}{k} d\, x_j$
  end
  for $r = 1, \ldots, k/2$
    sample i from $[d]$ based on $P(i) = \frac{|w_i|}{\|w\|_1}$ (if $w = 0$ set $P(i) = 1/d$)
    $\hat{y} = \hat{y} + \frac{2}{k} \operatorname{sgn}(w_i) \|w\|_1 x_i$
  end
  $w = \left(1 - \frac{1}{t}\right) w - \frac{2}{\lambda t}(\hat{y} - y) v$
  $w = \operatorname*{argmin}_{u : \|u\|_1 \leq B} \|u - w\|_2$
  $w = w + \frac{w}{m}$; $t = t + 1$
end

OUTPUT: w

Figure 2: An adaptation of the Pegasos algorithm to the local budget setting. Theorem 12 provides a performance guarantee for this algorithm.

where v and $\hat{y}$ are respectively defined by (6) and (7). The projection step (11) can be performed efficiently in time $O(d)$ using the technique described in Duchi et al. (2008). A pseudo-code of the resulting **A**ttribute **E**fficient **R**egression algorithm is given in Figure 2.

Note that the right-hand side of (10) is $w - \frac{1}{\lambda t} \nabla f$ for the function

$$f(w) = \tfrac{\lambda}{2} \|w\|_2^2 + 2(\hat{y} - y) \langle v, w \rangle \,. \tag{12}$$

This observation is used in the proof of the following result, providing convergence guarantees for AER.

Theorem 12 *Let $\mathcal{D}$ be a distribution on pairs $(x,y) \in \mathbb{R}^d \times \mathbb{R}$ such that $\|x\|_\infty \leq 1$ and $|y| \leq B$ with probability one. Let S be a training set of size m and let w be the output of AER(S,k) run with $\lambda = 12d\sqrt{\log(m)/(mk)}$. Then, there exists a constant $c > 0$ such that*

$$L_{\mathcal{D}}(w) \leq \min_{w : \|w\|_1 \leq B} L_D(w) + c\, dB^2 \sqrt{\frac{1}{km} \ln \frac{m}{\delta}}$$

holds with probability at least $1-\delta$ *over both the choice of the training set and the algorithm's own randomization*.

Proof Let $y_t, \hat{y}_t, v_t, w_t$ be the values of $y, \hat{y}, v, w$, respectively, at each iteration t of the AER algorithm. Moreover, let $\nabla_t = 2(\langle w_t, x_t\rangle - y_t)x_t$ and $\widetilde{\nabla}_t = 2(\hat{y}_t - y_t)v_t$. From the convexity of the squared loss, and taking expectation with respect to the algorithm's own randomization, we have that for any vector $w^\star$ such that $\|w^\star\|_1 \le B$,

$$\begin{aligned}
\mathbb{E}\left[\sum_{t=1}^m \big(\langle w_t, x_t\rangle - y_t\big)^2\right] - \sum_{t=1}^m \big(\langle w^\star, x_t\rangle - y_t\big)^2 &\le \mathbb{E}\left[\sum_{t=1}^m \langle \nabla_t, w_t - w^\star\rangle\right] \\
&= \mathbb{E}\left[\sum_{t=1}^m \langle \widetilde{\nabla}_t, w_t - w^\star\rangle\right] \\
&= \mathbb{E}\left[\sum_{t=1}^m 2(\hat{y}_t - y_t)\langle v_t, w_t - w^\star\rangle\right] .
\end{aligned}$$

For the first equality we used Lemma 11, which states that, conditioned on w_t, $\mathbb{E}\big[\widetilde{\nabla}_t\big] = \nabla_t$.

We now deterministically bound the random quantity inside the above expectation as follows

$$\begin{aligned}
\sum_{t=1}^m 2(\hat{y}_t - y_t)\langle v_t, w_t - w^\star\rangle &= \sum_{t=1}^m \left(\frac{\lambda}{2}\|w_t\|_2^2 + 2(\hat{y}_t - y_t)\langle v_t, w_t\rangle\right) \\
&\quad - \sum_{t=1}^m \left(\frac{\lambda}{2}\|w^\star\|_2^2 + 2(\hat{y}_t - y_t)\langle v_t, w^\star\rangle\right) + m\frac{\lambda}{2}\|w^\star\|_2^2 \\
&= \sum_{t=1}^m f_t(w_t) - \sum_{t=1}^m f_t(w^\star) + m\frac{\lambda}{2}\|w^\star\|_2^2
\end{aligned}$$

where $f_t(w) = \frac{\lambda}{2}\|w\|_2^2 + 2(\hat{y}_t - y_t)\langle v_t, w\rangle$ is the λ-strongly convex function defined in (12). Recalling that the right-hand side in the AER update (10) is equal to $w_t - \frac{1}{\lambda t}\nabla f_t(w_t)$, we can apply the following logarithmic regret bound for λ-strongly convex functions (Hazan et al., 2006; Kakade and Shalev-Shwartz, 2008)

$$\sum_{t=1}^m f_t(w_t) - \sum_{t=1}^m f_t(w^\star) \le \frac{1}{\lambda}\left(\max_t \|\nabla f_t(w_t)\|^2\right)\ln m$$

which remains valid also in the presence of the projection steps (11). Similarly to the analysis of Pegasos, and using our assumptions on $\|x_t\|_\infty$ and $|y_t|$, the norm of the gradient $\nabla f_t(w_t)$ is bounded as follows

$$\big\|\nabla f_t(w_t)\big\| \le \lambda\|w_t\| + 2\big|\hat{y}_t - y_t\big|\,\|v_t\| \le \lambda\|w_t\| + 4Bd\sqrt{\frac{2}{k}} .$$

In addition, it is easy to verify (e.g., using an iductive argument) that

$$\|w_t\| \le \frac{1}{\lambda}4Bd\sqrt{\frac{2}{k}},$$

which yields

$$\big\|\nabla f_t(w_t)\big\| \le 8Bd\sqrt{\frac{2}{k}} .$$

This gives the bound

$$\sum_{t=1}^{m} 2(\hat{y}_t - y_t)\langle v_t, w_t - w^\star\rangle \leq \frac{128(dB)^2}{\lambda k}\ln m + m\frac{\lambda}{2}\|w^\star\|_2^2 .$$

Choosing $\lambda = 16d\sqrt{\log(m)/(km)}$ and noting that $\|\cdot\|_2 \leq \|\cdot\|_1$ we get that

$$\sum_{t=1}^{m} 2(\hat{y}_t - y_t)\langle v_t, w_t - w^\star\rangle \leq 16dB^2\sqrt{\frac{m}{k}\ln m} .$$

The resulting bound is then

$$\mathbb{E}\left[\sum_{t=1}^{m}\big(\langle w_t, x_t\rangle - y_t\big)^2\right] \leq \sum_{t=1}^{m}\big(\langle w^\star, x_t\rangle - y_t\big)^2 + 16dB^2\sqrt{\frac{m}{k}\ln m} .$$

To conclude the proof, we apply the online-to-batch conversion of Cesa-Bianchi et al. (2004, Corollary 2) to the probability space that includes both the algorithm's own randomization and the product distribution from which the training set is drawn. Since $\big(\langle w, x_t\rangle - y_t\big)^2 \leq 4B^2$ for all w such that $\|w\|_1 \leq B$ (recall our assumptions on x_t and y_t), and using the convexity of the square loss, we obtain that

$$L_{\mathcal{D}}(w) \leq \inf_{w:\,\|w\|\leq B} L_{\mathcal{D}}(w) + 16dB^2\sqrt{\frac{1}{km}\ln m} + 4B^2\sqrt{\frac{2}{m}\ln\frac{1}{\delta}}$$

holds with probability at least $1-\delta$ with respect to all random events. ■

Note that for small values of k (which is the reasonable regime here) the bound for AER is much better than the bound for Baseline: ignoring logarithmic factors, instead of quadratic dependence on d, we have only linear dependence on d.

It is interesting to compare the bound for AER to the Lasso bound (2) for the full information case. As it can be seen, to achieve the same level of risk, AER needs a factor of d^2/k more examples than the full information Lasso.[4] Since each AER example uses only k attributes while each Lasso example uses all d attributes, the ratio between the total number of *attributes* AER needs and the number of attributes Lasso needs to achieve the same error is $O(d)$. Intuitively, when having d times total number of attributes, we can fully compensate for the partial information protocol.

However, in some situations even this extra d factor is not needed. Indeed, suppose we know that the vector $w^\star$, which minimizes the risk, is dense. That is, it satisfies $\|w^\star\|_1 \approx \sqrt{d}\,\|w^\star\|_2 \leq B$. In this case, by setting $\lambda = d^{3/2}\sqrt{\log(m)/(km)}$, and using the tighter bound $\|w^\star\|_2 \leq B/\sqrt{d}$ instead of $\|w^\star\|_2 \leq \|w^\star\|_1 \leq B$ in the proof of Theorem 12, we get a final bound of the form

$$L_{\mathcal{D}}(w) \leq L_{\mathcal{D}}(w^\star) + cB^2\sqrt{\frac{d}{km}\ln\frac{m}{\delta}} .$$

Therefore, the number of examples AER needs in order to achieve the same error as Lasso is only a factor d/k more than the number of examples Lasso uses. But, this implies that both AER and Lasso needs the same number of *attributes* in order to achieve the same level of error! Crucially, the above holds only if $w^\star$ is dense. When $w^\star$ is sparse we have $\|w^\star\|_1 \approx \|w^\star\|_2$ and then AER needs more attributes than Lasso.

4. We note that when $d = k$ we still do not recover the full information bound. However, it is possible to improve the analysis and replace the factor $d/\sqrt{k}$ with a factor $d\big(\max_t \|x_t\|_2\big)/k$.

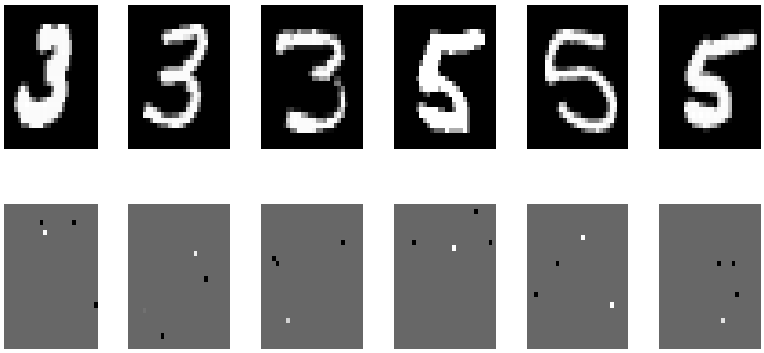


Figure 3: In the upper row six examples from the training set (of digits 3 and 5) are shown. In the lower row we show the same six examples, where only four randomly sampled pixels from each original image are displayed.

7. Experiments

We performed some experiments to test the behavior of our algorithm on the well-known MNIST digit recognition data set (Le Cun et al., 1998), which contains 70,000 images (28×28 pixels each) of the digits $0-9$. The advantages of this data set for our purposes is that it is not a small scale data set, has a reasonable dimensionality-to-data-size ratio, and the setting is clearly interpretable graphically. While this data set is designed for classification (e.g., recognizing the digit in the image), we can still apply our algorithms on it by regressing to the label.

First, to demonstrate the hardness of our settings, we provide in Figure 3 below some examples of images from the data set, in the full information setting and the partial information setting. The upper row contains six images from the data set, as available to a full information algorithm. A partial information algorithm, however, will have a much more limited access to these images. In particular, if the algorithm may only choose $k = 4$ pixels from each image, the same six images as available to it might look like the bottom row of Figure 3.

We began by looking at a data set composed of "3" vs. "5", where all the "3" digits were labeled as -1 and all the "5" digits were labeled as $+1$. We ran four different algorithms on this data set: the simple Baseline algorithm, AER, as well as ridge regression and Lasso for comparison (for Lasso, we solved (1) with $p = 1$). Both ridge regression and Lasso were run in the full information setting: Namely, they enjoyed full access to all attributes of all examples in the training set. The Baseline algorithm and AER, however, were given access to only four attributes from each training example.

We randomly split the data set into a training set and a test set (with the test set being 10% of the original data set). For each algorithm, parameter tuning was performed using 10-fold cross validation. Then, we ran the algorithm on increasingly long prefixes of the training set, and measured the average regression error $(\langle w,x\rangle - y)^2$ on the test set. The results (averaged over runs on 10 random train-test splits) are presented in Figure 4. In the upper plot, we see how the test regression error improves with the number of examples. The Baseline algorithm is highly unstable at the beginning, probably due to the ill-conditioning of the estimated covariance matrix, although it eventually stabilizes (to prevent a graphical mess at the left hand side of the figure, we removed the error bars from the corresponding plot). Its performance is worse than AER, completely in line with our earlier theoretical analysis.

The bottom plot of Figure 4 is similar, only that now the X-axis represents the accumulative number of attributes seen by each algorithm rather than the number of examples. For the partial-information algorithm, the graph ends at approximately 49,000 attributes, which is the total number of attributes accessed by the algorithm after running over all training examples, seeing $k = 4$ pixels from each example. However, for the full-information algorithms 49,000 attributes are already seen after just 62 examples. When we compare the algorithms in this way, we see that our AER algorithm achieves excellent performance for a given attribute budget, significantly better than the other 1-norm-based algorithms (Baseline and Lasso). Moreover, AER is even comparable to the full information 2-norm-based ridge regression algorithm, which performs best on this data set.

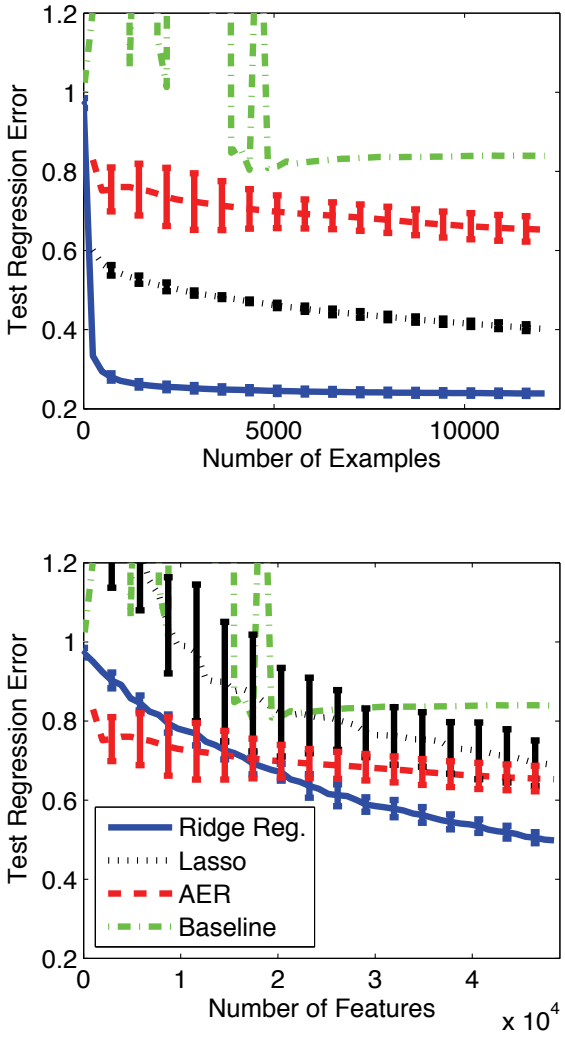


Figure 4: Test regression error for each one of the four algorithms (ridge regression, Lasso, AER, and Baseline), over increasing prefixes of the training set for "3" vs. "5". The results are averaged over 10 runs.

Finally, we tested the algorithms over 45 data sets generated from MNIST, one for each possible pair of digits. For each data set and each of 10 random train-test splits, we performed parameter tuning for each algorithm separately, and checked the average squared error on the test set. The median test errors over all data sets are presented in the table below.

		Test Error
Full Information	**Ridge**	0.110
	Lasso	0.222
Partial Information	**AER**	0.320
	Baseline	0.812

As can be seen, the AER algorithm manages to achieve good performance, not much worse than the full information Lasso algorithm. The Baseline algorithm, however, achieves a substantially worse performance, in line with our theoretical analysis above. We also calculated the test classification error of AER, that is, $\text{sign}(\langle w,x\rangle) \neq y$, and found out that AER, which can see only 4 pixels per image, usually performs only a little worse than the full information algorithms (ridge regression and Lasso), which enjoy full access to all 784 pixels in each image. In particular, the median test classification errors of AER, Lasso, and Ridge are 3.5%, 1.1%, and 1.3% respectively.

8. Discussion and Extensions

In this paper we have investigated three budgeted learning settings with different constraints on the way instance attributes may be accessed: a local constraint on each training example (local budget), a global constraint on the set of all training examples (global budget), and a constraint on each test example (prediction on a budget). In the local budget setting, we have introduced a simple and efficient algorithm, AER, that learns by accessing a pre-specified number of attributes from each training example. The AER algorithm comes with formal guarantees, is provably competitive with algorithms which enjoy full access to the data, and performs well in simple experiments. This result is complemented by a general lower bound for the global budget setting which is a factor d smaller than the upper bound achieved by our algorithm. We note that this gap has been recently closed by Hazan and Koren (2011), which in our local budget setting, show 1-norm and 2-norm-based algorithms for learning linear predictors using only $\widetilde{O}(d)$ attributes, thus matching our lower bound to within logarithmic factors.

Whereas AER is based on Pegasos, our adaptive sampling approach easily extends to other gradient-based algorithms. For example, generalized additive algorithms such as p-norm Perceptrons and Winnow—see, for example, Cesa-Bianchi and Lugosi (2006).

In contrast to the local/global budget settings, where we can learn efficiently by accessing few attributes of each training example, we showed that accessing a limited number of attributes at test time is a significantly harder setting. Indeed, we proved that is not possible to build an active linear predictor that uses two attributes of each test example and whose error is smaller than a certain constant, even when there exists a linear predictor achieving zero error on the same data source.

An obvious direction for future research is how to deal with loss functions other than the squared loss. In related work (Cesa-Bianchi et al., 2010), we developed a technique which allows us to deal with arbitrary analytic loss functions. However, in the setting of this paper, those techniques would lead to sample complexity bounds which are exponential in d. Another interesting extension we are considering is connecting our results to the field of privacy-preserving learning (Dwork,

2008), where the goal is to exploit the attribute efficiency property in order to prevent acquisition of information about individual data instances.

Acknowledgments

We would like to thank the anonymous reviewers for their detailed and helpful comments. N. Cesa-Bianchi gratefully acknowledges partial support by the PASCAL2 Network of Excellence under EC grant no. 216886. This publication only reflects the authors' views.

Appendix A. Proof of Theorem 1

The outline of the proof is as follows. We define a specific distribution such that only one "good" feature is slightly correlated with the label. We then show that if some algorithm learns a linear predictor with an extra risk of at most ε, then it must know the value of the good feature. Next, we construct a variant of a multi-armed bandit problem out of our distribution and show that a good learner can yield a good prediction strategy. Finally, we adapt a lower bound for the multi-armed bandit problem given in Auer et al. (2003), to conclude that the number k of attributes viewed by a good learner must satisfy $k = \Omega\left(\frac{d}{\varepsilon}\right)$.

A.1 The Distribution

We generate a joint distribution over $\mathbb{R}^d \times \mathbb{R}$ as follows. Choose some $j \in [d]$. First, we generate $y_1, y_2, \ldots \in \{\pm 1\}$ i.i.d. according to $\mathbb{P}[y_t = 1] = \mathbb{P}[y_t = -1] = \frac{1}{2}$. Given j and y_t, $x_t \in \{\pm 1\}$ is generated according to $\mathbb{P}[x_{t,i} = y_t] = \frac{1}{2} + \mathbb{1}\{i = j\}p$ where $p > 0$ is chosen later. Denote by $\mathbb{P}_j$ the distribution mentioned above assuming the "good" feature is j. Also denote by $\mathbb{P}_u$ the uniform distribution over $\{\pm 1\}^{d+1}$. Analogously, we denote by $\mathbb{E}_j$ and $\mathbb{E}_u$ expectations w.r.t. $\mathbb{P}_j$ and $\mathbb{P}_u$.

A.2 A Good Regressor "Knows" j

We now show that if we have a good linear regressor than we can know the value of j. It is easy to see that the optimal linear predictor under the distribution $\mathbb{P}_j$ is $w^\star = 2p\,e^j$, and the risk of $w^\star$ is

$$L_{\mathbb{P}_j}(w^\star) = \mathbb{E}_j\left[(\langle w^\star, x\rangle - y)^2\right] = \left(\tfrac{1}{2}+p\right)(1-2p)^2 + \left(\tfrac{1}{2}-p\right)(1+2p)^2 = 1+4p^2-8p^2 = 1-4p^2\,.$$

The risk of an arbitrary weight vector w under P_j is

$$L_{\mathbb{P}_j}(w) = \mathbb{E}_j\left[(\langle w, x\rangle - y)\right]^2 = \sum_{i\neq j} w_i^2 + \mathbb{E}_j\left[(w_j x_j - y)^2\right] = \sum_{i\neq j} w_i^2 + w_j^2 + 1 - 4pw_j\,.$$

Suppose that $L_{\mathbb{P}_j}(w) - L_{\mathbb{P}_j}(w^\star) < \varepsilon$. This implies that:

1. For all $i \neq j$ we have $w_i^2 < \varepsilon$, or equivalently, $|w_i| < \sqrt{\varepsilon}$.
2. $1 + w_j^2 - 4pw_j - (1-4p^2) < \varepsilon$ and thus $|w_j - 2p| < \sqrt{\varepsilon}$ which gives $|w_j| > 2p - \sqrt{\varepsilon}$.

By choosing $p = \sqrt{\varepsilon}$, the above implies that we can identify the value of j from any w whose risk is strictly smaller than $L_{\mathbb{P}_j}(w^\star) + \varepsilon$.

A.3 Constructing A Variant Of A Multi-Armed Bandit Problem

We now construct a variant of the multi-armed bandit problem out of the distribution $\mathbb{P}_j$. Each coordinate $i \in \{1,\ldots,d\}$ is an arm and the reward of pulling i at time t is $\mathbb{1}\{x_{N_{i,t},i} = y_{N_{i,t}}\} \in \{0,1\}$, where $N_{i,t}$ denotes the random number of times arm i has been pulled in the first t plays. Hence the expected reward of pulling i is $\frac{1}{2} + \mathbb{1}\{i=j\}p$. At the end of each round t the player observes $x_{N_{i,t},i}$ and $y_{N_{i,t}}$.

A.4 A Good Learner Yields A Bandit Strategy

Suppose that we have a learner that, for any $j = 1,\ldots,d$, can learn a linear predictor with $L_{\mathbb{P}_j}(w) - L_{\mathbb{P}_j}(w^\star) < \varepsilon$ using k attributes. Since we have shown that once $L_{\mathbb{P}_j}(w) - L_{\mathbb{P}_j}(w^\star) < \varepsilon$ we know the value of j, we can construct a strategy for the multi-armed bandit problem in a straightforward way. Simply use the first m examples to learn w and from then on always pull the arm j. The expected reward of this strategy under any $\mathbb{P}_j$ after $T \geq k$ plays is at least

$$\frac{k}{2} + (T-k)\left(\frac{1}{2}+p\right) = \frac{T}{2} + (T-k)p\,. \tag{13}$$

A.5 An Upper Bound On the Reward Of Any Bandit Strategy

Recall that under distribution $\mathbb{P}_j$ the expected reward for pulling arm I is $\frac{1}{2} + p\mathbb{1}\{I=j\}$. Hence, the total expected reward of a player that runs for T rounds is upper bounded by $\frac{1}{2}T + p\mathbb{E}_j[N_j]$, where $N_j = N_{j,T}$ is the overall number of pulls of arm j. Moreover, at the end of each round t the player observes $x_{s,i}$ and y_s, where $s = N_{i,t}$. This allows the player to compute the value of the reward for the current play. For any s, note that y_s is observed whenever some arm i is pulled for the s-th time. However, since $\mathbb{P}_j\big[x_{i,s} = y_s\big] = \mathbb{P}_j\big[x_{i,s} = y_s \mid y_s\big]$ for all i (including $i=j$), the knowledge of y_s does not provide any information about the distribution of rewards for arm i. Therefore, without loss of generality, we can assume that at each play the bandit strategy observes only the obtained binary reward. This implies that our bandit construction is identical to the one used in the proof of Theorem 5.1 in Auer et al. (2003). In particular, for any bandit strategy there exists some arm j such that the expected reward of the strategy under distribution $\mathbb{P}_j$ is at most

$$\frac{T}{2} + p\left(\frac{T}{d} + T\sqrt{-\frac{T}{d}\ln(1-4p^2)}\right) \leq \frac{T}{2} + p\left(\frac{T}{d} + T\sqrt{\frac{6T}{d}p^2}\right) \tag{14}$$

where we used the inequality $-\ln(1-q) \leq \frac{3}{2}q$ for $q \in [0,1/4]$. Note that $q = 4p^2 = 4\varepsilon \in [0,1/4]$ when $\varepsilon \leq 1/16$.

A.6 Concluding The Proof

Take a learning algorithm that finds an ε-good predictor using k attributes. Since the reward of the strategy based on this learning algorithm cannot exceed the upper bound given in (14), from (13) we obtain that

$$\frac{T}{2} + (T-k)p \leq \frac{T}{2} + p\left(\frac{T}{d} + T\sqrt{\frac{6T}{d}p^2}\right)$$

which solved for k gives

$$k \geq T\left(1 - \frac{1}{d} - \sqrt{\frac{6T}{d}p^2}\right) .$$

Since we assume $d \geq 4$, choosing $T = \lfloor d/(96p^2) \rfloor$, and recalling $p^2 = \varepsilon$, gives

$$k \geq \frac{T}{2} = \frac{1}{2}\left\lfloor \frac{d}{96\varepsilon} \right\rfloor .$$

References

P. Auer, N. Cesa-Bianchi, Y. Freund, and R.E. Schapire. The nonstochastic multiarmed bandit problem. *SIAM Journal on Computing*, 32, 2003.

M-F Balcan, A. Beygelzimer, and J. Langford. Agnostic active learning. In *ICML*, 2006.

S. Ben-David and E. Dichterman. Learning with restricted focus of attention. *JCSS: Journal of Computer and System Sciences*, 56, 1998.

A. Beygelzimer, S. Dasgupta, and J. Langford. Importance weighted active learning. In *ICML*, 2009.

R. Calderbank, S. Jafarpour, and R. Schapire. Compressed learning: Universal sparse dimensionality reduction and learning in the measurement domain. Manuscript, 2009.

N. Cesa-Bianchi and G. Lugosi. *Prediction, Learning, and Games*. Cambridge University Press, 2006.

N. Cesa-Bianchi, A. Conconi, and C. Gentile. On the generalization ability of on-line learning algorithms. *IEEE Transactions on Information Theory*, 50(9):2050–2057, September 2004.

N. Cesa-Bianchi, S. Shalev-Shwartz, and O. Shamir. Online learning of noisy data with kernels. In *COLT*, 2010.

D. Cohn, L. Atlas, and R. Ladner. Improving generalization with active learning. *Machine Learning*, 15:201–221, 1994.

K. Deng, C. Bourke, S. Scott, J. Sunderman, and Y. Zheng. Bandit-based algorithms for budgeted learning. In *ICDM*, 2007.

L. Devroye, L. Györfi, and G. Lugosi. *A Probabilistic Theory of Pattern Recognition*. Springer, 1996.

J. Duchi, S. Shalev-Shwartz, Y. Singer, and T. Chandra. Efficient projections onto the ℓ_1-ball for learning in high dimensions. In *ICML*, 2008.

C. Dwork. Differential privacy: A survey of results. In M. Agrawal, D.-Z. Du, Z. Duan, and A. Li, editors, *TAMC*, volume 4978 of *Lecture Notes in Computer Science*, pages 1–19. Springer, 2008.

R. Greiner, A. J. Grove, and D. Roth. Learning cost-sensitive active classifiers. *Artificial Intelligence*, 139(2):137–174, 2002.

S. Guha and K. Munagala. Approximation algorithms for budgeted learning problems. In *STOC*, 2007.

S. Hanneke. A bound on the label complexity of agnostic active learning. In *ICML*, 2007.

S. Hanneke. Adaptive rates of convergence in active learning. In *COLT*, 2009.

D. Haussler. Decision theoretic generalizations of the PAC model for neural net and other learning applications. *Information and Computation*, 100(1):78–150, 1992.

E. Hazan and T. Koren. Optimal algorithms for ridge and Lasso regression with partially observed attributes. *CoRR*, abs/1108.4559, 2011.

E. Hazan, A. Kalai, S. Kale, and A. Agarwal. Logarithmic regret algorithms for online convex optimization. In *COLT*, 2006.

S.M. Kakade and S. Shalev-Shwartz. Mind the duality gap: Logarithmic regret algorithms for online optimization. In *Advances in Neural Information Processing Systems 22*, 2008.

S.M. Kakade, K. Sridharan, and A. Tewari. On the complexity of linear prediction: Risk bounds, margin bounds, and regularization. In *NIPS*, 2008.

A. Kapoor and R. Greiner. Learning and classifying under hard budgets. In *ECML*, 2005a.

A. Kapoor and R. Greiner. Budgeted learning of bounded active classifiers. In *Proceedings of the ACM SIGKDD Workshop on Utility-Based Data Mining*, 2005b.

Y. L. Le Cun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of IEEE*, 86(11):2278–2324, November 1998.

O. Madani, D.J. Lizotte, and R. Greiner. Active model selection. In *UAI*, 2004.

S. Shalev-Shwartz, Y. Singer, and N. Srebro. Pegasos: Primal Estimated sub-GrAdient SOlver for SVM. In *ICML*, 2007.

R. Tibshirani. Regression shrinkage and selection via the lasso. *J. Royal. Statist. Soc B.*, 58(1): 267–288, 1996.

V. N. Vapnik. *Statistical Learning Theory*. Wiley, 1998.

S. Zhou, J. Lafferty, and L. Wasserman. Compressed and privacy-sensitive sparse regression. *IEEE Transactions on Information Theory*, 55(2):846–866, 2009.

Journal of Machine Learning Research 12 (2011) 2879-2904 Submitted 1/11; Revised 7/11; Published 10/11

Convergence Rates of Efficient Global Optimization Algorithms

Adam D. Bull A.BULL@STATSLAB.CAM.AC.UK
Statistical Laboratory
University of Cambridge
Cambridge, CB3 0WB, UK

Editor: Manfred Opper

Abstract

In the efficient global optimization problem, we minimize an unknown function f, using as few observations $f(x)$ as possible. It can be considered a continuum-armed-bandit problem, with noiseless data, and simple regret. Expected-improvement algorithms are perhaps the most popular methods for solving the problem; in this paper, we provide theoretical results on their asymptotic behaviour.

Implementing these algorithms requires a choice of Gaussian-process prior, which determines an associated space of functions, its reproducing-kernel Hilbert space (RKHS). When the prior is fixed, expected improvement is known to converge on the minimum of any function in its RKHS. We provide convergence rates for this procedure, optimal for functions of low smoothness, and describe a modified algorithm attaining optimal rates for smoother functions.

In practice, however, priors are typically estimated sequentially from the data. For standard estimators, we show this procedure may never find the minimum of f. We then propose alternative estimators, chosen to minimize the constants in the rate of convergence, and show these estimators retain the convergence rates of a fixed prior.

Keywords: convergence rates, efficient global optimization, expected improvement, continuum-armed bandit, Bayesian optimization

1. Introduction

Suppose we wish to minimize a continuous function $f : X \to \mathbb{R}$, where X is a compact subset of $\mathbb{R}^d$. Observing $f(x)$ is costly (it may require a lengthy computer simulation or physical experiment), so we wish to use as few observations as possible. We know little about the shape of f; in particular we will be unable to make assumptions of convexity or unimodality. We therefore need a *global* optimization algorithm, one which attempts to find a global minimum.

Many standard global optimization algorithms exist, including genetic algorithms, multistart, and simulated annealing (Pardalos and Romeijn, 2002), but these algorithms are designed for functions that are cheap to evaluate. When f is expensive, we need an *efficient* algorithm, one which will choose its observations to maximize the information gained.

We can consider this a continuum-armed-bandit problem (Srinivas et al., 2010, and references therein), with noiseless data, and loss measured by the simple regret (Bubeck et al., 2009). At time n, we choose a design point $x_n \in X$, make an observation $z_n = f(x_n)$, and then report a point x_n^* where we believe $f(x_n^*)$ will be low. Our goal is to find a strategy for choosing the x_n and x_n^*, in terms of previous observations, so as to minimize $f(x_n^*)$.

We would like to find a strategy which can guarantee convergence: for functions f in some smoothness class, $f(x_n^*)$ should tend to $\min f$, preferably at some fast rate. The simplest method

would be to fix a sequence of x_n in advance, and set $x_n^* = \arg\min \hat{f}_n$, for some approximation $\hat{f}_n$ to f. We will show that if $\hat{f}_n$ converges in supremum norm at the optimal rate, then $f(x_n^*)$ also converges at its optimal rate. However, while this strategy gives a good worst-case bound, on average it is clearly a poor method of optimization: the design points x_n are completely independent of the observations z_n.

We may therefore ask if there are more efficient methods, with better average-case performance, that nevertheless provide good guarantees of convergence. The difficulty in designing such a method lies in the trade-off between *exploration* and *exploitation*. If we exploit the data, observing in regions where f is known to be low, we will be more likely to find the optimum quickly; however, unless we explore every region of X, we may not find it at all (Macready and Wolpert, 1998).

Initial attempts at this problem include work on Lipschitz optimization (summarized in Hansen et al., 1992) and the DIRECT algorithm (Jones et al., 1993), but perhaps the best-known strategy is expected improvement. It is sometimes called Bayesian optimization, and first appeared in Močkus (1974) as a Bayesian decision-theoretic solution to the problem. Contemporary computers were not powerful enough to implement the technique in full, and it was later popularized by Jones et al. (1998), who provided a computationally efficient implementation. More recently, it has also been called a knowledge-gradient policy by Frazier et al. (2009). Many extensions and alterations have been suggested by further authors; a good summary can be found in Brochu et al. (2010).

Expected improvement performs well in experiments (Osborne, 2010, §9.5), but little is known about its theoretical properties. The behaviour of the algorithm depends crucially on the Gaussian process prior π chosen for f. Each prior has an associated space of functions $\mathcal{H}$, its reproducing-kernel Hilbert space. $\mathcal{H}$ contains all functions $X \to \mathbb{R}$ as smooth as a posterior mean of f, and is the natural space in which to study questions of convergence.

Vazquez and Bect (2010) show that when π is a fixed Gaussian process prior of finite smoothness, expected improvement converges on the minimum of any $f \in \mathcal{H}$, and almost surely for f drawn from π. Grunewalder et al. (2010) bound the convergence rate of a computationally infeasible version of expected improvement: for priors π of smoothness ν, they show convergence at a rate $O^*(n^{-(\nu \wedge 0.5)/d})$ on f drawn from π. We begin by bounding the convergence rate of the feasible algorithm, and show convergence at a rate $O^*(n^{-(\nu \wedge 1)/d})$ on all $f \in \mathcal{H}$. We go on to show that a modification of expected improvement converges at the near-optimal rate $O^*(n^{-\nu/d})$.

For practitioners, however, these results are somewhat misleading. In typical applications, the prior is not held fixed, but depends on parameters estimated sequentially from the data. This process ensures the choice of observations is invariant under translation and scaling of f, and is believed to be more efficient (Jones et al., 1998, §2). It has a profound effect on convergence, however: Locatelli (1997, §3.2) shows that, for a Brownian motion prior with estimated parameters, expected improvement may not converge at all.

We extend this result to more general settings, showing that for standard priors with estimated parameters, there exist smooth functions f on which expected improvement does not converge. We then propose alternative estimates of the prior parameters, chosen to minimize the constants in the convergence rate. We show that these estimators give an automatic choice of parameters, while retaining the convergence rates of a fixed prior.

Table 1 summarizes the notation used in this paper. We say $f : \mathbb{R}^d \to \mathbb{R}$ is a bump function if f is infinitely differentiable and of compact support, and $f : \mathbb{R}^d \to \mathbb{C}$ is Hermitian if $\overline{f(x)} = f(-x)$. We use the Landau notation $f = O(g)$ to denote $\limsup |f/g| < \infty$, and $f = o(g)$ to denote $f/g \to 0$. If $g = O(f)$, we say $f = \Omega(g)$, and if both $f = O(g)$ and $f = \Omega(g)$, we say $f = \Theta(g)$. If further

$f/g \to 1$, we say $f \sim g$. Finally, if f and g are random, and $\mathbb{P}(\sup|f/g| \leq M) \to 1$ as $M \to \infty$, we say $f = O_p(g)$.

In Section 2, we briefly describe the expected-improvement algorithm, and detail our assumptions on the priors used. We state our main results in Section 3, and discuss implications for further work in Section 4. Finally, we give proofs in Appendix A.

2. Expected Improvement

Suppose we wish to minimize an unknown function f, choosing design points x_n and estimated minima x_n^* as in the introduction. If we pick a prior distribution π for f, representing our beliefs about the unknown function, we can describe this problem in terms of decision theory. Let $(\Omega, \mathcal{F}, \mathbb{P})$ be a probability space, equipped with a random process f having law π. A strategy u is a collection of random variables (x_n), (x_n^*) taking values in X. Set $z_n := f(x_n)$, and define the filtration $\mathcal{F}_n := \sigma(x_i, z_i : i \leq n)$. The strategy u is valid if x_n is conditionally independent of f given $\mathcal{F}_{n-1}$, and likewise x_n^* given $\mathcal{F}_n$. (Note that we allow random strategies, provided they do not depend on unknown information about f.)

When taking probabilities and expectations we will write $\mathbb{P}_\pi^u$ and $\mathbb{E}_\pi^u$, denoting the dependence on both the prior π and strategy u. The average-case performance at some future time N is then given by the expected loss,

$$\mathbb{E}_\pi^u[f(x_N^*) - \min f],$$

and our goal, given π, is to choose the strategy u to minimize this quantity.

2.1 Bayesian Optimization

For $N > 1$ this problem is very computationally intensive (Osborne, 2010, §6.3), but we can solve a simplified version of it. First, we restrict the choice of x_n^* to the previous design points $x_1, \ldots, x_n$. (In practice this is reasonable, as choosing an x_n^* we have not observed can be unreliable.) Secondly, rather than finding an optimal strategy for the problem, we derive the myopic strategy: the strategy which is optimal if we always assume we will stop after the next observation. This strategy is suboptimal (Ginsbourger et al., 2008, §3.1), but performs well, and greatly simplifies the calculations involved.

In this setting, given $\mathcal{F}_n$, if we are to stop at time n we should choose $x_n^* := x_{i_n^*}$, where $i_n^* := \arg\min_{1,\ldots,n} z_i$. (In the case of ties, we may pick any minimizing i_n^*.) We then suffer a loss $z_n^* - \min f$, where $z_n^* := z_{i_n^*}$. Were we to observe at x_{n+1} before stopping, the expected loss would be

$$\mathbb{E}_\pi^u[z_{n+1}^* - \min f \mid \mathcal{F}_n],$$

so the myopic strategy should choose x_{n+1} to minimize this quantity. Equivalently, it should maximize the expected improvement over the current loss,

$$EI_n(x_{n+1}; \pi) := \mathbb{E}_\pi^u[z_n^* - z_{n+1}^* \mid \mathcal{F}_n] = \mathbb{E}_\pi^u[(z_n^* - z_{n+1})^+ \mid \mathcal{F}_n], \tag{1}$$

where $x^+ = \max(x, 0)$.

So far, we have merely replaced one optimization problem with another. However, for suitable priors, EI_n can be evaluated cheaply, and thus maximized by standard techniques. The expected-improvement algorithm is then given by choosing x_{n+1} to maximize (1).

Section 1	
f	unknown function $X \to \mathbb{R}$ to be minimized
X	compact subset of $\mathbb{R}^d$ to minimize over
d	number of dimensions to minimize over
x_n	points in X at which f is observed
z_n	observations $z_n = f(x_n)$ of f
x_n^*	estimated minimum of f, given $z_1, \ldots, z_n$
Section 2.1	
π	prior distribution for f
u	strategy for choosing x_n, x_n^*
$\mathcal{F}_n$	filtration $\mathcal{F}_n = \sigma(x_i, z_i : i \leq n)$
z_n^*	best observation $z_n^* = \min_{i=1,\ldots,n} z_i$
EI_n	expected improvement given $\mathcal{F}_n$
Section 2.2	
μ, σ^2	global mean and variance of Gaussian-process prior π
K	underlying correlation kernel for π
K_θ	correlation kernel for π with length-scales θ
ν, α	smoothness parameters of K
$\hat{\mu}_n, \hat{f}_n, s_n^2, \hat{R}_n^2$	quantities describing posterior distribution of f given $\mathcal{F}_n$
Section 2.3	
$EI(\pi)$	expected improvement strategy with fixed prior
$\hat{\sigma}_n^2, \hat{\theta}_n$	estimates of prior parameters σ^2, θ
c_n	rate of decay of $\hat{\sigma}_n^2$
θ^L, θ^U	bounds on $\hat{\theta}_n$
$EI(\hat{\pi})$	expected improvement strategy with estimated prior
Section 3.1	
$\mathcal{H}_\theta(S)$	reproducing-kernel Hilbert space of K_θ on S
$H^s(D)$	Sobolev Hilbert space of order s on D
Section 3.2	
L_n	loss suffered over an RKHS ball after n steps
Section 3.3	
$EI(\tilde{\pi})$	expected improvement strategy with robust estimated prior
Section 3.4	
$EI(\cdot, \varepsilon)$	ε-greedy expected improvement strategies

Table 1: Notation

2.2 Gaussian Process Models

We still need to choose a prior π for f. Typically, we model f as a stationary Gaussian process: we consider the values $f(x)$ to be jointly Gaussian, with mean and covariance

$$\mathbb{E}_\pi[f(x)] = \mu, \quad \mathbb{C}\mathrm{ov}_\pi[f(x), f(y)] = \sigma^2 K_\theta(x-y). \tag{2}$$

$\mu \in \mathbb{R}$ is the global mean of f; we place a flat prior on μ, reflecting our uncertainty over the location of f.

$\sigma > 0$ is the global scale of variation of f, and $K_\theta : \mathbb{R}^d \to \mathbb{R}$ its correlation kernel, governing the local properties of f. In the following, we will consider kernels

$$K_\theta(t_1, \ldots, t_d) \coloneqq K(t_1/\theta_1, \ldots, t_d/\theta_d), \tag{3}$$

for an underlying kernel K with $K(0) = 1$. (Note that we can always satisfy this condition by suitably scaling K and σ.) The $\theta_i > 0$ are the length-scales of the process: two values $f(x)$ and $f(y)$ will be highly correlated if each $x_i - y_i$ is small compared with θ_i. For now, we will assume the parameters σ and θ are fixed in advance.

For (2) and (3) to define a consistent Gaussian process, K must be a symmetric positive-definite function. We will also make the following assumptions.

Assumption 1. *K is continuous and integrable.*

K thus has Fourier transform

$$\widehat{K}(\xi) \coloneqq \int_{\mathbb{R}^d} e^{-2\pi i \langle x, \xi \rangle} K(x)\, dx,$$

and by Bochner's theorem, $\widehat{K}$ is non-negative and integrable.

Assumption 2. *$\widehat{K}$ is isotropic and radially non-increasing.*

In other words, $\widehat{K}(x) = \widehat{k}(\|x\|)$ for a non-increasing function $\widehat{k} : [0, \infty) \to [0, \infty)$; as a consequence, K is isotropic.

Assumption 3. *As $x \to \infty$, either:*

(i) $\widehat{K}(x) = \Theta(\|x\|^{-2\nu-d})$ *for some* $\nu > 0$*; or*

(ii) $\widehat{K}(x) = O(\|x\|^{-2\nu-d})$ *for all* $\nu > 0$ *(we will then say that* $\nu = \infty$*).*

Note the condition $\nu > 0$ is required for $\widehat{K}$ to be integrable.

Assumption 4. *K is C^k, for k the largest integer less than 2ν, and at the origin, K has k-th order Taylor approximation P_k satisfying*

$$|K(x) - P_k(x)| = O\left(\|x\|^{2\nu}(-\log\|x\|)^{2\alpha}\right)$$

as $x \to 0$, for some $\alpha \geq 0$.

When $\alpha = 0$, this is just the condition that K be 2ν-Hölder at the origin; when $\alpha > 0$, we instead require this condition up to a log factor.

The rate ν controls the smoothness of functions from the prior: almost surely, f has continuous derivatives of any order $k < \nu$ (Adler and Taylor, 2007, §1.4.2). Popular kernels include the Matérn class,

$$K^\nu(x) := \frac{2^{1-\nu}}{\Gamma(\nu)} \left(\sqrt{2\nu}\|x\|\right)^\nu k_\nu \left(\sqrt{2\nu}\|x\|\right), \quad \nu \in (0,\infty),$$

where k_ν is a modified Bessel function of the second kind, and the Gaussian kernel,

$$K^\infty(x) := e^{-\frac{1}{2}\|x\|^2},$$

obtained in the limit $\nu \to \infty$ (Rasmussen and Williams, 2006, §4.2). Between them, these kernels cover the full range of smoothness $0 < \nu \leq \infty$. Both kernels satisfy Assumptions 1–4 for the ν given; $\alpha = 0$ except for the Matérn kernel with $\nu \in \mathbb{N}$, where $\alpha = \frac{1}{2}$ (Abramowitz and Stegun, 1965, §9.6).

Having chosen our prior distribution, we may now derive its posterior. We find

$$f(x) \mid z_1, \ldots, z_n \sim N\left(\hat{f}_n(x;\theta), \sigma^2 s_n^2(x;\theta)\right),$$

where

$$\hat{\mu}_n(\theta) := \frac{1^T V^{-1} z}{1^T V^{-1} 1}, \tag{4}$$

$$\hat{f}_n(x;\theta) := \hat{\mu}_n + v^T V^{-1}(z - \hat{\mu}_n 1), \tag{5}$$

and

$$s_n^2(x;\theta) := 1 - v^T V^{-1} v + \frac{(1 - 1^T V^{-1} v)^2}{1^T V^{-1} 1}, \tag{6}$$

for $z = (z_i)_{i=1}^n$, $V = (K_\theta(x_i - x_j))_{i,j=1}^n$, and $v = (K_\theta(x - x_i))_{i=1}^n$ (Santner et al., 2003, §4.1.3). Equivalently, these expressions are the best linear unbiased predictor of $f(x)$ and its variance, as given in Jones et al. (1998, §2). We will also need the reduced sum of squares,

$$\hat{R}_n^2(\theta) := (z - \hat{\mu}_n 1)^T V^{-1} (z - \hat{\mu}_n 1). \tag{7}$$

2.3 Expected Improvement Strategies

Under our assumptions on π, we may now derive an analytic form for (1), as in Jones et al. (1998, §4.1). We obtain

$$EI_n(x_{n+1};\pi) = \rho\left(z_n^* - \hat{f}_n(x_{n+1};\theta), \sigma s_n(x_{n+1};\theta)\right), \tag{8}$$

where

$$\rho(y,s) := \begin{cases} y\Phi(y/s) + s\varphi(y/s), & s > 0, \\ \max(y,0), & s = 0, \end{cases} \tag{9}$$

and Φ and φ are the standard normal distribution and density functions respectively.

For a prior π as above, expected improvement chooses x_{n+1} to maximize (8), but this does not fully define the strategy. Firstly, we must describe how the strategy breaks ties, when more than one $x \in X$ maximizes EI_n. In general, this will not affect the behaviour of the algorithm, so we allow any choice of x_{n+1} maximizing (8).

Secondly, we must say how to choose x_1, as the above expressions are undefined when $n = 0$. In fact, Jones et al. (1998, §4.2) find that expected improvement can be unreliable given few data points, and recommend that several initial design points be chosen in a random quasi-uniform arrangement. We will therefore assume that until some fixed time k, points $x_1, \ldots, x_k$ are instead chosen by some (potentially random) method independent of f. We thus obtain the following strategy.

Definition 1. *An EI(π) strategy chooses:*

(i) initial design points $x_1, \ldots, x_k$ independently of f; and

(ii) further design points x_{n+1} $(n \geq k)$ from the maximizers of (8).

So far, we have not considered the choice of parameters σ and θ. While these can be fixed in advance, doing so requires us to specify characteristic scales of the unknown function f, and causes expected improvement to behave differently on a rescaling of the same function. We would prefer an algorithm which could adapt automatically to the scale of f.

A natural approach is to take maximum likelihood estimates of the parameters, as recommended by Jones et al. (1998, §2). Given θ, the MLE $\hat{\sigma}_n^2 = \hat{R}_n^2(\theta)/n$; for full generality, we will allow any choice $\hat{\sigma}_n^2 = c_n \hat{R}_n^2(\theta)$, where $c_n = o(1/\log n)$. Estimates of θ, however, must be obtained by numerical optimization. As θ can vary widely in scale, this optimization is best performed over $\log\theta$; as the likelihood surface is typically multimodal, this requires the use of a global optimizer. We must therefore place (implicit or explicit) bounds on the allowed values of $\log\theta$. We have thus described the following strategy.

Definition 2. *Let $\hat{\pi}_n$ be a sequence of priors, with parameters $\hat{\sigma}_n$, $\hat{\theta}_n$ satisfying:*

(i) $\hat{\sigma}_n^2 = c_n \hat{R}_n^2(\hat{\theta}_n)$ for constants $c_n > 0$, $c_n = o(1/\log n)$; and

(ii) $\theta^L \leq \hat{\theta}_n \leq \theta^U$ for constants $\theta^L, \theta^U \in \mathbb{R}_+^d$.

An EI($\hat{\pi}$) strategy satisfies Definition 1, replacing π with $\hat{\pi}_n$ in (8).

3. Convergence Rates

To discuss convergence, we must first choose a smoothness class for the unknown function f. Each kernel K_θ is associated with a space of functions $\mathcal{H}_\theta(X)$, its reproducing-kernel Hilbert space (RKHS) or native space. $\mathcal{H}_\theta(X)$ contains all functions $X \to \mathbb{R}$ as smooth as a posterior mean of f, and is the natural space to study convergence of expected-improvement algorithms, allowing a tractable analysis of their asymptotic behaviour.

3.1 Reproducing-Kernel Hilbert Spaces

Given a symmetric positive-definite kernel K on $\mathbb{R}^d$, set $k_x(t) = K(t-x)$. For $S \subseteq \mathbb{R}^d$, let $\mathcal{E}(S)$ be the space of functions $S \to \mathbb{R}$ spanned by the k_x, for $x \in S$. Furnish $\mathcal{E}(S)$ with the inner product defined by

$$\langle k_x, k_y \rangle := K(x-y).$$

The completion of $\mathcal{E}(S)$ under this inner product is the reproducing-kernel Hilbert space $\mathcal{H}(S)$ of K on S. The members $f \in \mathcal{H}(S)$ are abstract objects, but we can identify them with functions $f : S \to \mathbb{R}$ through the reproducing property,

$$f(x) = \langle f, k_x \rangle,$$

which holds for all $f \in \mathcal{E}(S)$. See Aronszajn (1950), Berlinet and Thomas-Agnan (2004), Wendland (2005) and van der Vaart and van Zanten (2008).

We will find it convenient also to use an alternative characterization of $\mathcal{H}(S)$. We begin by describing $\mathcal{H}(\mathbb{R}^d)$ in terms of Fourier transforms. Let $\widehat{f}$ denote the Fourier transform of a function $f \in L^2$. The following result is stated in Parzen (1963, §2), and proved in Wendland (2005, §10.2); we give a short proof in Appendix A.

Lemma 1. *$\mathcal{H}(\mathbb{R}^d)$ is the space of real continuous $f \in L^2(\mathbb{R}^d)$ whose norm*

$$\|f\|^2_{\mathcal{H}(\mathbb{R}^d)} := \int \frac{|\widehat{f}(\xi)|^2}{\widehat{K}(\xi)}\, d\xi$$

is finite, taking $0/0 = 0$.

We may now describe $\mathcal{H}(S)$ in terms of $\mathcal{H}(\mathbb{R}^d)$.

Lemma 2 (Aronszajn, 1950, §1.5)**.** *$\mathcal{H}(S)$ is the space of functions $f = g|_S$ for some $g \in \mathcal{H}(\mathbb{R}^d)$, with norm*

$$\|f\|_{\mathcal{H}(S)} := \inf_{g|_S = f} \|g\|_{\mathcal{H}(\mathbb{R}^d)},$$

and there is a unique g minimizing this expression.

These spaces are in fact closely related to the Sobolev Hilbert spaces of functional analysis. Say a domain $D \subseteq \mathbb{R}^d$ is Lipschitz if its boundary is locally the graph of a Lipschitz function (see Tartar, 2007, §12, for a precise definition). For such a domain D, the Sobolev Hilbert space $H^s(D)$ is the space of functions $f : D \to \mathbb{R}$, given by the restriction of some $g : \mathbb{R}^d \to \mathbb{R}$, whose norm

$$\|f\|^2_{H^s(D)} := \inf_{g|_D = f} \int \frac{|\widehat{g}(\xi)|^2}{(1 + \|\xi\|^2)^{s/2}}\, d\xi$$

is finite. Thus, for the kernel K with Fourier transform $\widehat{K}(\xi) = (1 + \|\xi\|^2)^{s/2}$, this is just the RKHS $\mathcal{H}(D)$. More generally, if K satisfies our assumptions with $\nu < \infty$, these spaces are equivalent in the sense of normed spaces: they contain the same functions, and have norms $\|\cdot\|_1, \|\cdot\|_2$ satisfying

$$C\|f\|_1 \leq \|f\|_2 \leq C'\|f\|_1,$$

for constants $0 < C \leq C'$.

Lemma 3. *Let $\mathcal{H}_\theta(S)$ denote the RKHS of K_θ on S, and $D \subseteq \mathbb{R}^d$ be a Lipschitz domain.*

(i) If $\nu < \infty$, $\mathcal{H}_\theta(D)$ is equivalent to the Sobolev Hilbert space $H^{\nu + d/2}(D)$.

(ii) If $\nu = \infty$, $\mathcal{H}_\theta(D)$ is continuously embedded in $H^s(D)$ for all s.

Thus if $\nu < \infty$, and X is, say, a product of intervals $\prod_{i=1}^d [a_i, b_i]$, the RKHS $\mathcal{H}_\theta(X)$ is equivalent to the Sobolev Hilbert space $H^{\nu+d/2}(\prod_{i=1}^d (a_i, b_i))$, identifying each function in that space with its unique continuous extension to X.

3.2 Fixed Parameters

We are now ready to state our main results. Let $X \subset \mathbb{R}^d$ be compact with non-empty interior. For a function $f : X \to \mathbb{R}$, let $\mathbb{P}_f^u$ and $\mathbb{E}_f^u$ denote probability and expectation when minimizing the fixed function f with strategy u. (Note that while f is fixed, u may be random, so its performance is still probabilistic in nature.) We define the loss suffered over the ball B_R in $\mathcal{H}_\theta(X)$ after n steps by a strategy u,

$$L_n(u, \mathcal{H}_\theta(X), R) := \sup_{\|f\|_{\mathcal{H}_\theta(X)} \leq R} \mathbb{E}_f^u[f(x_n^*) - \min f].$$

We will say that u converges on the optimum at rate r_n, if

$$L_n(u, \mathcal{H}_\theta(X), R) = O(r_n)$$

for all $R > 0$. Note that we do not allow u to vary with R; the strategy must achieve this rate without prior knowledge of $\|f\|_{\mathcal{H}_\theta(X)}$.

We begin by showing that the minimax rate of convergence is $n^{-\nu/d}$.

Theorem 1. *If $\nu < \infty$, then for any $\theta \in \mathbb{R}_+^d$, $R > 0$,*

$$\inf_u L_n(u, \mathcal{H}_\theta(X), R) = \Theta(n^{-\nu/d}),$$

and this rate can be achieved by a strategy u not depending on R.

The upper bound is provided by a naive strategy as in the introduction: we fix a quasi-uniform sequence x_n in advance, and take x_n^* to minimize a radial basis function interpolant of the data. As remarked previously, however, this naive strategy is not very satisfying; in practice it will be outperformed by any good strategy varying with the data. We may thus ask whether more sophisticated strategies, with better practical performance, can still provide good worst-case bounds.

One such strategy is the $EI(\pi)$ strategy of Definition 1. We can show this strategy converges at least at rate $n^{-(\nu \wedge 1)/d}$, up to log factors.

Theorem 2. *Let π be a prior with length-scales $\theta \in \mathbb{R}_+^d$. For any $R > 0$,*

$$L_n(EI(\pi), \mathcal{H}_\theta(X), R) = \begin{cases} O(n^{-\nu/d}(\log n)^\alpha), & \nu \leq 1, \\ O(n^{-1/d}), & \nu > 1. \end{cases}$$

For $\nu \leq 1$, these rates are near-optimal. For $\nu > 1$, we are faced with a more difficult problem; we discuss this in more detail in Section 3.4.

3.3 Estimated Parameters

First, we consider the effect of the prior parameters on $EI(\pi)$. While the previous result gives a convergence rate for any fixed choice of parameters, the constant in that rate will depend on the parameters chosen; to choose well, we must somehow estimate these parameters from the data. The $EI(\hat{\pi})$ strategy, given by Definition 2, uses maximum likelihood estimates for this purpose. We can show, however, that this may cause the strategy to never converge.

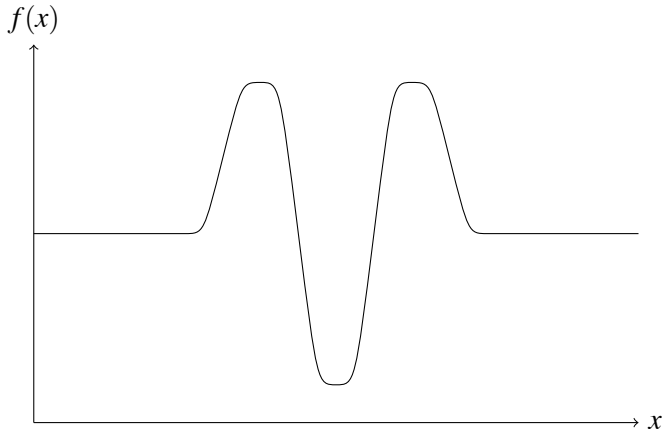


Figure 1: A counterexample from Theorem 3

Theorem 3. *Suppose* $\nu < \infty$. *Given* $\theta \in \mathbb{R}_+^d$, $R > 0$, $\varepsilon > 0$, *there exists* $f \in \mathcal{H}_\theta(X)$ *satisfying* $\|f\|_{\mathcal{H}_\theta(X)} \leq R$, *and for some fixed* $\delta > 0$,

$$\mathbb{P}_f^{EI(\hat{\pi})}\left(\inf_n f(x_n^*) - \min f \geq \delta\right) > 1 - \varepsilon.$$

The counterexamples constructed in the proof of the theorem may be difficult to minimize, but they are not badly-behaved (Figure 1). A good optimization strategy should be able to minimize such functions, and we must ask why expected improvement fails.

We can understand the issue by considering the constant in Theorem 2. Define

$$\tau(x) \coloneqq x\Phi(x) + \varphi(x).$$

From the proof of Theorem 2, the dominant term in the convergence rate has constant

$$C(R+\sigma)\frac{\tau(R/\sigma)}{\tau(-R/\sigma)}, \tag{10}$$

for $C > 0$ not depending on R or σ. In Appendix A, we will prove the following result.

Corollary 1. $\hat{R}_n(\theta)$ *is non-decreasing in* n, *and bounded above by* $\|f\|_{\mathcal{H}_\theta(X)}$.

Hence for fixed θ, the estimate $\hat{\sigma}_n^2 = \hat{R}_n^2(\theta)/n \leq R^2/n$, and thus $R/\hat{\sigma}_n \geq n^{1/2}$. Inserting this choice into (10) gives a constant growing exponentially in n, destroying our convergence rate.

To resolve the issue, we will instead try to pick σ to minimize (10). The term $R+\sigma$ is increasing in σ, and the term $\tau(R/\sigma)/\tau(-R/\sigma)$ is decreasing in σ; we may balance the terms by taking $\sigma = R$. The constant is then proportional to R, which we may minimize by taking $R = \|f\|_{\mathcal{H}_\theta(X)}$. In practice, we will not know $\|f\|_{\mathcal{H}_\theta(X)}$ in advance, so we must estimate it from the data; from Corollary 1, a convenient estimate is $\hat{R}_n(\theta)$.

Suppose, then, that we make some bounded estimate $\hat{\theta}_n$ of θ, and set $\hat{\sigma}_n^2 = \hat{R}_n^2(\hat{\theta}_n)$. As Theorem 3 holds for any $\hat{\sigma}_n^2$ of faster than logarithmic decay, such a choice is necessary to ensure convergence.

(We may also choose θ to minimize (10); we might then pick $\hat{\theta}_n$ minimizing $\hat{R}_n(\theta)\prod_{i=1}^d \theta_i^{-\nu/d}$, but our assumptions on $\hat{\theta}_n$ are weak enough that we need not consider this further.)

If we believe our Gaussian-process model, this estimate $\hat{\sigma}_n$ is certainly unusual. We should, however, take care before placing too much faith in the model. The function in Figure 1 is a reasonable function to optimize, but as a Gaussian process it is highly atypical: there are intervals on which the function is constant, an event which in our model occurs with probability zero. If we want our algorithm to succeed on more general classes of functions, we will need to choose our parameter estimates appropriately.

To obtain good rates, we must add a further condition to our strategy. If $z_1 = \cdots = z_n$, $EI_n(\cdot;\hat{\pi}_n)$ is identically zero, and all choices of x_{n+1} are equally valid. To ensure we fully explore f, we will therefore require that when our strategy is applied to a constant function $f(x) = c$, it produces a sequence x_n dense in X. (This can be achieved, for example, by choosing x_{n+1} uniformly at random from X when $z_1 = \cdots = z_n$.) We have thus described the following strategy.

Definition 3. *An $EI(\tilde{\pi})$ strategy satisfies Definition 2, except:*

(i) we instead set $\hat{\sigma}_n^2 = \hat{R}_n^2(\hat{\theta}_n)$; and

(ii) we require the choice of x_{n+1} maximizing (8) *to be such that, if f is constant, the design points are almost surely dense in X.*

We cannot now prove a convergence result uniform over balls in $\mathcal{H}_\theta(X)$, as the rate of convergence depends on the ratio $R/\hat{R}_n$, which is unbounded. (Indeed, any estimator of $\|f\|_{\mathcal{H}_\theta(X)}$ must sometimes perform poorly: f can appear from the data to have arbitrarily small norm, while in fact having a spike somewhere we have not yet observed.) We can, however, provide the same convergence rates as in Theorem 2, in a slightly weaker sense.

Theorem 4. *For any $f \in \mathcal{H}_{\theta^U}(X)$, under $\mathbb{P}_f^{EI(\tilde{\pi})}$,*

$$f(x_n^*) - \min f = \begin{cases} O_p(n^{-\nu/d}(\log n)^\alpha), & \nu \le 1, \\ O_p(n^{-1/d}), & \nu > 1. \end{cases}$$

3.4 Near-Optimal Rates

So far, our rates have been near-optimal only for $\nu \le 1$. To obtain good rates for $\nu > 1$, standard results on the performance of Gaussian-process interpolation (Narcowich et al., 2003, §6) then require the design points x_i to be quasi-uniform in a region of interest. It is unclear whether this occurs naturally under expected improvement, but there are many ways we can modify the algorithm to ensure it.

Perhaps the simplest, and most well-known, is an ε-greedy strategy (Sutton and Barto, 1998, §2.2). In such a strategy, at each step with probability $1-\varepsilon$ we make a decision to maximize some greedy criterion; with probability ε we make a decision completely at random. This random choice ensures that the short-term nature of the greedy criterion does not overshadow our long-term goal.

The parameter ε controls the trade-off between global and local search: a good choice of ε will be small enough to not interfere with the expected-improvement algorithm, but large enough to prevent it from getting stuck in a local minimum. Sutton and Barto (1998, §2.2) consider the values $\varepsilon = 0.1$ and $\varepsilon = 0.01$, but in practical work ε should of course be calibrated to a typical problem set.

We therefore define the following strategies.

Definition 4. *Let $\cdot$ denote π, $\hat{\pi}$ or $\tilde{\pi}$. For $0 < \varepsilon < 1$, an $EI(\cdot, \varepsilon)$ strategy:*

(i) chooses initial design points $x_1, \cdots, x_k$ independently of f;

(ii) with probability $1 - \varepsilon$, chooses design point x_{n+1} $(n \geq k)$ as in $EI(\cdot)$; or

(iii) with probability ε, chooses x_{n+1} $(n \geq k)$ uniformly at random from X.

We can show that these strategies achieve near-optimal rates of convergence for all $\nu < \infty$.

Theorem 5. *Let $EI(\cdot, \varepsilon)$ be one of the strategies in Definition 4. If $\nu < \infty$, then for any $R > 0$,*

$$L_n(EI(\cdot, \varepsilon), \mathcal{H}_{\theta^U}(X), R) = O((n/\log n)^{-\nu/d}(\log n)^{\alpha}),$$

while if $\nu = \infty$, the statement holds for all $\nu < \infty$.

Note that unlike a typical ε-greedy algorithm, we do not rely on random choice to obtain global convergence: as above, the $EI(\pi)$ and $EI(\tilde{\pi})$ strategies are already globally convergent. Instead, we use random choice simply to improve upon the worst-case rate. Note also that the result does not in general hold when $\varepsilon = 1$; to obtain good rates, we must combine global search with inference about f.

4. Conclusions

We have shown that expected improvement can converge near-optimally, but a naive implementation may not converge at all. We thus echo Diaconis and Freedman (1986) in stating that, for infinite-dimensional problems, Bayesian methods are not always guaranteed to find the right answer; such guarantees can only be provided by considering the problem at hand.

We might ask, however, if our framework can also be improved. Our upper bounds on convergence were established using naive algorithms, which in practice would prove inefficient. If a sophisticated algorithm fails where a naive one succeeds, then the sophisticated algorithm is certainly at fault; we might, however, prefer methods of evaluation which do not consider naive algorithms so successful.

Vazquez and Bect (2010) and Grunewalder et al. (2010) consider a more Bayesian formulation of the problem, where the unknown function f is distributed according to the prior π, but this approach can prove restrictive: as we saw in Section 3.3, placing too much faith in the prior may exclude functions of interest. Further, Grunewalder et al. find the same issues are present also within the Bayesian framework.

A more interesting approach is given by the continuum-armed-bandit problem (Srinivas et al., 2010, and references therein). Here the goal is to minimize the cumulative regret,

$$R_n := \sum_{i=1}^{n} (f(x_i) - \min f),$$

in general observing the function f under noise. Algorithms controlling the cumulative regret at rate r_n also solve the optimization problem, at rate r_n/n (Bubeck et al., 2009, §3). The naive algorithms above, however, have poor cumulative regret. We might, then, consider the cumulative regret to be a better measure of performance, but this approach too has limitations. Firstly, the cumulative regret

is necessarily increasing, so cannot establish rates of optimization faster than n^{-1}. (This is not an issue under noise, where typically $r_n = \Omega(n^{1/2})$, see Kleinberg and Slivkins, 2010.) Secondly, if our goal is optimization, then minimizing the regret, a cost we do not incur, may obscure the problem at hand.

Bubeck et al. (2010) study this problem with the additional assumption that f has finitely many minima, and is, say, quadratic in a neighbourhood of each. This assumption may suffice in practice, and allows the authors to obtain impressive rates of convergence. For optimization, however, a further weakness is that these rates hold only once the algorithm has found a basin of attraction; they thus measure local, rather than global, performance. It may be that convergence rates alone are not sufficient to capture the performance of a global optimization algorithm, and the time taken to find a basin of attraction is more relevant. In any case, the choice of an appropriate framework to measure performance in global optimization merits further study.

Finally, we should also ask how to choose the smoothness parameter ν (or the equivalent parameter in similar algorithms). Van der Vaart and van Zanten (2009) show that Bayesian Gaussian-process models can, in some contexts, automatically adapt to the smoothness of an unknown function f. Their technique requires, however, that the estimated length-scales $\hat{\theta}_n$ to tend to 0, posing both practical and theoretical challenges. The question of how best to optimize functions of unknown smoothness remains open.

Acknowledgments

We would like to thank the referees, as well as Richard Nickl and Steffen Grunewalder, for their valuable comments and suggestions.

Appendix A. Proofs

We now prove the results in Section 3.

A.1 Reproducing-Kernel Hilbert Spaces

Proof of Lemma 1. Let V be the space of functions described, and W be the closed real subspace of Hermitian functions in $L^2(\mathbb{R}^d, \widehat{K}^{-1})$. We will show $f \mapsto \widehat{f}$ is an isomorphism $V \to W$, so we may equivalently work with W. Given $\widehat{f} \in W$, by Cauchy-Schwarz and Bochner's theorem,

$$\int |\widehat{f}| \leq \left(\int \widehat{K} \right)^{1/2} \left(\int |\widehat{f}|^2 / \widehat{K} \right)^{1/2} < \infty,$$

and as $\|\widehat{K}\|_\infty \leq \|K\|_1$,

$$\int |\widehat{f}|^2 \leq \|\widehat{K}\|_\infty \int |\widehat{f}|^2 / \widehat{K} < \infty,$$

so $\widehat{f} \in L^1 \cap L^2$. $\widehat{f}$ is thus the Fourier transform of a real continuous $f \in L^2$, satisfying the Fourier inversion formula everywhere.

$f \mapsto \widehat{f}$ is hence an isomorphism $V \to W$. It remains to show that $V = \mathcal{H}(\mathbb{R}^d)$. W is complete, so V is. Further, $\mathcal{E}(\mathbb{R}^d) \subset V$, and by Fourier inversion each $f \in V$ satisfies the reproducing property,

$$f(x) = \int e^{2\pi i \langle x,\xi\rangle} \widehat{f}(\xi)\, d\xi = \int \frac{\widehat{f}(\xi)\overline{\widehat{k}_x(\xi)}}{\widehat{K}(\xi)}\, d\xi = \langle f, k_x\rangle,$$

so $\mathcal{H}(\mathbb{R}^d)$ is a closed subspace of V. Given $f \in \mathcal{H}(\mathbb{R}^d)^\perp$, $f(x) = \langle f, k_x\rangle = 0$ for all x, so $f = 0$. Thus $V = \mathcal{H}(\mathbb{R}^d)$. □

Proof of Lemma 3. By Lemma 1, the norm on $\mathcal{H}_\theta(\mathbb{R}^d)$ is

$$\|f\|^2_{\mathcal{H}_\theta(\mathbb{R}^d)} = \int \frac{|\widehat{f}(\xi)|^2}{\widehat{K}_\theta(\xi)}\, d\xi,$$

and K_θ has Fourier transform

$$\widehat{K}_\theta(\xi) = \frac{\widehat{K}(\xi_1/\theta_1, \ldots, \xi_d/\theta_d)}{\prod_{i=1}^d \theta_i}.$$

If $\nu < \infty$, by assumption $\widehat{K}(\xi) = \widehat{k}(\|\xi\|)$, for a finite non-increasing function $\widehat{k}$ satisfying $\widehat{k}(\|\xi\|) = \Theta(\|\xi\|^{-2\nu-d})$ as $\xi \to \infty$. Hence

$$C(1+\|\xi\|^2)^{-(\nu+d/2)} \leq \widehat{K}_\theta(\xi) \leq C'(1+\|\xi\|^2)^{-(\nu+d/2)},$$

for constants $C, C' > 0$, and we obtain that $\mathcal{H}_\theta(\mathbb{R}^d)$ is equivalent to the Sobolev space $H^{\nu+d/2}(\mathbb{R}^d)$.

From Lemma 2, $\mathcal{H}_\theta(D)$ is given by the restriction of functions in $\mathcal{H}_\theta(\mathbb{R}^d)$; as D is Lipschitz, the same is true of $H^{\nu+d/2}$. $\mathcal{H}_\theta(D)$ is thus equivalent to $H^{\nu+d/2}(D)$. Finally, functions in $\mathcal{H}_\theta(D)$ are continuous, so uniquely identified by their restriction to D, and

$$\mathcal{H}_\theta(D) \simeq \mathcal{H}_\theta(D) \simeq H^{\nu+d/2}(D).$$

If $\nu = \infty$, by a similar argument $\mathcal{H}_\theta(D)$ is continuously embedded in all $H^s(D)$. □

From Lemma 1, we can derive results on the behaviour of $\|f\|_{\mathcal{H}_\theta(S)}$ as θ varies. For small θ, we obtain the following result.

Lemma 4. *If $f \in \mathcal{H}_\theta(S)$, then $f \in \mathcal{H}_{\theta'}(S)$ for all $0 < \theta' \leq \theta$, and*

$$\|f\|^2_{\mathcal{H}_{\theta'}(S)} \leq \left(\prod_{i=1}^d \theta_i/\theta'_i\right) \|f\|^2_{\mathcal{H}_\theta(S)}.$$

Proof. Let $C = \prod_{i=1}^d (\theta'_i/\theta_i)$. As $\widehat{K}$ is isotropic and radially non-increasing,

$$\widehat{K}_{\theta'}(\xi) = C\widehat{K}_\theta((\theta'_1/\theta_1)\xi_1, \ldots, (\theta'_d/\theta_d)\xi_d) \geq C\widehat{K}_\theta(\xi).$$

Given $f \in \mathcal{H}_\theta(S)$, let $g \in \mathcal{H}_\theta(\mathbb{R}^d)$ be its minimum norm extension, as in Lemma 2. By Lemma 1,

$$\|f\|^2_{\mathcal{H}_{\theta'}(S)} \leq \|g\|^2_{\mathcal{H}_{\theta'}(\mathbb{R}^d)} = \int \frac{|\widehat{g}|^2}{\widehat{K}_{\theta'}} \leq \int \frac{|\widehat{g}|^2}{C\widehat{K}_\theta} = C^{-1}\|f\|^2_{\mathcal{H}_\theta(S)}.$$ □

Likewise, for large θ, we obtain the following.

Lemma 5. *If $\nu < \infty$, $f \in \mathcal{H}_\theta(S)$, then $f \in \mathcal{H}_{t\theta}(S)$ for $t \geq 1$, and*

$$\|f\|^2_{\mathcal{H}_{t\theta}(S)} \leq C'' t^{2\nu} \|f\|^2_{\mathcal{H}_\theta(S)},$$

for a $C'' > 0$ depending only on K and θ.

Proof. As in the proof of Lemma 3, we have constants $C, C' > 0$ such that

$$C(1+\|\xi\|^2)^{-(\nu+d/2)} \leq \widehat{K}_\theta(\xi) \leq C'(1+\|\xi\|^2)^{-(\nu+d/2)}.$$

Thus for $t \geq 1$,

$$\begin{aligned}\widehat{K}_{t\theta}(\xi) = t^d \widehat{K}_\theta(t\xi) &\geq C t^d (1+t^2\|\xi\|^2)^{-(\nu+d/2)} \\ &\geq C t^{-2\nu}(1+\|\xi\|^2)^{-(\nu+d/2)} \\ &\geq CC'^{-1} t^{-2\nu} \widehat{K}_\theta(\xi),\end{aligned}$$

and we may argue as in the previous lemma. □

We can also describe the posterior distribution of f in terms of $\mathcal{H}_\theta(S)$; as a consequence, we may deduce Corollary 1.

Lemma 6. *Suppose $f(x) = \mu + g(x)$, $g \in \mathcal{H}_\theta(S)$.*

(i) $\hat{f}_n(x;\theta) = \hat{\mu}_n + \hat{g}_n(x)$ solves the optimization problem

$$\text{minimize } \|\hat{g}\|^2_{\mathcal{H}_\theta(S)}, \quad \text{subject to } \hat{\mu} + \hat{g}(x_i) = z_i, \quad 1 \leq i \leq n,$$

with minimum value $\hat{R}_n^2(\theta)$.

(ii) The prediction error satisfies

$$|f(x) - \hat{f}_n(x;\theta)| \leq s_n(x;\theta)\|g\|_{\mathcal{H}_\theta(S)}$$

with equality for some $g \in \mathcal{H}_\theta(S)$.

Proof.

(i) Let $W = \operatorname{span}(k_{x_1}, \ldots, k_{x_n})$, and write $\hat{g} = \hat{g}^\| + \hat{g}^\perp$ for $\hat{g}^\| \in W$, $\hat{g}^\perp \in W^\perp$. $\hat{g}^\perp(x_i) = \langle \hat{g}^\perp, k_{x_i} \rangle = 0$, so $\hat{g}^\perp$ affects the optimization only through $\|\hat{g}\|$. The minimal $\hat{g}$ thus has $\hat{g}^\perp = 0$, so $\hat{g} = \sum_{i=1}^n \lambda_i k_{x_i}$. The problem then becomes

$$\text{minimize } \lambda^T V \lambda, \quad \text{subject to } \hat{\mu} 1 + V\lambda = z.$$

The solution is given by (4) and (5), with value (7).

(ii) By symmetry, the prediction error does not depend on μ, so we may take $\mu = 0$. Then

$$f(x) - \hat{f}_n(x;\theta) = g(x) - (\hat{\mu}_n + \hat{g}_n(x)) = \langle g, e_{n,x} \rangle,$$

for $e_{n,x} = k_x - \sum_{i=1}^n \lambda_i k_{x_i}$, and

$$\lambda = \frac{V^{-1}1}{1^T V^{-1} 1} + \left(I - \frac{V^{-1}1}{1^T V^{-1}1} 1^T \right) V^{-1} v.$$

Now, $\|e_{n,x}\|^2_{\mathcal{H}_\theta(S)} = s_n^2(x;\theta)$, as given by (6); this is a consequence of Loève's isometry, but is easily verified algebraically. The result then follows by Cauchy-Schwarz. □

A.2 Fixed Parameters

Proof of Theorem 1. We first establish the lower bound. Suppose we have $2n$ functions ψ_m with disjoint supports. We will argue that, given n observations, we cannot distinguish between all the ψ_m, and thus cannot accurately pick a minimum x_n^*.

To begin with, assume $X = [0,1]^d$. Let $\psi : \mathbb{R}^d \to [0,1]$ be a C^∞ function, supported inside X and with minimum -1. By Lemma 3, $\psi \in \mathcal{H}_\theta(\mathbb{R}^d)$. Fix $k \in \mathbb{N}$, and set $n = (2k)^d/2$. For vectors $m \in \{0,\dots,2k-1\}^d$, construct functions $\psi_m(x) = C(2k)^{-\nu}\psi(2kx-m)$, where $C > 0$ is to be determined. ψ_m is given by a translation and scaling of ψ, so by Lemmas 1, 2 and 5, for some $C' > 0$,

$$\|\psi_m\|_{\mathcal{H}_\theta(X)} \le \|\psi_m\|_{\mathcal{H}_\theta(\mathbb{R}^d)} = C(2k)^{-\nu}\|\psi\|_{\mathcal{H}_{2k\theta}(\mathbb{R}^d)} \le CC'\|\psi\|_{\mathcal{H}_\theta(\mathbb{R}^d)}.$$

Set $C = R/C'\|\psi\|_{\mathcal{H}_\theta(\mathbb{R}^d)}$, so that $\|\psi_m\|_{\mathcal{H}_\theta(X)} \le R$ for all m and k.

Suppose $f = 0$, and let x_n and x_n^* be chosen by any valid strategy u. Set $\chi = \{x_1,\dots,x_{n-1},x_{n-1}^*\}$, and let A_m be the event that $\psi_m(x) = 0$ for all $x \in \chi$. There are n points in χ, and the $2n$ functions ψ_m have disjoint support, so $\sum_m \mathbb{I}(A_m) \ge n$. Thus

$$\sum_m \mathbb{P}_0^u(A_m) = \mathbb{E}_0^u\left[\sum_m \mathbb{I}(A_m)\right] \ge n,$$

and we have some fixed m, depending only on u, for which $\mathbb{P}_0^u(A_m) \ge \frac{1}{2}$. On the event A_m,

$$\psi_m(x_{n-1}^*) - \min\psi_m = C(2k)^{-\nu},$$

but on that event, u cannot distinguish between 0 and ψ_m before time n, so

$$C^{-1}(2k)^{\nu}\mathbb{E}_{\psi_m}^u[f(x_{n-1}^*) - \min f] \ge \mathbb{P}_{\psi_m}^u(A_m) = \mathbb{P}_0^u(A_m) \ge \tfrac{1}{2}.$$

As the minimax loss is non-increasing in n, for $(2(k-1))^d/2 \le n < (2k)^d/2$ we conclude

$$\begin{aligned}\inf_u L_n(u,\mathcal{H}_\theta(X),R) &\ge \inf_u L_{(2k)^d/2-1}(u,\mathcal{H}_\theta(X),R)\\ &\ge \inf_u \sup_m \mathbb{E}_{\psi_m}^u\left[f\left(x_{(2k)^d/2-1}^*\right) - \min f\right]\\ &\ge \tfrac{1}{2}C(2k)^{-\nu} = \Omega(n^{-\nu/d}).\end{aligned}$$

For general X having non-empty interior, we can find a hypercube $S = x_0 + [0,\varepsilon]^d \subseteq X$, with $\varepsilon > 0$. We may then proceed as above, picking functions ψ_m supported inside S.

For the upper bound, consider a strategy u choosing a fixed sequence x_n, independent of the z_n. Fit a radial basis function interpolant $\hat{f}_n$ to the data, and pick x_n^* to minimize $\hat{f}_n$. Then if x^* minimizes f,

$$\begin{aligned}f(x_n^*) - f(x^*) &\le f(x_n^*) - \hat{f}_n(x_n^*) + \hat{f}_n(x^*) - f(x^*)\\ &\le 2\|\hat{f}_n - f\|_\infty,\end{aligned}$$

so the loss is bounded by the error in $\hat{f}_n$.

From results in Narcowich et al. (2003, §6) and Wendland (2005, §11.5), for suitable radial basis functions the error is uniformly bounded by

$$\sup_{\|f\|_{\mathcal{H}_\theta(X)} \le R} \|\hat{f}_n - f\|_\infty = O(h_n^{-\nu}),$$

where the mesh norm

$$h_n := \sup_{x \in X} \min_{i=1}^{n} \|x - x_i\|.$$

(For $\nu \notin \mathbb{N}$, this result is given by Narcowich et al. for the radial basis function K^ν, which is ν-Hölder at 0 by Abramowitz and Stegun, 1965, §9.6; for $\nu \in \mathbb{N}$, the result is given by Wendland for thin-plate splines.) As X is bounded, we may choose the x_n so that $h_n = O(n^{-1/d})$, giving

$$L_n(u, H_\theta(X), R) = O(n^{-\nu/d}). \qquad \square$$

To prove Theorem 2, we first show that some observations z_n will be well-predicted by past data.

Lemma 7. *Set*

$$\beta := \begin{cases} \alpha, & \nu \leq 1, \\ 0, & \nu > 1. \end{cases}$$

Given $\theta \in \mathbb{R}^d_+$, there is a constant $C' > 0$ depending only on X, K and θ which satisfies the following. For any $k \in \mathbb{N}$, and sequences $x_n \in X$, $\theta_n \geq \theta$, the inequality

$$s_n(x_{n+1}; \theta_n) \geq C' k^{-(\nu \wedge 1)/d} (\log k)^\beta$$

holds for at most k distinct n.

Proof. We first show that the posterior variance s_n^2 is bounded by the distance to the nearest design point. Let π_n denote the prior with variance $\sigma^2 = 1$, and length-scales θ_n. Then for any $i \leq n$, as $\hat{f}_n(x; \theta_n) = \mathbb{E}_{\pi_n}[f(x) \mid \mathcal{F}_n]$,

$$\begin{aligned} s_n^2(x; \theta_n) &= \mathbb{E}_{\pi_n}[(f(x) - \hat{f}_n(x; \theta_n))^2 \mid \mathcal{F}_n] \\ &= \mathbb{E}_{\pi_n}[(f(x) - f(x_i))^2 - (f(x_i) - \hat{f}_n(x; \theta_n))^2 \mid \mathcal{F}_n] \\ &\leq \mathbb{E}_{\pi_n}[(f(x) - f(x_i))^2 \mid \mathcal{F}_n] \\ &= 2(1 - K_{\theta_n}(x - x_i)). \end{aligned}$$

If $\nu \leq \frac{1}{2}$, then by assumption

$$|K(x) - K(0)| = O\left(\|x\|^{2\nu} (-\log \|x\|)^{2\alpha}\right)$$

as $x \to 0$. If $\nu > \frac{1}{2}$, then K is differentiable, so as K is symmetric, $\nabla K(0) = 0$. If further $\nu \leq 1$, then

$$|K(x) - K(0)| = |K(x) - K(0) - x \cdot \nabla K(0)| = O\left(\|x\|^{2\nu} (-\log \|x\|)^{2\alpha}\right).$$

Similarly, if $\nu > 1$, then K is C^2, so

$$|K(x) - K(0)| = |K(x) - K(0) - x \cdot \nabla K(0)| = O(\|x\|^2).$$

We may thus conclude

$$|1 - K(x)| = |K(x) - K(0)| = O\left(\|x\|^{2(\nu \wedge 1)} (-\log \|x\|)^{2\beta}\right),$$

and

$$s_n^2(x;\theta_n) \le C^2\|x-x_i\|^{2(\nu\wedge 1)}(-\log\|x-x_i\|)^{2\beta},$$

for a constant $C>0$ depending only on X, K and θ.

We next show that most design points x_{n+1} are close to a previous x_i. X is bounded, so can be covered by k balls of radius $O(k^{-1/d})$. If x_{n+1} lies in a ball containing some earlier point x_i, $i \le n$, then we may conclude

$$s_n^2(x_{n+1};\theta_n) \le C'^2 k^{-2(\nu\wedge 1)/d}(\log k)^{2\beta},$$

for a constant $C'>0$ depending only on X, K and θ. Hence as there are k balls, at most k points x_{n+1} can satisfy

$$s_n(x_{n+1};\theta_n) \ge C' k^{-(\nu\wedge 1)/d}(\log k)^{\beta}. \qquad \square$$

Next, we provide bounds on the expected improvement when f lies in the RKHS.

Lemma 8. *Let $\|f\|_{\mathcal{H}_\theta(X)} \le R$. For $x \in X$, $n \in \mathbb{N}$, set $I = (f(x_n^*) - f(x))^+$, and $s = s_n(x;\theta)$. Then for*

$$\tau(x) := x\Phi(x) + \phi(x),$$

we have

$$\max\left(I - Rs, \frac{\tau(-R/\sigma)}{\tau(R/\sigma)}I\right) \le EI_n(x;\pi) \le I + (R+\sigma)s.$$

Proof. If $s=0$, then by Lemma 6, $\hat{f}_n(x;\theta) = f(x)$, so $EI_n(x;\pi) = I$, and the result is trivial. Suppose $s>0$, and set $t = (f(x_n^*) - f(x))/s$, $u = (f(x_n^*) - \hat{f}_n(x;\theta))/s$. From (8) and (9),

$$EI_n(x;\pi) = \sigma s \tau(u/\sigma),$$

and by Lemma 6, $|u-t| \le R$. As $\tau'(z) = \Phi(z) \in [0,1]$, τ is non-decreasing, and $\tau(z) \le 1+z$ for $z \ge 0$. Hence

$$EI_n(x;\pi) \le \sigma s \tau\left(\frac{t^+ + R}{\sigma}\right) \le \sigma s\left(\frac{t^+ + R}{\sigma} + 1\right) = I + (R+\sigma)s.$$

If $I=0$, then as EI is the expectation of a non-negative quantity, $EI \ge 0$, and the lower bounds are trivial. Suppose $I>0$. Then as $EI \ge 0$, $\tau(z) \ge 0$ for all z, and $\tau(z) = z + \tau(-z) \ge z$. Thus

$$EI_n(x;\pi) \ge \sigma s \tau\left(\frac{t-R}{\sigma}\right) \ge \sigma s\left(\frac{t-R}{\sigma}\right) = I - Rs.$$

Also, as τ is increasing,

$$EI_n(x;\pi) \ge \sigma\tau\left(\frac{-R}{\sigma}\right)s.$$

Combining these bounds, and eliminating s, we obtain

$$EI_n(x;\pi) \ge \frac{\sigma\tau(-R/\sigma)}{R + \sigma\tau(-R/\sigma)}I = \frac{\tau(-R/\sigma)}{\tau(R/\sigma)}I. \qquad \square$$

We may now prove the theorem. We will use the above bounds to show that there must be times n_k when the expected improvement is low, and thus $f(x_{n_k}^*)$ is close to $\min f$.

Proof of Theorem 2. From Lemma 7 there exists $C > 0$, depending on X, K and θ, such that for any sequence $x_n \in X$ and $k \in \mathbb{N}$, the inequality

$$s_n(x_{n+1};\theta) > Ck^{-(\nu\wedge 1)/d}(\log k)^\beta$$

holds at most k times. Furthermore, $z_n^* - z_{n+1}^* \geq 0$, and for $\|f\|_{\mathcal{H}_\theta(X)} \leq R$,

$$\sum_n z_n^* - z_{n+1}^* \leq z_1^* - \min f \leq 2\|f\|_\infty \leq 2R,$$

so $z_n^* - z_{n+1}^* > 2Rk^{-1}$ at most k times. Since $z_n^* - f(x_{n+1}) \leq z_n^* - z_{n+1}^*$, we have also $z_n^* - f(x_{n+1}) > 2Rk^{-1}$ at most k times. Thus there is a time n_k, $k \leq n_k \leq 3k$, for which $s_{n_k}(x_{n_k+1};\theta) \leq Ck^{-(\nu\wedge 1)/d}(\log k)^\beta$ and $z_{n_k}^* - f(x_{n_k+1}) \leq 2Rk^{-1}$.

Let f have minimum z^* at x^*. For k large, x_{n_k+1} will have been chosen by expected improvement (rather than being an initial design point, chosen at random). Then as z_n^* is non-increasing in n, for $3k \leq n < 3(k+1)$ we have by Lemma 8,

$$\begin{aligned}
z_n^* - z^* &\leq z_{n_k}^* - z^* \\
&\leq \frac{\tau(R/\sigma)}{\tau(-R/\sigma)} EI_{n_k}(x^*;\pi) \\
&\leq \frac{\tau(R/\sigma)}{\tau(-R/\sigma)} EI_{n_k}(x_{n_k+1};\pi) \\
&\leq \frac{\tau(R/\sigma)}{\tau(-R/\sigma)} \left(2Rk^{-1} + C(R+\sigma)k^{-(\nu\wedge 1)/d}(\log k)^\beta\right).
\end{aligned}$$

This bound is uniform in f with $\|f\|_{\mathcal{H}_\theta(X)} \leq R$, so we obtain

$$L_n(EI(\pi), \mathcal{H}_\theta(X), R) = O(n^{-(\nu\wedge 1)/d}(\log n)^\beta). \qquad \square$$

A.3 Estimated Parameters

To prove Theorem 3, we first establish lower bounds on the posterior variance.

Lemma 9. *Given $\theta^L, \theta^U \in \mathbb{R}_+^d$, pick sequences $x_n \in X$, $\theta^L \leq \theta_n \leq \theta^U$. Then for open $S \subset X$,*

$$\sup_{x\in S} s_n(x;\theta_n) = \Omega(n^{-\nu/d}),$$

uniformly in the sequences x_n, θ_n.

Proof. S is open, so contains a hypercube T. For $k \in \mathbb{N}$, let $n = \frac{1}{2}(2k)^d$, and construct $2n$ functions ψ_m on T with $\|\psi_m\|_{\mathcal{H}_{\theta^U}(X)} \leq 1$, as in the proof of Theorem 1. Let $C^2 = \prod_{i=1}^d (\theta_i^U/\theta_i^L)$; then by Lemma 4, $\|\psi_m\|_{\mathcal{H}_{\theta_n}(X)} \leq C$.

Given n design points $x_1, \ldots, x_n$, there must be some ψ_m such that $\psi_m(x_i) = 0$, $1 \leq i \leq n$. By Lemma 6, the posterior mean of ψ_m given these observations is the zero function. Thus for $x \in T$ minimizing ψ_m,

$$s_n(x;\theta_n) \geq C^{-1} s_n(x;\theta_n)\|\psi_m\|_{\mathcal{H}_{\theta_n}(X)} \geq C^{-1}|\psi_m(x) - 0| = \Omega(k^{-\nu}).$$

As $s_n(x;\theta)$ is non-increasing in n, for $\frac{1}{2}(2(k-1))^d < n \leq \frac{1}{2}(2k)^d$ we obtain

$$\sup_{x\in S} s_n(x;\theta_n) \geq \sup_{x\in S} s_{\frac{1}{2}(2k)^d}(x;\theta_n) = \Omega(k^{-\nu}) = \Omega(n^{-\nu/d}). \qquad \square$$

Next, we bound the expected improvement when prior parameters are estimated by maximum likelihood.

Lemma 10. *Let $\|f\|_{\mathcal{H}_{\theta^U}(X)} \leq R$, $x_n, y_n \in X$. Set $I_n(x) = z_n^* - f(x)$, $s_n(x) = s_n(x;\hat{\theta}_n)$, and $t_n(x) = I_n(x)/s_n(x)$. Suppose:*

(i) for some $i < j$, $z_i \neq z_j$;

(ii) for some $T_n \to -\infty$, $t_n(x_{n+1}) \leq T_n$ whenever $s_n(x_{n+1}) > 0$;

(iii) $I_n(y_{n+1}) \geq 0$; and

(iv) for some $C > 0$, $s_n(y_{n+1}) \geq e^{-C/c_n}$.

Then for $\hat{\pi}_n$ as in Definition 2, eventually $EI_n(x_{n+1};\hat{\pi}_n) < EI_n(y_{n+1};\hat{\pi}_n)$. If the conditions hold on a subsequence, so does the conclusion.

Proof. Let $\hat{R}_n^2(\theta)$ be given by (7), and set $\hat{R}_n^2 = \hat{R}_n^2(\hat{\theta}_n)$. For $n \geq j$, $\hat{R}_n^2 > 0$, and by Lemma 4 and Corollary 1,

$$\hat{R}_n^2 \leq \|f\|^2_{\mathcal{H}_{\hat{\theta}_n}(X)} \leq S^2 = R^2 \prod_{i=1}^{d} (\theta_i^U/\theta_i^L).$$

Thus $0 < \hat{\sigma}_n^2 \leq S^2 c_n$. Then if $s_n(x) > 0$, for some $|u_n(x) - t_n(x)| \leq S$,

$$EI_n(x;\hat{\pi}_n) = \hat{\sigma}_n s_n(x) \tau(u_n(x)/\hat{\sigma}_n),$$

as in the proof of Lemma 8.

If $s_n(x_{n+1}) = 0$, then $x_{n+1} \in \{x_1, \ldots, x_n\}$, so

$$EI_n(x_{n+1};\hat{\pi}_n) = 0 < EI_n(y_{n+1};\hat{\pi}_n).$$

When $s_n(x_{n+1}) > 0$, as τ is increasing we may upper bound $EI_n(x_{n+1};\hat{\pi}_n)$ using $u_n(x_{n+1}) \leq T_n + S$, and lower bound $EI_n(y_{n+1};\hat{\pi}_n)$ using $u_n(y_{n+1}) \geq -S$. Since $s_n(x_{n+1}) \leq 1$, and $\tau(x) = \Theta(x^{-2}e^{-x^2/2})$ as $x \to -\infty$ (Abramowitz and Stegun, 1965, §7.1),

$$\begin{aligned}
\frac{EI_n(x_{n+1};\hat{\pi}_n)}{EI_n(y_{n+1};\hat{\pi}_n)} &\leq \frac{\tau((T_n+S)/\hat{\sigma}_n)}{e^{-C/c_n}\tau(-S/\hat{\sigma}_n)} \\
&= O\left((T_n+S)^{-2} e^{C/c_n - (T_n^2+2ST_n)/2\hat{\sigma}_n^2}\right) \\
&= O\left((T_n+S)^{-2} e^{-(T_n^2+2ST_n-2CS^2)/2S^2c_n}\right) \\
&= o(1).
\end{aligned}$$

If the conditions hold on a subsequence, we may similarly argue along that subsequence. □

Finally, we will require the following technical lemma.

Lemma 11. *Let $x_1, \ldots, x_n$ be random variables taking values in $\mathbb{R}^d$. Given open $S \subseteq \mathbb{R}^d$, there exist open $U \subseteq S$ for which $\mathbb{P}(\bigcup_{i=1}^n \{x_i \in U\})$ is arbitrarily small.*

Proof. Given $\varepsilon > 0$, fix $m \geq n/\varepsilon$, and pick disjoint open sets $U_1, \ldots, U_m \subset S$. Then

$$\sum_{j=1}^{m} \mathbb{E}[\#\{x_i \in U_j\}] \leq \mathbb{E}[\#\{x_i \in \mathbb{R}^d\}] = n,$$

so there exists U_j with

$$\mathbb{P}\left(\bigcup_i \{x_i \in U_j\}\right) \leq \mathbb{E}[\#\{x_i \in U_j\}] \leq n/m \leq \varepsilon. \qquad \square$$

We may now prove the theorem. We will construct a function f on which the $EI(\hat{\pi})$ strategy never observes within a region W. We may then construct a function g, agreeing with f except on W, but having different minimum. As the strategy cannot distinguish between f and g, it cannot successfully find the minimum of both.

Proof of Theorem 3. Let the $EI(\hat{\pi})$ strategy choose initial design points $x_1, \ldots, x_k$, independently of f. Given $\varepsilon > 0$, by Lemma 11 there exists open $U_0 \subseteq X$ for which $\mathbb{P}^{EI(\hat{\pi})}(x_1, \ldots, x_k \in U_0) \leq \varepsilon$; we may choose U_0 so that $V_0 = X \setminus U_0$ has non-empty interior. Pick open U_1 such that $V_1 = U_1 \subset U_0$, and set f to be a C^∞ function, 0 on V_0, 1 on V_1, and everywhere non-negative. By Lemma 1, $f \in \mathcal{H}_{\theta^U}(X)$.

We work conditional on the event A, having probability at least $1-\varepsilon$, that $z_k^* = 0$, and thus $z_n^* = 0$ for all $n \geq k$. Suppose $x_n \in V_1$ infinitely often, so the z_n are not all equal. By Lemma 7, $s_n(x_{n+1}; \hat{\theta}_n) \to 0$, so on a subsequence with $x_{n+1} \in V_1$, we have

$$t_n = (z_n^* - f(x_{n+1}))/s_n(x_{n+1}; \hat{\theta}_n) = -s_n(x_{n+1}; \hat{\theta}_n)^{-1} \to -\infty$$

whenever $s_n(x_{n+1}; \hat{\theta}_n) > 0$. However, by Lemma 9, there are points $y_n \in V_0$ with $z_n^* - f(y_{n+1}) = 0$, and $s_n(y_{n+1}; \hat{\theta}_n) = \Omega(n^{-\nu/d})$. Hence by Lemma 10, $EI_n(x_{n+1}; \hat{\pi}_n) < EI_n(y_{n+1}; \hat{\pi}_n)$ for some n, contradicting the definition of x_{n+1}.

Hence, on A, there is a random variable T taking values in $\mathbb{N}$, for which $n > T \implies x_n \notin V_1$. Hence there exists a constant $t \in \mathbb{N}$ for which the event $B = A \cap \{T \leq t\}$ has $\mathbb{P}_f^{EI(\hat{\pi})}$-probability at least $1 - 2\varepsilon$. By Lemma 11, we thus have an open set $W \subset V_1$ for which the event

$$C = B \cap \{x_n \notin W : n \in \mathbb{N}\} = B \cap \{x_n \notin W : n \leq t\}$$

has $\mathbb{P}_f^{EI(\hat{\pi})}$-probability at least $1 - 3\varepsilon$.

Construct a smooth function g by adding to f a C^∞ function which is 0 outside W, and has minimum -2. Then $\min g = -1$, but on the event C, $EI(\hat{\pi})$ cannot distinguish between f and g, and $g(x_n^*) \geq 0$. Thus for $\delta = 1$,

$$\mathbb{P}_g^{EI(\hat{\pi})}\left(\inf_n g(x_n^*) - \min g \geq \delta\right) \geq \mathbb{P}_g^{EI(\hat{\pi})}(C) = \mathbb{P}_f^{EI(\hat{\pi})}(C) \geq 1 - 3\varepsilon.$$

As the behaviour of $EI(\hat{\pi})$ is invariant under rescaling, we may scale g to have norm $\|g\|_{\mathcal{H}_\theta(X)} \leq R$, and the above remains true for some $\delta > 0$. $\square$

Proof of Theorem 4. As in the proof of Theorem 2, we will show there are times n_k when the expected improvement is small, so $f(x_{n_k})$ must be close to the minimum. First, however, we must control the estimated parameters $\hat{\sigma}_n^2, \hat{\theta}_n$.

If the z_n are all equal, then by assumption the x_n are dense in X, so f is constant, and the result is trivial. Suppose the z_n are not all equal, and let T be a random variable satisfying $z_T \neq z_i$ for some $i < T$. Set $U = \inf_{\theta^L \leq \theta \leq \theta^U} \hat{R}_T(\theta)$. $\hat{R}_T(\theta)$ is a continuous positive function, so $U > 0$. Let $S^2 = R^2 \prod_{i=1}^d (\theta_i^U / \theta_i^L)$. By Lemma 4, $\|f\|_{\mathcal{H}_{\hat{\theta}_n}(X)} \leq S$, so by Corollary 1, for $n \geq T$,

$$U \leq \hat{R}_T(\hat{\theta}_n) \leq \hat{\sigma}_n \leq \|f\|_{\mathcal{H}_{\hat{\theta}_n}(X)} \leq S.$$

As in the proof of Theorem 2, we have a constant $C > 0$, and some n_k, $k \leq n_k \leq 3k$, for which $z^*_{n_k} - f(x_{n_k+1}) \leq 2Rk^{-1}$ and $s_{n_k}(x_{n_k+1}; \hat{\theta}_{n_k}) \leq Ck^{-\alpha}(\log k)^\beta$. Then for $k \geq T$, $3k \leq n < 3(k+1)$, arguing as in Theorem 2 we obtain

$$\begin{aligned} z^*_n - z^* &\leq z^*_{n_k} - z^* \\ &\leq \frac{\tau(S/\hat{\sigma}_{n_k})}{\tau(-S/\hat{\sigma}_{n_k})} \left(2Rk^{-1} + C(S + \hat{\sigma}_{n_k}) k^{-(\nu \wedge 1)/d} (\log k)^\beta\right) \\ &\leq \frac{\tau(S/U)}{\tau(-S/U)} \left(2Rk^{-1} + 2CSk^{-(\nu \wedge 1)/d} (\log k)^\beta\right). \end{aligned}$$

We thus have a random variable C' satisfying $z^*_n - z^* \leq C' n^{-(\nu \wedge 1)/d} (\log n)^\beta$ for all n, and the result follows. □

A.4 Near-Optimal Rates

To prove Theorem 5, we first show that the points chosen at random will be quasi-uniform in X.

Lemma 12. *Let x_n be i.i.d. random variables, distributed uniformly over X, and define their mesh norm,*

$$h_n := \sup_{x \in X} \min_{i=1}^n \|x - x_i\|.$$

For any $\gamma > 0$, there exists $C > 0$ such that

$$\mathbb{P}(h_n > C(n/\log n)^{-1/d}) = O(n^{-\gamma}).$$

Proof. We will partition X into n regions of size $O(n^{-1/d})$, and show that with high probability we will place an x_i in each one. Then every point x will be close to an x_i, and the mesh norm will be small.

Suppose $X = [0,1]^d$, fix $k \in \mathbb{N}$, and divide X into $n = k^d$ sub-cubes $X_m = \frac{1}{k}(m + [0,1]^d)$, for $m \in \{0, \ldots, k-1\}^d$. Let I_m be the indicator function of the event

$$\{x_i \notin X_m : 1 \leq i \leq \lfloor \gamma n \log n \rfloor\},$$

and define

$$\mu_n = \mathbb{E}\left[\sum_m I_m\right] = n\mathbb{E}[I_0] = n(1 - 1/n)^{\lfloor \gamma n \log n \rfloor} \sim ne^{-\gamma \log n} = n^{-(\gamma - 1)}.$$

For n large, $\mu_n \leq 1$, so by the generalized Chernoff bound of Panconesi and Srinivasan (1997, §3.1),

$$\mathbb{P}\left(\sum_m I_m \geq 1\right) \leq \left(\frac{e^{(\mu_n^{-1} - 1)}}{\mu_n^{-\mu_n^{-1}}}\right)^{\mu_n} \leq e\mu_n \sim en^{-(\gamma - 1)}.$$

On the event $\sum_m I_m < 1$, $I_m = 0$ for all m. For any $x \in X$, we then have $x \in X_m$ for some m, and $x_j \in X_m$ for some $1 \le j \le \lfloor \gamma n \log n \rfloor$. Thus

$$\min_{i=1}^{\lfloor \gamma n \log n \rfloor} \|x - x_i\| \le \|x - x_j\| \le \sqrt{d} k^{-1}.$$

As this bound is uniform in x, we obtain $h_{\lfloor \gamma n \log n \rfloor} \le \sqrt{d} k^{-1}$. Thus for $n = k^d$,

$$\mathbb{P}(h_{\lfloor \gamma n \log n \rfloor} > \sqrt{d} k^{-1}) = O(k^{-d(\gamma-1)}),$$

and as h_n is non-increasing in n, this bound holds also for $k^d \le n < (k+1)^d$. By a change of variables, we then obtain

$$\mathbb{P}(h_n > C(n/\gamma \log n)^{-1/d}) = O((n/\gamma \log n)^{-(\gamma-1)}),$$

and the result follows by choosing γ large. For general X, as X is bounded it can be partitioned into n regions of measure $\Theta(n^{-1/d})$, so we may argue similarly. □

We may now prove the theorem. We will show that the points x_n must be quasi-uniform in X, so posterior variances must be small. Then, as in the proofs of Theorems 2 and 4, we have times when the expected improvement is small, so $f(x_n^*)$ is close to $\min f$.

Proof of Theorem 5. First suppose $\nu < \infty$. Let the $EI(\cdot, \varepsilon)$ choose k initial design points independent of f, and suppose $n \ge 2k$. Let A_n be the event that $\lfloor \frac{\varepsilon}{4} n \rfloor$ of the points $x_{k+1}, \ldots, x_n$ are chosen uniformly at random, so by a Chernoff bound,

$$\mathbb{P}^{EI(\cdot,\varepsilon)}(A_n^c) \le e^{-\varepsilon n/16}.$$

Let B_n be the event that one of the points $x_{n+1}, \ldots, x_{2n}$ is chosen by expected improvement, so

$$\mathbb{P}^{EI(\cdot,\varepsilon)}(B_n^c) = \varepsilon^n.$$

Finally, let C_n be the event that A_n and B_n occur, and further the mesh norm $h_n \le C(n/\log n)^{-1/d}$, for the constant C from Lemma 12. Set $r_n = (n/\log n)^{-\nu/d} (\log n)^{\alpha}$. Then by Lemma 12, since $C_n \subset A_n$,

$$\mathbb{P}_f^{EI(\cdot,\varepsilon)}(C_n^c) \le C' r_n,$$

for a constant $C' > 0$ not depending on f.

Let $EI(\cdot, \varepsilon)$ have prior π_n at time n, with (fixed or estimated) parameters σ_n, θ_n. Suppose $\|f\|_{\mathcal{H}_{\theta^U}(X)} \le R$, and set $S^2 = R^2 \prod_{i=1}^d (\theta_i^U / \theta_i^L)$, so by Lemma 4, $\|f\|_{\mathcal{H}_{\theta_n}(X)} \le S$. If $\alpha = 0$, then by Narcowich et al. (2003, §6),

$$\sup_{x \in X} s_n(x; \theta) = O(M(\theta) h_n^{\nu})$$

uniformly in θ, for $M(\theta)$ a continuous function of θ. Hence on the event C_n,

$$\sup_{x \in X} s_n(x; \theta_n) \le \sup_{x \in X} \sup_{\theta^L \le \theta \le \theta^U} s_n(x; \theta) \le C'' r_n,$$

for a constant $C'' > 0$ depending only on X, K, C, θ^L and θ^U. If $\alpha > 0$, the same result holds by a similar argument.

On the event C_n, we have some x_m chosen by expected improvement, $n < m \leq 2n$. Let f have minimum z^* at x^*. Then by Lemma 8,

$$\begin{aligned} z^*_{m-1} - z^* &\leq EI_{m-1}(x^*; \cdot) + C'' S r_{m-1} \\ &\leq EI_{m-1}(x_m; \cdot) + C'' S r_{m-1} \\ &\leq (f(x_{m-1}) - f(x_m))^+ + C''(2S + \sigma_{m-1}) r_{m-1} \\ &\leq z^*_{m-1} - z^*_m + C'' T r_n, \end{aligned}$$

for a constant $T > 0$. (Under $EI(\pi, \varepsilon)$, we have $T = 2S + \sigma$; otherwise $\sigma_{m-1} \leq S$ by Corollary 1, so $T = 3S$.) Thus, rearranging,

$$z^*_{2n} - z^* \leq z^*_m - z^* \leq C'' T r_n.$$

On the event C_n^c, we have $z^*_{2n} - z^* \leq 2\|f\|_\infty \leq 2R$, so

$$\begin{aligned} \mathbb{E}_f^{EI(\cdot,\varepsilon)}[z^*_{2n+1} - z^*] &\leq \mathbb{E}_f^{EI(\cdot,\varepsilon)}[z^*_{2n} - z^*] \\ &\leq 2R \mathbb{P}_f^{EI(\cdot,\varepsilon)}(C_n^c) + C'' T r_n \\ &\leq (2C'R + C'' T) r_n. \end{aligned}$$

As this bound is uniform in f with $\|f\|_{\mathcal{H}_{\theta^U}(X)} \leq R$, the result follows. If instead $\nu = \infty$, the above argument holds for any $\nu < \infty$. □

References

Milton Abramowitz and Irene A. Stegun, editors. *Handbook of Mathematical Functions*. Dover, New York, 1965.

Robert J. Adler and Jonathan E. Taylor. *Random Fields and Geometry*. Springer Monographs in Mathematics. Springer, New York, 2007.

Nachman Aronszajn. Theory of reproducing kernels. *Trans. Amer. Math. Soc.*, 68(3):337–404, 1950.

Alain Berlinet and Christine Thomas-Agnan. *Reproducing Kernel Hilbert Spaces in Probability and Statistics*. Kluwer Academic Publishers, Boston, Massachusetts, 2004.

Eric Brochu, Mike Cora, and Nando de Freitas. A tutorial on Bayesian optimization of expensive cost functions, with application to active user modeling and hierarchical reinforcement learning. Arxiv preprint arXiv:1012.2599, 2010.

Sébastien Bubeck, Rémi Munos, and Gilles Stoltz. Pure exploration in multi-armed bandits problems. In *Proc. 20th International Conference on Algorithmic Learning Theory (ALT '09)*, pages 23–37, Porto, Portugal, 2009.

Sébastien Bubeck, Rémi Munos, Gilles Stoltz, and Csaba Szepesvari. X-armed bandits. Arxiv preprint arXiv:1001.4475, 2010.

Persi Diaconis and David Freedman. On the consistency of Bayes estimates. *Ann. Statist.*, 14(1): 1–26, 1986.

Peter Frazier, Warren Powell, and Savas Dayanik. The knowledge-gradient policy for correlated normal beliefs. *INFORMS J. Comput.*, 21(4):599–613, 2009.

David Ginsbourger, Rodolphe le Riche, and Laurent Carraro. A multi-points criterion for deterministic parallel global optimization based on Gaussian processes. HAL preprint hal-00260579, 2008.

Steffen Grunewalder, Jean-Yves Audibert, Manfred Opper, and John Shawe-Taylor. Regret bounds for Gaussian process bandit problems. In *Proc. 13th International Conference on Artificial Intelligence and Statistics (AISTATS '10)*, pages 273–280, Sardinia, Italy, 2010.

Pierre Hansen, Brigitte Jaumard, and Shi-Hui Lu. Global optimization of univariate Lipschitz functions: I. survey and properties. *Math. Program.*, 55(1):251–272, 1992.

Donald R. Jones, Cary D. Perttunen, and Bruce E. Stuckman. Lipschitzian optimization without the Lipschitz constant. *J. Optim. Theory Appl.*, 79(1):157–181, 1993.

Donald R. Jones, Matthias Schonlau, and William J. Welch. Efficient global optimization of expensive black-box functions. *J. Global Optim.*, 13(4):455–492, 1998.

Robert Kleinberg and Aleksandrs Slivkins. Sharp dichotomies for regret minimization in metric spaces. In *Proc. ACM-SIAM Symposium on Discrete Algorithms (SODA '10)*, pages 827–846, Austin, Texas, 2010.

Marco Locatelli. Bayesian algorithms for one-dimensional global optimization. *J. Global Optim.*, 10(1):57–76, 1997.

William G. Macready and David H. Wolpert. Bandit problems and the exploration / exploitation tradeoff. *IEEE Trans. Evol. Comput.*, 20(1):2–22, 1998.

Jonas Močkus. On Bayesian methods for seeking the extremum. In *Proc. IFIP Technical Conference*, pages 400–404, Novosibirsk, Russia, 1974.

Francis J. Narcowich, Joseph D. Ward, and Holger Wendland. Refined error estimates for radial basis function interpolation. *Constr. Approx.*, 19(4):541–564, 2003.

Michael Osborne. *Bayesian Gaussian processes for sequential prediction, optimisation and quadrature*. DPhil thesis, University of Oxford, Oxford, UK, 2010.

Alessandro Panconesi and Aravind Srinivasan. Randomized distributed edge coloring via an extension of the Chernoff-Hoeffding bounds. *SIAM J. Comput.*, 26(2):350–368, 1997.

Panos M. Pardalos and H. Edwin Romeijn, editors. *Handbook of Global Optimization, Volume 2*. Nonconvex Optimization and its Applications. Kluwer Academic Publishers, Dordrecht, the Netherlands, 2002.

Emanuel Parzen. Probability density functionals and reproducing kernel Hilbert spaces. In *Proc. Symposium on Time Series Analysis*, pages 155–169, Providence, Rhode Island, 1963.

Carl E. Rasmussen and Christopher K. I. Williams. *Gaussian Processes for Machine Learning*. MIT Press, Cambridge, Massachusetts, 2006.

Thomas J. Santner, Brian J. Williams, and William I. Notz. *The Design and Analysis of Computer Experiments*. Springer Series in Statistics. Springer, New York, 2003.

Niranjan Srinivas, Andreas Krause, Sham M. Kakade, and Matthias Seeger. Gaussian process optimization in the bandit setting: no regret and experimental design. In *Proc. 27th International Conference on Machine Learning (ICML '10)*, Haifa, Israel, 2010.

Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: an Introduction*. MIT Press, Cambridge, Massachusetts, 1998.

Luc Tartar. *An Introduction to Sobolev Spaces and Interpolation Spaces*, volume 3 of *Lecture Notes of the Unione Matematica Italiana*. Springer, New York, 2007.

Aad W. van der Vaart and J. Harry van Zanten. Reproducing kernel Hilbert spaces of Gaussian priors. In *Pushing the Limits of Contemporary Statistics: Contributions in Honor of Jayanta K. Ghosh*, volume 3 of *Institute of Mathematical Statistics Collections*, pages 200–222. Institute of Mathematical Statistics, Beachwood, Ohio, 2008.

Aad W. van der Vaart and J. Harry van Zanten. Adaptive Bayesian estimation using a Gaussian random field with inverse gamma bandwidth. *Ann. Statist.*, 37(5B):2655–2675, 2009.

Emmanuel Vazquez and Julien Bect. Convergence properties of the expected improvement algorithm with fixed mean and covariance functions. *J. Statist. Plann. Inference*, 140(11):3088–3095, 2010.

Holger Wendland. *Scattered Data Approximation*. Cambridge Monographs on Applied and Computational Mathematics. Cambridge University Press, Cambridge, UK, 2005.

Journal of Machine Learning Research 12 (2011) 2905-2929 Submitted 1/11; Revised 8/11; Published 10/11

On Equivalence Relationships Between Classification and Ranking Algorithms

Şeyda Ertekin SEYDA@MIT.EDU
Cynthia Rudin RUDIN@MIT.EDU
MIT Sloan School of Management
Massachusetts Institute of Technology
Cambridge, MA 02142, USA

Editor: Yoav Freund

Abstract

We demonstrate that there are machine learning algorithms that can achieve success for two separate tasks simultaneously, namely the tasks of classification and bipartite ranking. This means that advantages gained from solving one task can be carried over to the other task, such as the ability to obtain conditional density estimates, and an order-of-magnitude reduction in computational time for training the algorithm. It also means that some algorithms are robust to the choice of evaluation metric used; they can theoretically perform well when performance is measured either by a misclassification error or by a statistic of the ROC curve (such as the area under the curve). Specifically, we provide such an equivalence relationship between a generalization of Freund et al.'s RankBoost algorithm, called the "P-Norm Push," and a particular cost-sensitive classification algorithm that generalizes AdaBoost, which we call "P-Classification." We discuss and validate the potential benefits of this equivalence relationship, and perform controlled experiments to understand P-Classification's empirical performance. There is no established equivalence relationship for logistic regression and its ranking counterpart, so we introduce a logistic-regression-style algorithm that aims in between classification and ranking, and has promising experimental performance with respect to both tasks.

Keywords: supervised classification, bipartite ranking, area under the curve, rank statistics, boosting, logistic regression

1. Introduction

The success of a machine learning algorithm can be judged in many different ways. Thus, algorithms that are somehow robust to multiple performance metrics may be more generally useful for a wide variety of problems. Experimental evaluations of machine learning algorithms tend to reflect this by considering several different measures of success (see, for example, the study of Caruana and Niculescu-Mizil, 2006). For instance, classification algorithms are commonly judged both by their classification accuracy and the Area Under the ROC curve (AUC), even though these algorithms are designed only to optimize the classification accuracy, and not the AUC.

If algorithms should be judged using multiple measures of success, it makes sense to analyze and design algorithms that achieve success with respect to multiple performance metrics. This is the topic considered in this work, and we show that several additional advantages, such as probabilistic interpretability and computational speed, can be gained from finding equivalence relationships between algorithms for different problems. It is true that there is no "free lunch" (Wolpert and

Macready, 1997), we are not claiming that one algorithm can solve all problems. We instead point out that it is possible to optimize two objectives simultaneously if they have an overlapping set of optima. The objectives in this case are convexified versions of the performance metrics for classification and ranking. In this work, we show that a particular equivalence relationship exists between two algorithms: a ranking algorithm called the P-Norm Push (Rudin, 2009), which is a generalization of RankBoost (Freund et al., 2003) that aims to maximize a weighted area under the curve, and a classification algorithm that we call P-Classification, which is a generalization of AdaBoost (Freund and Schapire, 1997) that aims to minimize a weighted misclassification error. Specifically, we show that P-Classification not only optimizes its objective, but also optimizes the objective of the P-Norm Push (and vice versa, the P-Norm Push can be made to optimize P-Classification's objective function). Thus, P-Classification and the P-Norm Push perform equally well on both of their objectives; P-Classification can be used as a ranking algorithm, and the P-Norm Push can be made into a classification algorithm. This equivalence relationship allows us to: 1) obtain conditional density estimates for $P(y=1|x)$ for the P-Norm Push (and thus RankBoost as a special case), 2) obtain solutions of the P-Norm Push an order of magnitude faster without sacrificing the quality of the solution at all, and 3) show a relationship between the P-Norm Push's objective and the "precision" metric used in Information Retrieval. This relationship will allow us to conduct a set of controlled experiments to better understand P-Classification's empirical performance. It is not clear that such an equivalence relationship holds between logistic regression and its ranking counterpart; in fact, our experiments indicate that no such relationship can hold.

Bipartite ranking problems are similar to but distinct from binary classification problems. In both bipartite ranking and classification problems, the learner is given a training set of examples $\{(x_1,y_1),\ldots,(x_m,y_m)\}$ consisting of instances $x \in \mathcal{X}$ that are either positive ($y=1$) or negative ($y=-1$). The goal of bipartite ranking is to learn a real-valued ranking function $f : \mathcal{X} \to \mathbb{R}$ that ranks future positive instances higher than negative ones. Bipartite ranking algorithms optimize rank statistics, such as the AUC. There is no decision boundary, and the absolute scores of the examples do not matter, instead the values of the scores relative to each other are important. Classification algorithms optimize a misclassification error, and are they are not designed to optimize rank statistics. The "equivalence" is where a classification (or ranking) algorithm aims to optimize not only a misclassification error, but also a rank statistic.

The first work that suggested such equivalence relationships could hold is that of Rudin and Schapire (2009), showing that AdaBoost is equivalent to RankBoost. They showed that AdaBoost, which iteratively minimizes the exponential misclassification loss, also iteratively minimizes RankBoost's exponential ranking loss, and vice versa, that RankBoost with trivial modifications can be made to minimize the exponential misclassification loss. The first result of our work, provided in Section 3, is to broaden that proof to handle more general ranking losses and classification losses. The more general ranking loss is that of the P-Norm Push, which concentrates on "pushing" negatives away from the top of the ranked list. The more general classification loss, determined mainly by the number of false positives, is minimized by P-Classification, which is introduced formally in Section 3. Also in Section 3, we consider another simple cost-sensitive version of AdaBoost, and show the forward direction of its equivalence to RankBoost; in this case, the cost-sensitive version of AdaBoost minimizes RankBoost's objective no matter what the cost parameter is. In Section 4, we will verify the equivalence relationship empirically and provide evidence that no such relationship holds for logistic regression and its ranking counterpart. In Section 5 we discuss the first two main benefits gained by this equivalence relationship described above, namely obtaining probability esti-

mates, and computing solutions faster. In Section 6 we discuss the relationship of P-Classification's objective to the "precision" metric, and evaluate several parameters influencing the performance of P-Classification. As a result, we are able to suggest improvements to boost performance. Section 7 introduces a new logistic-regression-style algorithm that solves a problem in between classification and ranking, in the hopes of performing better than AdaBoost on both counts. Note that this work does not relate directly to work on reductions (e.g., Balcan et al., 2008), since in this work, the same set of features are used for both the classification and ranking problems without any modification. Another recent work that addresses the use of classification algorithms for solving ranking problems is that of Kotlowski et al. (2011), who show loss and regret bounds on the ranking performance of classifiers. Their analysis is useful when equivalence relationships, such as the ones we show here, do not hold.

2. Definitions

We denote the set of instances with positive labels as $\{x_i\}_{i=1,\ldots,I}$, and the set of instances with negative labels as $\{\tilde{x}_k\}_{i=1,\ldots,K}$, where $x_i, \tilde{x}_k \in \mathcal{X}$. Throughout most of the paper, the subscripts i and k will be used as indices over positive and negative instances, respectively. We assume that (x_i, y_i) are drawn from a joint distribution D on $\mathcal{X} \times \{-1, 1\}$. Our goal is to construct a scoring function $f : \mathcal{X} \to \mathbb{R}$ which gives a real valued score to each instance in $\mathcal{X}$. Let $\mathcal{F}$ denote the hypothesis space that is the class of convex combinations of features $\{h_j\}_{j=1\ldots n}$, where $h_j : \mathcal{X} \to \{-1, 1\}$. The function $f \in \mathcal{F}$ is then defined as a linear combination of the features:

$$f := f_{\boldsymbol{\lambda}} := \sum_j \lambda_j h_j,$$

where $\boldsymbol{\lambda} \in \mathbb{R}^n$ will be chosen to minimize (or approximately minimize) an objective function. We always include a y-intercept (feature that is 1 for all x), assigned to the index j. This y-intercept term is important in the equivalence proof; in order to turn the P-Norm Push into a classification algorithm, we need to place a decision boundary by adjusting the y-intercept.

In bipartite ranking, the goal is to rank positive instances higher than the negative ones. The quality of a ranking function is often measured by the area under the ROC curve (AUC). The associated misranking loss, related to $1 - \text{AUC}$, is the number of positive instances that are ranked below negative instances:

$$\text{standard misranking loss } (f) = \sum_{i=1}^{I} \sum_{k=1}^{K} \mathbb{1}_{[f(x_i) \leq f(\tilde{x}_k)]}. \tag{1}$$

The ranking loss is zero when all negative instances are ranked below the positives instances.

In binary classification, the goal is to correctly predict the true labels of positive and negative examples. The loss is measured by the misclassification error:

$$\text{standard misclassification loss } (f) = \sum_{i=1}^{I} \mathbb{1}_{[f(x_i) \leq 0]} + \sum_{k=1}^{K} \mathbb{1}_{[f(\tilde{x}_k) \geq 0]}. \tag{2}$$

Since it is difficult to minimize (1) and (2) directly, a widely used approach is to minimize a convex upper bound on the loss. The exponential loss that is iteratively minimized by AdaBoost (Freund

and Schapire, 1997) is one such example:

$$\mathcal{R}^{AB}(\boldsymbol{\lambda}) := \sum_{i=1}^{I} e^{-f_{\boldsymbol{\lambda}}(x_i)} + \sum_{k=1}^{K} e^{f_{\boldsymbol{\lambda}}(\tilde{x}_k)} =: \mathcal{R}_{+}^{AB}(\boldsymbol{\lambda}) + \mathcal{R}_{-}^{AB}(\boldsymbol{\lambda}). \tag{3}$$

The ranking counterpart of AdaBoost is RankBoost (Freund et al., 2003). RankBoost's objective function is a sum of exponentiated differences in scores, a convexified version of (1):

$$\mathcal{R}^{RB}(\boldsymbol{\lambda}) := \sum_{i=1}^{I}\sum_{k=1}^{K} e^{-(f_{\boldsymbol{\lambda}}(x_i)-f_{\boldsymbol{\lambda}}(\tilde{x}_k))} = \sum_{i=1}^{I} e^{-f_{\boldsymbol{\lambda}}(x_i)} \sum_{k=1}^{K} e^{f_{\boldsymbol{\lambda}}(\tilde{x}_k)} = \mathcal{R}_{+}^{AB}(\boldsymbol{\lambda})\mathcal{R}_{-}^{AB}(\boldsymbol{\lambda}). \tag{4}$$

A generalization of RankBoost that is considered in this paper is the P-Norm Push (Rudin, 2009), which minimizes the following objective:

$$\mathcal{R}^{PN}(\boldsymbol{\lambda}) := \sum_{k=1}^{K} \left(\sum_{i=1}^{I} e^{-(f_{\boldsymbol{\lambda}}(x_i)-f_{\boldsymbol{\lambda}}(\tilde{x}_k))} \right)^{p}.$$

By increasing p, one changes how hard the algorithm concentrates on "pushing" high scoring negative examples down from the top of the list. The power p acts as a soft maximum for the highest scoring negative instance. When $p = 1$, P-Norm Push's objective reduces to RankBoost's objective.

We also investigate the equivalence relationship between logistic regression and its ranking counterpart, which we call "Logistic Regression Ranking" (LRR). Logistic regression minimizes the objective:

$$\mathcal{R}^{LR}(\boldsymbol{\lambda}) := \sum_{i=1}^{I} \log\left(1 + e^{-f_{\boldsymbol{\lambda}}(x_i)}\right) + \sum_{k=1}^{K} \log\left(1 + e^{f_{\boldsymbol{\lambda}}(\tilde{x}_k)}\right), \tag{5}$$

whereas LRR is defined with the following objective:

$$\mathcal{R}^{LRR}(\boldsymbol{\lambda}) := \sum_{i=1}^{I}\sum_{k=1}^{K} \log\left(1 + e^{-(f_{\boldsymbol{\lambda}}(x_i)-f_{\boldsymbol{\lambda}}(\tilde{x}_k))}\right). \tag{6}$$

LRR bears a strong resemblance to the algorithm RankNet (Burges et al., 2005) in that its objective (6) is similar to the second term of RankNet's objective (Equation 3 in Burges et al., 2005). LRR's objective (6) is an upper bound on the 0-1 ranking loss in (1), using the logistic loss $\log(1+e^{-z})$ to upper bound the 0-1 loss $\mathbb{1}_{z\leq 0}$.

3. Equivalence Relationships

We now introduce P-Classification, which is a boosting-style algorithm that minimizes a weighted misclassification error. Like the P-Norm Push, it concentrates on "pushing" the negative examples down from the top of the list. Unlike the P-Norm Push, it minimizes a weighted misclassification error (rather than a weighted misranking error), though we will show that minimization of either objective yields an equally good result for either problem. P-Classification minimizes the following loss:

$$\mathcal{R}^{PC}(\boldsymbol{\lambda}) := \sum_{i=1}^{I} e^{-f_{\boldsymbol{\lambda}}(x_i)} + \frac{1}{p}\sum_{k=1}^{K} e^{f_{\boldsymbol{\lambda}}(\tilde{x}_k)p} =: \mathcal{R}_{+}^{PC}(\boldsymbol{\lambda}) + \mathcal{R}_{-}^{PC}(\boldsymbol{\lambda}) \tag{7}$$

where $f_{\boldsymbol{\lambda}} = \sum_j \lambda_j h_j$. P-Classification is a generalization of AdaBoost, in that when $p = 1$ (i.e., no emphasis on the top-scoring negatives), P-Classification's loss reduces exactly to that of AdaBoost's. We implemented P-Classification as coordinate descent (functional gradient descent) on $\mathcal{R}^{PC}(\boldsymbol{\lambda})$. Pseudocode is presented in Figure 1, using the notation of Collins et al. (2002), where i is the index over all examples (not just positive examples), and $\mathbf{M}$ is the "game matrix" for AdaBoost, where $M_{ij} = y_i h_j(x_i)$. AdaBoost was originally shown to be a coordinate descent algorithm by Breiman (1997), Friedman et al. (2000), Rätsch et al. (2001), Duffy and Helmbold (1999) and Mason et al. (2000).

1. **Input:** examples $\{(x_i, y_i)\}_{i=1}^m$, where $(x_i, y_i) \in \mathcal{X} \times \{-1, 1\}$, features $\{h_j\}_{j=1}^n$, $h_j : \mathcal{X} \to \mathbb{R}$, number of iterations $t_{\max}$, parameter p.

2. **Define:** $M_{ij} := y_i h_j(x_i)$ for all i, j,
   $\phi_i := \mathbb{1}_{\{y_i=1\}} + p\mathbb{1}_{\{y_i=-1\}}$ for all i.

3. **Initialize:** $\lambda_{1,j} = 0$ for $j = 1, \ldots, n$, $\quad d_{1,i} = 1/m$ for $i = 1, \ldots, m$.

4. **Loop for** $t = 1, ..., t_{\max}$

   (a) $j_t \in \operatorname{argmax}_j \sum_{i=1}^m d_{t,i} M_{ij}$

   (b) Perform a linesearch for α_t. That is, find a value α_t that minimizes:

   $$\left(\sum_{i=1}^m d_{t,i} M_{i,j_t} e^{-\alpha_t \phi_i M_{i,j_t}} \right)^2$$

   (c) $\lambda_{t+1,j} = \lambda_{t,j} + \alpha_t \mathbb{1}_{j=j_t}$

   (d) $d_{t+1,i} = d_{t,i} e^{-\alpha_t \phi_i M_{i,j_t}}$ for $i = 1, \ldots, m$

   (e) $d_{t+1,i} = \frac{d_{t+1,i}}{\sum_i d_{t+1,i}}$

5. **Output:** $\boldsymbol{\lambda}_{t_{\max}}$

Figure 1: Pseudocode for the P-Classification algorithm.

There are other cost-sensitive boosting algorithms similar to P-Classification. Sun et al. (2007) introduced three "modifications" of AdaBoost's weight update scheme, in order to make it cost sensitive. Modifications I and II incorporate an arbitrary constant (a cost item) for each example somewhere within the update scheme, where the third modification is a blend of Modifications I and II. Since Sun et al. (2007) consider the iteration scheme itself, some of their modifications are not easy to interpret with respect to a global objective, particularly Modification II (and thus III). Although no global objective is provided explicitly, Modification I seems to correspond to approximate minimization of the following objective: $\left[\sum_i \frac{1}{C_i} e^{-(\mathbf{M}\boldsymbol{\lambda}_i) C_i}\right]$. In that sense, P-Classification is almost a special case of Modification I, where in our case, we would take their arbitrary C_i to be assigned the value of ϕ_i. The step size α_t within Modification I is an approximate solution to a linesearch, whereas we use a numerical (exact) linesearch. The AdaCost algorithm of Fan et al. (1999)

is another cost-sensitive variation of AdaBoost, however it is not associated with a global objective and in our experiments (not shown here) it tended to choose the intercept repeatedly as the weak classifier, and thus the combined classifier was also the intercept. Lozano and Abe (2008) also use ℓ_p norms within a boosting-style algorithm, but for the problem of label ranking instead of example ranking. However, their objective is totally different than ours; for instance, it does not correspond directly to a 0-1 misranking loss like (1).

We will now show that the P-Norm Push is equivalent to P-Classification, meaning that minimizers of P-Classification's objective are also minimizers of the P-Norm Push's objective, and that there is a trivial transformation of the P-Norm Push's output that will minimize P-Classification's objective. This trivial transformation simply puts a decision boundary in the right place. This proof will generalize the result of Rudin and Schapire (2009), but with a simpler proof strategy; since there are pathological cases where minimizers of the objectives occur only at infinity, the result of Rudin and Schapire (2009) used a Bregman distance technique to incorporate these points at infinity. Our proof does not handle the cases at infinity, but does handle, in a much simpler way, all other cases. These points at infinity occur because the objectives $\mathcal{R}^{PC}$ and $\mathcal{R}^{PN}$ are not strictly convex (though they are convex). For AdaBoost, there is work showing that the minimization problem can be essentially split into two subproblems, one which handles examples near the decision boundary and is strictly convex, and the other which handles examples that become infinitely far away from the boundary (Mukherjee et al., 2011).

The forward direction of the equivalence relationship is as follows:

Theorem 1 *(P-Classification minimizes P-Norm Push's objective)*
If $\boldsymbol{\lambda}^{PC} \in \operatorname{argmin}_{\boldsymbol{\lambda}} \mathcal{R}^{PC}(\boldsymbol{\lambda})$ *(assuming minimizers exist), then* $\boldsymbol{\lambda}^{PC} \in \operatorname{argmin}_{\boldsymbol{\lambda}} \mathcal{R}^{PN}(\boldsymbol{\lambda})$.

The corresponding proof within Rudin and Schapire (2009) used four main steps, which we follow also in our proof: 1) characterizing the conditions to be at a minimizer of the classification objective function, 2) using those conditions to develop a "skew" condition on the classes, 3) characterizing the conditions to be at a minimizer of the ranking objective function and 4) plugging the skew condition into the equations arising from step 3, and simplifying to show that a minimizer of the classification objective is also a minimizer of the ranking objective.

Proof Step 1 is to characterize the conditions to be at a minimizer of the classification loss. Define $\boldsymbol{\lambda}^{PC}$ to be a minimizer of $\mathcal{R}^{PC}$ (assuming minimizers exist). At $\boldsymbol{\lambda}^{PC}$ we have: for all $j = 1, \ldots, n$:

$$\begin{aligned} 0 = \frac{\partial \mathcal{R}^{PC}(\boldsymbol{\lambda})}{\partial \lambda_j}\Big|_{\boldsymbol{\lambda}=\boldsymbol{\lambda}^{PC}} &= \sum_{k=1}^{K} e^{p\sum_j \lambda_j^{PC} h_j(\tilde{x}_k)} h_j(\tilde{x}_k) + \sum_{i=1}^{I} e^{-\sum_j \lambda_j^{PC} h_j(x_i)}(-h_j(x_i)) \\ &= \sum_{k=1}^{K} v_k^p h_j(\tilde{x}_k) + \sum_{i=1}^{I} q_i(-h_j(x_i)) \end{aligned} \tag{8}$$

where $v_k := e^{f_{\boldsymbol{\lambda}^{PC}}(\tilde{x}_k)}$ and $q_i := e^{-f_{\boldsymbol{\lambda}^{PC}}(x_i)}$.

Step 2 is to develop a skew condition on the classes. When j is j then $h_j(x_i) = 1$ for all i, and $h_j(\tilde{x}_k) = 1$ for all k. Using this, we can derive the skew condition:

$$0 = \sum_{k=1}^{K} v_k^p - \sum_{i=1}^{I} q_i := p\mathcal{R}_{-}^{PC}(\boldsymbol{\lambda}^{PC}) - \mathcal{R}_{+}^{PC}(\boldsymbol{\lambda}^{PC}). \qquad \text{(skew condition)} \tag{9}$$

Step 3 is to characterize the conditions to be at a minimizer of the ranking loss. First we simplify the derivatives:

$$\begin{aligned}
\frac{\partial \mathcal{R}^{PN}(\boldsymbol{\lambda})}{\partial \lambda_j} &= \sum_{k=1}^{K} p \left(\sum_{i=1}^{I} e^{(-f_{\boldsymbol{\lambda}}(x_i) - f_{\boldsymbol{\lambda}}(\tilde{x}_k))} \right)^{p-1} \left[\sum_{i=1}^{I} e^{-(f_{\boldsymbol{\lambda}}(x_i) - f_{\boldsymbol{\lambda}}(\tilde{x}_k))} \left[-(h_j(x_i) - h_j(\tilde{x}_k)) \right] \right] \\
&= p \sum_{k=1}^{K} e^{f_{\boldsymbol{\lambda}}(\tilde{x}_k) p} \left(\sum_{i=1}^{I} e^{-f_{\boldsymbol{\lambda}}(x_i)} \right)^{p-1} \left(h_j(\tilde{x}_k) \sum_{i=1}^{I} e^{-f_{\boldsymbol{\lambda}}(x_i)} - \sum_{i=1}^{I} h_j(x_i) e^{-f_{\boldsymbol{\lambda}}(x_i)} \right) \\
&= p \left(\sum_{i=1}^{I} e^{-f_{\boldsymbol{\lambda}}(x_i)} \right)^{p-1} \left[\sum_{k=1}^{K} h_j(\tilde{x}_k) e^{f_{\boldsymbol{\lambda}}(\tilde{x}_k) p} \sum_{i=1}^{I} e^{-f_{\boldsymbol{\lambda}}(x_i)} \right. \\
&\qquad \left. - \sum_{k=1}^{K} e^{f_{\boldsymbol{\lambda}}(\tilde{x}_k) p} \sum_{i=1}^{I} h_j(x_i) e^{-f_{\boldsymbol{\lambda}}(x_i)} \right]. \qquad (10)
\end{aligned}$$

To be at a minimizer, the derivatives above must all be zero. Continuing to step 4, when $\boldsymbol{\lambda} = \boldsymbol{\lambda}^{PC}$, we have:

$$\frac{\partial \mathcal{R}^{PN}(\boldsymbol{\lambda})}{\partial \lambda_j}\Big|_{\boldsymbol{\lambda}=\boldsymbol{\lambda}^{PC}} = p \left(\sum_{i=1}^{I} q_i \right)^{p-1} \left[\sum_{k=1}^{K} h_j(\tilde{x}_k) v_k^p \sum_{i=1}^{I} q_i - \sum_{k=1}^{K} v_k^p \sum_{i=1}^{I} h_j(x_i) q_i \right].$$

Using the skew condition (9), and then (8),

$$\begin{aligned}
\frac{\partial \mathcal{R}^{PN}(\boldsymbol{\lambda})}{\partial \lambda_j}\Big|_{\boldsymbol{\lambda}=\boldsymbol{\lambda}^{PC}} &= p \left(\sum_{i=1}^{I} q_i \right)^{p-1} \left(\sum_{i=1}^{I} q_i \right) \left[\sum_{k=1}^{K} h_j(\tilde{x}_k) v_k^p - \sum_{i=1}^{I} h_j(x_i) q_i \right] \\
&= p \left(\sum_{i=1}^{I} q_i \right)^{p} \frac{\partial \mathcal{R}^{PC}(\boldsymbol{\lambda})}{\partial \lambda_j}\Big|_{\boldsymbol{\lambda}=\boldsymbol{\lambda}^{PC}} = 0. \qquad (11)
\end{aligned}$$

This means that $\boldsymbol{\lambda}^{PC}$ is a minimizer of the P-Norm Push's objective. ■

The backwards direction of the equivalence relationship is as follows:

Theorem 2 *(The P-Norm Push can be trivially altered to minimize P-Classification's objective.) Let j index the constant feature $h_j(x) = 1 \ \ \forall x$. Take $\boldsymbol{\lambda}^{PN} \in \operatorname{argmin}_{\boldsymbol{\lambda}} \mathcal{R}^{PN}(\boldsymbol{\lambda})$ (assuming minimizers exist). Create a "corrected" $\boldsymbol{\lambda}^{PN,\text{corr}}$ as follows:*

$$\boldsymbol{\lambda}^{PN,\text{corr}} = \boldsymbol{\lambda}^{PN} + b \cdot \mathbf{e}_j,$$

where

$$b = \frac{1}{p+1} \ln \frac{\sum_i e^{-f_{\boldsymbol{\lambda}^{PN}}(x_i)}}{\sum_k e^{f_{\boldsymbol{\lambda}^{PN}}(\tilde{x}_k) p}}. \qquad (12)$$

Then, $\boldsymbol{\lambda}^{PN,\text{corr}} \in \operatorname{argmin}_{\boldsymbol{\lambda}} \mathcal{R}^{PC}(\boldsymbol{\lambda})$.

Rudin and Schapire (2009) have a very simple proof for the reverse direction, but this technique could not easily be applied here. We have instead used the following proof outline: first, we

show that the corrected vector $\boldsymbol{\lambda}^{PN,\text{corr}}$ satisfies the skew condition; then by deriving an expression similar to (11), we show that if the corrected P-Norm Push's derivatives are zero, then so are P-Classification's derivatives at the corrected P-Norm Push's solution.

Proof We will first show that the skew condition is satisfied for corrected vector $\boldsymbol{\lambda}^{PN,\text{corr}}$. The condition we need to prove is:

$$\sum_{i=1}^{I} q_i^{\text{corr}} = \sum_{k=1}^{K} (v_k^{\text{corr}})^p \qquad \text{(skew condition)}, \tag{13}$$

where $v_k^{\text{corr}} := e^{f_{\boldsymbol{\lambda}^{PN,\text{corr}}}(\tilde{x}_k)}$ and $q_i^{\text{corr}} := e^{-f_{\boldsymbol{\lambda}^{PN,\text{corr}}}(x_i)}$.

From (12), we have:

$$e^{-b} = \left[\frac{\sum_k e^{f_{\boldsymbol{\lambda}^{PN}}(\tilde{x}_k)p}}{\sum_i e^{-f_{\boldsymbol{\lambda}^{PN}}(x_i)}}\right]^{\frac{1}{p+1}}.$$

The left side of (13) thus reduces as follows:

$$\begin{aligned}
\sum_{i=1}^{I} q_i^{\text{corr}} &= \sum_{i=1}^{I} e^{-f_{\boldsymbol{\lambda}^{PN}}(x_i)-b} = e^{-b}\sum_{i=1}^{I} e^{-f_{\boldsymbol{\lambda}^{PN}}(x_i)} \\
&= \left[\frac{\sum_{k=1}^{K} e^{f_{\boldsymbol{\lambda}^{PN}}(\tilde{x}_k)p}}{\sum_{i=1}^{I} e^{-f_{\boldsymbol{\lambda}^{PN}}(x_i)}}\right]^{\frac{1}{p+1}} \sum_{i=1}^{I} e^{-f_{\boldsymbol{\lambda}^{PN}}(x_i)} \\
&= \left[\sum_{i=1}^{I} e^{-f_{\boldsymbol{\lambda}^{PN}}(x_i)}\right]^{\frac{p}{p+1}} \left[\sum_{k=1}^{K} e^{f_{\boldsymbol{\lambda}^{PN}}(\tilde{x}_k)p}\right]^{\frac{1}{p+1}}.
\end{aligned} \tag{14}$$

Now consider the right side:

$$\begin{aligned}
\sum_{k=1}^{K} (v_k^{\text{corr}})^p &= \sum_{k=1}^{K} e^{f_{\boldsymbol{\lambda}^{PN}}(\tilde{x}_k)p} e^{bp} \\
&= \sum_{k=1}^{K} e^{f_{\boldsymbol{\lambda}^{PN}}(\tilde{x}_k)p} \left[\frac{\sum_{i=1}^{I} e^{-f_{\boldsymbol{\lambda}^{PN}}(x_i)}}{\sum_k e^{f_{\boldsymbol{\lambda}^{PN}}(\tilde{x}_k)p}}\right]^{\frac{p}{p+1}} \\
&= \left[\sum_{i=1}^{I} e^{-f_{\boldsymbol{\lambda}^{PN}}(x_i)}\right]^{\frac{p}{p+1}} \left[\sum_{k=1}^{K} e^{f_{\boldsymbol{\lambda}^{PN}}(\tilde{x}_k)p}\right]^{\frac{1}{p+1}}.
\end{aligned} \tag{15}$$

Expression (14) is equal to expression (15), so the skew condition in (13) holds. According to (10), at $\boldsymbol{\lambda} = \boldsymbol{\lambda}^{PN,\text{corr}}$,

$$\begin{aligned}
\frac{\partial \mathcal{R}^{PN}(\boldsymbol{\lambda})}{\partial \lambda_j}\Big|_{\boldsymbol{\lambda}=\boldsymbol{\lambda}^{PN,\text{corr}}} = p\left(\sum_i q_i^{\text{corr}}\right)^{p-1} \Bigg[&\sum_k h_j(\tilde{x}_k)(v_k^{\text{corr}})^p \sum_i q_i^{\text{corr}} \\
&- \sum_k (v_k^{\text{corr}})^p \sum_i h_j(x_i) q_i^{\text{corr}}\Bigg].
\end{aligned}$$

Incorporating (13),

$$\frac{\partial \mathcal{R}^{PN}(\boldsymbol{\lambda})}{\partial \lambda_j}\Big|_{\boldsymbol{\lambda}=\boldsymbol{\lambda}^{PN,\text{corr}}} = p\left(\sum_i q_i^{\text{corr}}\right)^{p} \left[\sum_k h_j(\tilde{x}_k)(v_k^{\text{corr}})^p - \sum_i h_j(x_i) q_i^{\text{corr}}\right],$$

which includes the derivatives of $\mathcal{R}^{PC}$:

$$\frac{\partial \mathcal{R}^{PN}(\boldsymbol{\lambda})}{\partial \lambda_j}\Big|_{\boldsymbol{\lambda}=\boldsymbol{\lambda}^{PN,\text{corr}}} = p\left(\sum_i q_i^{\text{corr}}\right)^p \left[\frac{\partial \mathcal{R}^{PC}(\boldsymbol{\lambda})}{\partial \lambda_j}\Big|_{\boldsymbol{\lambda}=\boldsymbol{\lambda}^{PN,\text{corr}}}\right].$$

By our assumption that $\boldsymbol{\lambda}^{PN,\text{corr}}$ exists, $\sum_i q_i^{\text{corr}}$ is positive and finite. Thus, whenever $\frac{\partial \mathcal{R}^{PN}(\boldsymbol{\lambda})}{\partial \lambda_j}\Big|_{\boldsymbol{\lambda}=\boldsymbol{\lambda}^{PN,\text{corr}}} = 0$ for all j we have

$$\frac{\partial \mathcal{R}^{PC}(\boldsymbol{\lambda})}{\partial \lambda_j}\Big|_{\boldsymbol{\lambda}=\boldsymbol{\lambda}^{PN,\text{corr}}} = 0.$$

We need only that $\frac{\partial \mathcal{R}^{PN}(\boldsymbol{\lambda})}{\partial \lambda_j}\Big|_{\boldsymbol{\lambda}=\boldsymbol{\lambda}^{PN,\text{corr}}} = 0, \quad \forall j$. This is not difficult to show, since the correction b never influences the value of $\mathcal{R}^{PN}$, that is, $\mathcal{R}^{PN}(\boldsymbol{\lambda}) = \mathcal{R}^{PN}(\boldsymbol{\lambda}^{\text{corr}})$. ■

As an alternative to P-Classification, we consider a simple weighted version of AdaBoost. The objective for this algorithm, which we call "Cost-Sensitive AdaBoost," is a weighted combination of $\mathcal{R}_-^{AB}$ and $\mathcal{R}_+^{AB}$. The objective is:

$$\mathcal{R}^{CSA}(\boldsymbol{\lambda}) := \sum_{i=1}^{I} e^{-f_{\boldsymbol{\lambda}}(x_i)} + C\sum_{k=1}^{K} e^{f_{\boldsymbol{\lambda}}(\tilde{x}_k)} =: \mathcal{R}_+^{AB}(\boldsymbol{\lambda}) + C\mathcal{R}_-^{AB}(\boldsymbol{\lambda}).$$

Cost-Sensitive AdaBoost can be implemented by using AdaBoost's usual update scheme, where the only change from AdaBoost is the initial weight vector: d_0 is set so that the negatives are each weighted C times as much as the positives.

No matter what C is, we prove that Cost-Sensitive AdaBoost minimizes RankBoost's objective. This indicates that Cost-Sensitive AdaBoost is not fundamentally different than AdaBoost itself. To show this, we prove the forward direction of the equivalence relationship between Cost-Sensitive AdaBoost (for any C) and RankBoost. We did not get this type of result earlier for P-Classification, because P-Classification produces different solutions than AdaBoost (and the P-Norm Push produces different solutions than RankBoost). Here is the forward direction of the equivalence relationship:

Theorem 3 *(Cost-Sensitive AdaBoost minimizes RankBoost's objective.)*
If $\boldsymbol{\lambda}^{CSA} \in \text{argmin}_{\boldsymbol{\lambda}} \mathcal{R}^{CSA}(\boldsymbol{\lambda})$ *(assuming minimizers exist), then* $\boldsymbol{\lambda}^{CSA} \in \text{argmin}_{\boldsymbol{\lambda}} \mathcal{R}^{RB}(\boldsymbol{\lambda})$.

The proof follows the same four steps outlined for the proof of Theorem 1.

Proof Define $\boldsymbol{\lambda}^{CSA}$ to be a minimizer of $\mathcal{R}^{CSA}$ (assuming minimizers exist). At $\boldsymbol{\lambda}^{CSA}$ we have:

$$\begin{aligned} 0 = \frac{\partial \mathcal{R}^{CSA}(\boldsymbol{\lambda})}{\partial \lambda_j}\Big|_{\boldsymbol{\lambda}=\boldsymbol{\lambda}^{CSA}} &= \frac{\partial \mathcal{R}_+^{AB}(\boldsymbol{\lambda})}{\partial \lambda_j} + C\frac{\partial \mathcal{R}_-^{AB}(\boldsymbol{\lambda})}{\partial \lambda_j} \\ &= \sum_{i=1}^{I} q_i(-h_j(x_i)) + C\sum_{k=1}^{K} v_k h_j(\tilde{x}_k) \end{aligned} \tag{16}$$

where $v_k := e^{f_{\boldsymbol{\lambda}^{CSA}}(\tilde{x}_k)}$ and $q_i := e^{-f_{\boldsymbol{\lambda}^{CSA}}(x_i)}$. The next step is to develop a skew condition on the classes. When j is j then $h_j(x_i) = 1$ for all i, and $h_j(\tilde{x}_k) = 1$ for all k. Using this, the skew condition can be derived as follows:

$$0 = C\sum_{k=1}^{K} v_k - \sum_{i=1}^{I} q_i := C\mathcal{R}_{-}^{AB}(\boldsymbol{\lambda}^{CSA}) - \mathcal{R}_{+}^{AB}(\boldsymbol{\lambda}^{CSA}). \qquad \text{(skew condition)}$$

We now characterize the conditions to be at a minimizer of the ranking loss. We plug the skew condition into the derivatives from RankBoost's objective, which is given in (4):

$$\begin{aligned}
\frac{\partial \mathcal{R}^{RB}(\boldsymbol{\lambda})}{\partial \lambda_j} &= \sum_{k=1}^{K} h_j(\tilde{x}_k) v_k \sum_{i=1}^{I} q_i - \sum_{k=1}^{K} v_k \sum_{i=1}^{I} h_j(x_i) q_i \\
&= \sum_{k=1}^{K} h_j(\tilde{x}_k) v_k \left[C \sum_{k=1}^{K} v_k \right] - \left[\sum_{k=1}^{K} v_k \right] \sum_{i=1}^{I} h_j(x_i) q_i \\
&= \left[\sum_{k=1}^{K} v_k \right] \left[C \sum_{k=1}^{K} h_j(\tilde{x}_k) v_k - \sum_{i=1}^{I} h_j(x_i) q_i \right].
\end{aligned}$$

To be at a minimizer, the derivative must be zero. When $\boldsymbol{\lambda} = \boldsymbol{\lambda}^{CSA}$, from (16) we have:

$$\frac{\partial \mathcal{R}^{RB}(\boldsymbol{\lambda})}{\partial \lambda_j}\Big|_{\boldsymbol{\lambda}=\boldsymbol{\lambda}^{CSA}} = \left[\sum_{k=1}^{K} v_k \right] \frac{\partial \mathcal{R}^{CSA}(\boldsymbol{\lambda})}{\partial \lambda_j}\Big|_{\boldsymbol{\lambda}=\boldsymbol{\lambda}^{CSA}} = 0.$$

This means that $\boldsymbol{\lambda}^{CSA}$ is a minimizer of RankBoost's objective, regardless of the value of C. ■

4. Verification of Theoretical Results

The previous section presented theoretical results; in this section and in the following sections we demonstrate that these results can have direct implications for empirical practice. The equivalence of P-Classification and P-Norm Push can be observed easily in experiments, both in the special case $p = 1$ (AdaBoost and RankBoost are equivalent, Section 4.2) as well as for their generalizations when $p > 1$ (in Section 4.3). We further investigated whether a similar equivalence property holds for logistic regression and Logistic Regression Ranking ("LRR," defined in Section 2). We present empirical evidence in Section 4.4 suggesting that such an equivalence relationship does not hold between these two algorithms. Both algorithms have been implemented as coordinate descent on their objectives. Coordinate descent was first suggested for logistic regression by Friedman et al. (2000).

4.1 Data Sets

For the experimental evaluation, we used the Letter Recognition, MAGIC, Yeast and Banana data sets obtained from the UCI repository (Frank and Asuncion, 2010). The Letter Recognition data set consists of various statistics computed from black-and-white rectangular pixel displays, which each represent one of the 26 capital-letters of the English alphabet. The learning task is to determine which letter an image represents. The MAGIC data set contains data from the **M**ajor **A**tmospheric **G**amma **I**maging **C**herenkov Telescope project. The goal is to discriminate the statistical signatures of Monte Carlo simulated "gamma" particles from simulated "hadron" particles. The yeast data set is a collection of protein sequences and the goal is to predict cellular localization sites of each

Data Set	# Training examples	# Test examples	# Features
Letter Recognition	1000	4000	15
MAGIC	1000	4000	11
Yeast	500	984	9
Banana	1000	4000	5

Table 1: Sizes of training/test sets used in the experiments. The number of features column represents the total number of features for the data sets, including the intercept.

protein. Banana is an artificial data set with a banana-shaped distribution of two classes, represented by two features.

For MAGIC, Letter Recognition, and Yeast data sets, the weight of each feature $h_j(x_i)$ was quantized into -1 or +1 based on thresholding on mean$_i h_j(x_i)$. The MAGIC data set was not further pre-processed beyond this, and "hadrons" were used as the positive class. For the Letter Recognition data set, we transformed the data set to two distinct categories, where the letter *A* represents the positive class and the remaining letters collectively form the negative class. This transformation created a highly imbalanced data set and presented a challenge in our experimental setup for the RankBoost algorithm, which, in its original implementation uses an analytical solution for the line search for α_t at each iteration. In particular, the analytical solution requires that the fraction in the expression for α_t (Equation 2 in Freund et al., 2003) is neither zero nor infinity. To ensure this, each feature h_j in the training set must have at least one positive example where $h_j = 1$ and a negative example where $h_j = -1$, and similarly, the training set should also contain at least one positive example where $h_j = -1$ and a negative example where $h_j = 1$. Our random sampling of the training sets for the Letter Recognition data set did not often satisfy the requirement on the positive examples for "*x2bar*" and "*x-ege*" features; we thus removed these two features. Note that we could not use RankBoost in its original form, since our features are $\{-1,1\}$-valued rather than $\{0,1\}$-valued. We simply rederived Equation 2 in Freund et al. (2003) to accommodate this. For the Yeast data set, from the 10 different classes of localization sites, we used CYT (cytosolic or cytoskeletal) as the positive class and combined the remaining 9 classes as the negative class. We used the 8 numerical features of the Yeast data set and omitted the categorical feature. We increased the number of features of the Banana data set by mapping the original two features $\{x_1, x_2\}$ to a new four-dimensional feature space $\{x_1^{'}, x_1^{''}, x_2^{'}, x_2^{''}\}$ by thresholding the original feature values at values of -4 and -2. Namely, we used the mapping:

$$x_i^{'} = \begin{cases} +1 & \text{if } x_i > -2 \\ -1 & \text{otherwise} \end{cases} \quad \text{and} \quad x_i^{''} = \begin{cases} +1 & \text{if } x_i > -4 \\ -1 & \text{otherwise} \end{cases}.$$

for $i = 1,2$. The experimental results reported in this section are averaged over 10 random and distinct train/test splits. The size of train/test splits for each data set and the number of features are presented in Table 1.

4.2 Equivalence of AdaBoost and RankBoost

Although AdaBoost and RankBoost perform (asymptotically) equally well, it is not immediately clear whether this equivalence would be able to be observed if the algorithm is stopped before the

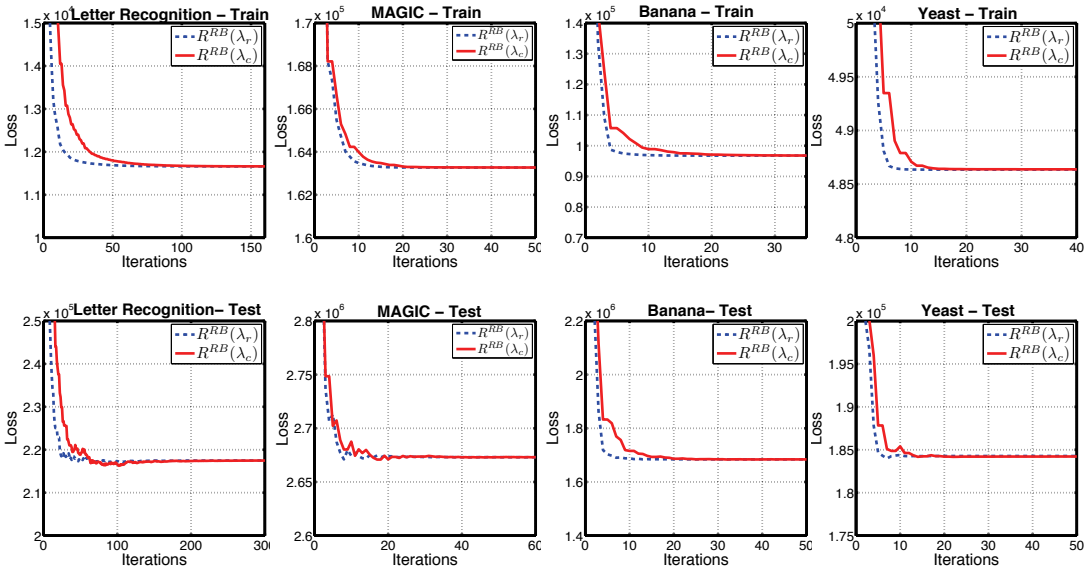


Figure 2: Verifying the forward direction of the equivalence relationship for AdaBoost and RankBoost

regime of convergence is reached. We present empirical evidence to support the forward direction of the theoretical equivalence relationship for $p = 1$, on both the training and test splits, for all data sets described above.

In Figure 2, $\{\boldsymbol{\lambda}_r\}_r$ and $\{\boldsymbol{\lambda}_c\}_c$ denote sequences of coefficients produced by RankBoost and AdaBoost respectively; the subscripts r and c stand for **r**anking and **c**lassification. The figure illustrates both the convergence of AdaBoost and the convergence of RankBoost, with respect to RankBoost's objective R^{RB}. The x-axis denotes the number of iterations. The illustration supports the convergence of AdaBoost to a minimizer of RankBoost's objective. Because of the way that RankBoost is designed, $\mathcal{R}^{RB}(\boldsymbol{\lambda}_r)$ converges more rapidly (in terms of the number of iterations), than $\mathcal{R}^{RB}(\boldsymbol{\lambda}_c)$.

4.3 Equivalence of P-Classification and P-Norm Push

In the same experimental setting, we now validate the equivalence of P-Classification and P-Norm Push. In Figure 3, $\{\boldsymbol{\lambda}_r\}_r$ and $\{\boldsymbol{\lambda}_c\}_c$ denote sequences of coefficients produced by the P-Norm Push and P-Classification respectively. Convergence is illustrated for both algorithms with respect to the P-Norm Push's objective R^{PN}. The x-axis again denotes the number of iterations. The figure illustrates that P-Classification can effectively be used to minimize the P-Norm Push's objective. Comparing with the $p = 1$ results in Figure 2, the convergence behavior on the training sets are similar, whereas there are small differences in convergence behavior on the test sets. One important distinction between training and testing phases is that the ranking loss is required to decrease monotonically on the training set, but not on the test set. As discussed in depth in Rudin (2009), generalization is more difficult as p grows, so we expect a larger difference between training and test behavior for Figure 3.

The next experiment verifies the backwards direction of the equivalence relationship. We demonstrate that a sequence of corrected $\boldsymbol{\lambda}$'s minimizing P-Norm Push's objective also minimizes P-Classification's objective. At each iteration of the P-Norm Push, we compute b as defined in (12) and update $\boldsymbol{\lambda}_r$ accordingly. The sequences of $\mathcal{R}^{PC}$ values for $\{\boldsymbol{\lambda}_c\}_c$ and the corrected $\{\boldsymbol{\lambda}_r\}_r$ are shown in Figure 4.

4.4 Equivalence Does Not Seem To Hold For Logistic Regression and Logistic Regression Ranking

We implemented a coordinate descent algorithm for minimizing LRR's objective function (6), where pseudocode is given in Figure 5. Note that the pseudocode for minimizing logistic regression's objective function would be similar, with the only change being that the definition of the matrix M is the same as in Figure 1.

Figure 6 provides evidence that no such equivalence relationship holds for logistic regression and Logistic Regression Ranking. For this experiment, $\{\boldsymbol{\lambda}_r\}_r$ and $\{\boldsymbol{\lambda}_c\}_c$ denote sequences of coefficients produced by LRR and logistic regression respectively. Convergence is illustrated for both algorithms with respect to LRR's objective. Even after many more iterations than the earlier experiments, and after LRR's objective function values have plateaued for the two algorithms, these values are not close together.

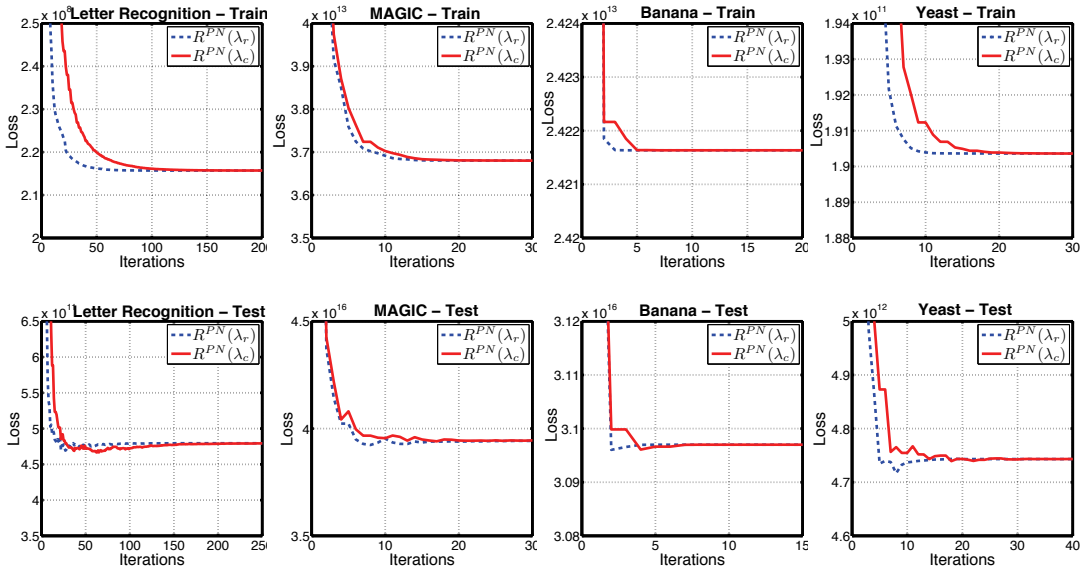


Figure 3: Verifying the forward direction of the equivalence theorem for P-Classification and the P-Norm Push (p=4)

5. Benefits of the Equivalence Relationship

Four major benefits of the equivalence relationship are (i) theoretical motivation, using an algorithm that optimizes classification and ranking objectives simultaneously (ii) gaining the ability to estimate conditional probabilities from a ranking algorithm, (iii) much faster runtimes for ranking tasks, by passing to a classification algorithm and using the equivalence relationship, and (iv) building a relationship between the P-Norm Push and the "precision" performance metric through P-Classification's objective (discussed in Section 6). We have already discussed theoretical motivation, and we will now discuss the other two benefits.

5.1 Estimating Probabilities

The main result in this section is that the scoring function $f_{\boldsymbol{\lambda}}(x)$ can be used to obtain estimates of the conditional probability $P(y=1|x)$. This result relies on properties of the loss functions, including smoothness, and the equivalence relationship of Theorem 2. Note that the conditional probability estimates for AdaBoost are known not to be very accurate asymptotically in many cases (e.g., see Mease et al., 2007), even though AdaBoost generally provides models with high classification and ranking accuracy (e.g., see Caruana and Niculescu-Mizil, 2006). In other words, even in cases where the probability estimates are not accurate, the relative ordering of probability estimates (the ranking) can be accurate.

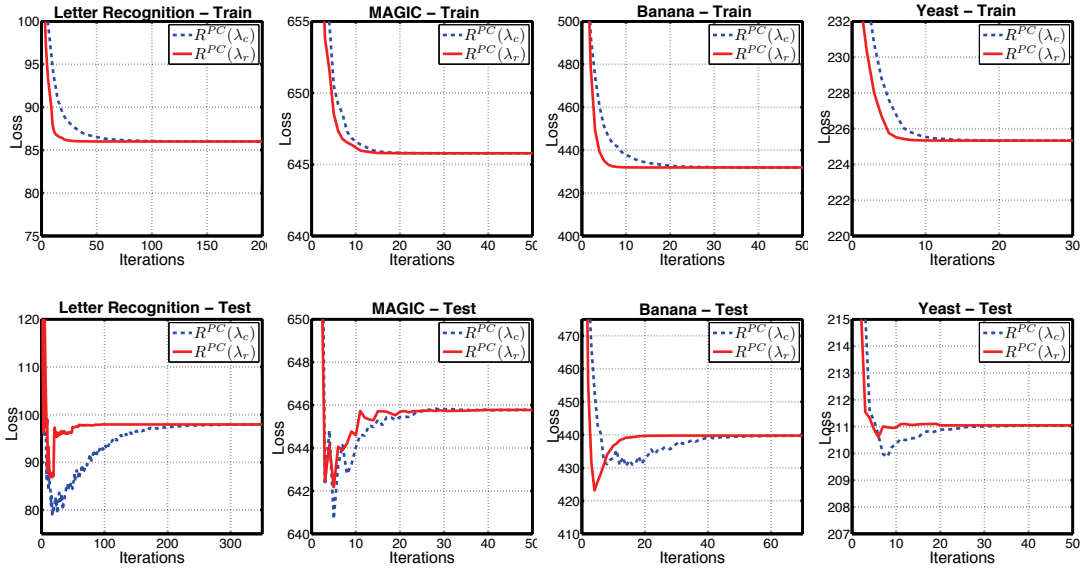


Figure 4: Verifying the backward direction of the equivalence for P-Classification and P-Norm Push (p=4). $\boldsymbol{\lambda}_r$ are corrected with b that is defined in (12).

1. **Input:** Examples $\{(x_i, y_i)\}_{i=1}^m$, where $x_i \in \mathcal{X}$, $y_i \in \{-1, 1\}$, features $\{h_j\}_{j=1}^n$, $h_j : \mathcal{X} \to \mathcal{R}$, number of iterations $t_{\max}$.

2. **Define:** $M_{ik,j} := h_j(x_i) - h_j(\tilde{x}_k)$ for all i, k, j, where the first index is over all positive-negative pairs indexed by ik, for $ik = 1, \ldots, IK$.

3. **Initialize:** $\lambda_{1,j} = 0$ for $j = 1, \ldots, n$, $\quad d_{1,ik} = 1/IK$ for $ik = 1, \ldots, IK$.

4. **Loop for** $t = 1, ..., t_{\max}$

   (a) $j_t \in \operatorname{argmax}_j \sum_{ik} d_{t,ik} M_{ik,j}$

   (b) Perform a linesearch for α_t. That is, find a value α_t that minimizes:

$$\left(\sum_{ik=1}^{IK} M_{ik,j_t} \frac{1}{1 + e^{\left[\left(\sum_j M_{ik,j_t} \lambda_t \right) + \alpha_{ik} M_{ik,j_t} \right]}} \right)^2$$

   (c) $\lambda_{t+1,j} = \lambda_{t,j} + \alpha_t \mathbb{1}_{j=j_t}$

   (d) $d_{t+1,ik} = \frac{1}{1+e^{\left(\sum_j M_{ik,j} \lambda_{t+1,j} \right)}}$ for $i = 1, \ldots, I, k = 1, \ldots, K$

   (e) $d_{t+1,ik} = \frac{d_{t+1,ik}}{\sum_{ik} d_{t+1,ik}}$

5. **Output:** $\boldsymbol{\lambda}_{t_{\max}}$

Figure 5: Pseudocode for the Logistic Regression Ranking algorithm.

Theorem 4 *Probability estimates for P-Classification and for the P-Norm Push algorithm (with $\boldsymbol{\lambda}$ corrected trivially as in Theorem 2) can be obtained from the scoring function $f_{\boldsymbol{\lambda}}(x)$ as follows:*

$$\hat{P}(y = 1|x) = \frac{1}{1 + e^{-f_{\boldsymbol{\lambda}}(x)(1+p)}}.$$

Proof This proof (in some sense) generalizes results from the analysis of AdaBoost and logistic regression (see Friedman et al., 2000; Schapire and Freund, 2011). A general classification objective function can be regarded as an estimate for the expected loss:

$$\mathcal{R}_{\text{true}}(f) := \mathbb{E}_{x,y\sim D}[l(y, f(x))],$$

where expectation is over randomly selected examples from the true distribution $\mathcal{D}$. This quantity can be split into two terms, as follows:

$$\begin{aligned}\mathbb{E}_{x,y\sim D}[l(y, f(x))] &= \mathbb{E}_x \left[\mathbb{E}_{y|x}[l(y, f(x)|x)] \right] \\ &= \mathbb{E}_x[P(y = 1|x) l(1, f(x)) + (1 - P(y = 1|x)) l(-1, f(x))].\end{aligned}$$

At each x, differentiating the inside with respect to $f(x)$ and setting the derivative to 0 at the point $f(x) = f^*(x)$, we obtain:

$$0 = P(y = 1|x) l'(1, f^*(x)) + (1 - P(y = 1|x)) l'(-1, f^*(x)),$$

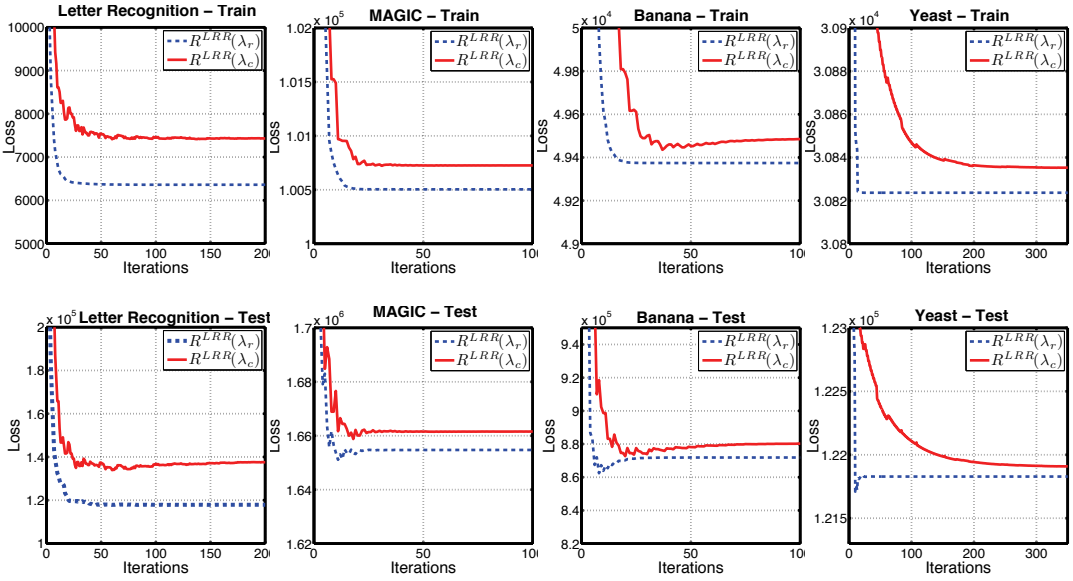


Figure 6: Doubt on equivalence relationship for logistic regression and LRR

and solving this equation for $P(y=1|x)$,

$$P(y=1|x) \quad = \quad \frac{1}{\frac{l'(1,f^*(x))}{-l'(-1,f^*(x))}+1}. \tag{17}$$

P-Classification's objective is an empirical sum over the training instances rather than an expectation over the true distribution:

$$\mathcal{R}^{PC}(\lambda) = \sum_{i=1}^{I} e^{-f_{\boldsymbol{\lambda}}(x_i)} + \sum_{k=1}^{K} \frac{1}{p} e^{f_{\boldsymbol{\lambda}}(\tilde{x}_k)p}.$$

Thus, estimates of the conditional probabilities $\hat{P}(y=1|x)$ can be obtained using (17) where:

$$l(1,f(x)) = e^{-f(x)} \text{ and } l(-1,f(x)) = \frac{1}{p} e^{f(x)p},$$

and instead of $f^*(x)$, which cannot be calculated, we use $f_{\boldsymbol{\lambda}}(x)$. To do this, we first find derivatives:

$$\begin{aligned} l(1,f(x)) = e^{-f(x)} \quad &\Rightarrow \quad l'(1,f(x)) = -e^{-f(x)} \\ l(-1,f(x)) = \frac{1}{p} e^{f(x)p} \quad &\Rightarrow \quad l'(-1,f(x)) = e^{f(x)p}. \end{aligned}$$

Substituting into (17), we obtain estimated conditional probabilities as follows:

$$\hat{P}(y=1|x) = \frac{1}{1+\frac{l'(1,f_{\boldsymbol{\lambda}}(x))}{-l'(-1,f_{\boldsymbol{\lambda}}(x))}} = \frac{1}{1+\frac{-e^{-f_{\boldsymbol{\lambda}}(x)}}{-e^{f_{\boldsymbol{\lambda}}(x)p}}} = \frac{1}{1+e^{-f_{\boldsymbol{\lambda}}(x)(1+p)}}.$$

This expression was obtained for P-Classification, and extends to the P-Norm Push (with $\boldsymbol{\lambda}$ corrected) by the equivalence relationship of Theorem 2. ■

Note that for $p=1$, Theorem 4 yields conditional probabilities for RankBoost.

Letter Recognition				
	Algorithm	*Objective*	*# of Iterations(for .05%)*	*Time (sec.)*
1K Train	AdaBoost	$\mathcal{R}^{RB}$	123.4±21.6	**0.1**±0.0
	RankBoost	$\mathcal{R}^{RB}$	50.2±10.4	1.1±0.3
	P-Classification	$\mathcal{R}^{PN}$	132.5±23.9	**0.6**±0.1
	PNormPush	$\mathcal{R}^{PN}$	43.3±7.0	3.3±0.6
	Logistic Regression	$\mathcal{R}^{LRR}$	N/A	N/A
	LRR	$\mathcal{R}^{LRR}$	50.0±11.9	8.0±3.0
3K Train	AdaBoost	$\mathcal{R}^{RB}$	112.3±28.5	**0.1**±0.0
	RankBoost	$\mathcal{R}^{RB}$	43.7±9.3	10.7±2.7
	P-Classification	$\mathcal{R}^{PN}$	136.5±36.4	**1.3**±0.3
	PNormPush	$\mathcal{R}^{PN}$	44.0±8.8	29.3±6.5
	Logistic Regression	$\mathcal{R}^{LRR}$	N/A	N/A
	LRR	$\mathcal{R}^{LRR}$	43.8±11.4	65.8±18.2
5K Train	AdaBoost	$\mathcal{R}^{RB}$	108.9±18.8	**0.1**±0.0
	RankBoost	$\mathcal{R}^{RB}$	43.2±7.5	29.2±7.8
	P-Classification	$\mathcal{R}^{PN}$	138.3±29.4	**1.7**±0.3
	PNormPush	$\mathcal{R}^{PN}$	41.9±6.2	72.0±12.5
	Logistic Regression	$\mathcal{R}^{LRR}$	N/A	N/A
	LRR	$\mathcal{R}^{LRR}$	44.4±8.3	218.3±40.6

Table 2: Comparison of runtime performances over varying training set sizes. $p = 4$ for P-Classification and P-Norm Push.

5.2 Runtime Performances

Faster runtime is a major practical benefit of the equivalence relationship proved in Section 3. As we have shown in Sections 4.2 and 4.3, when comparing how ranking algorithms and classification algorithms approach the minimizers of the misranking error, the ranking algorithms tend to converge more rapidly in terms of the number of iterations. Convergence with fewer iterations, however, does not translate into *faster* convergence. Each iteration of either algorithm requires a search for the optimal weak hypothesis j. For the P-Norm Push, each comparison requires quadratic space (involving a vector multiplication of size $I \times K$). In contrast, P-Classification's comparisons are linear in the number of examples (involving a vector multiplication of size $I + K$). For RankBoost, note that a more efficient implementation is possible for bipartite ranking (see Section 3.2 of Freund et al., 2003), though a more efficient implementation has not previously been explored in general for the P-Norm Push; in fact, the equivalence relationship allows us to use P-Classification instead, making it somewhat redundant to derive one.

Table 2 presents the number of iterations and the amount of time required to train each of the algorithms (AdaBoost, RankBoost, P-Classification for p=4, P-Norm Push for p=4, logistic regression, and Logistic Regression Ranking) in an experiment, using a 2.53 gHz macbook pro with 4 GB ram. We used the RankBoost algorithm with the "second method" for computing α_t given by Freund et al. (2003), since it is the special case corresponding to the P-Norm Push with

$p = 1$. The results are presented for the Letter Recognition data set, which is the largest data set in our experimental corpus. To assess the scalability of the algorithms, we generated 10 training sets each of sizes 1000 examples, 3000 examples, and 5000 examples (30 total training sets). For each algorithm, we report the mean and variance (over 10 training sets) of the number of iterations and the time elapsed (in seconds) for the ranking loss to be within 0.05% of the asymptotic minimum ranking loss. The asymptotic value was obtained using 200 iterations of the corresponding ranking algorithm (for AdaBoost and RankBoost, we used RankBoost; for P-Classification and the P-Norm Push, we used the P-Norm Push; and for logistic regression and LRR, we used LRR). Note that logistic regression may never converge to within 0.05% of the ranking loss obtained by LRR (there is no established equivalence relationship), so "N/A" has been placed in the table when this occurs.

Comparing the runtime performances of classification and ranking algorithms in Table 2, AdaBoost and P-Classification yield dramatic improvement over their ranking counterparts. Despite the fact that they require more than double the number of iterations to obtain the same quality of solution, it only takes them a fraction of the time. Further, AdaBoost and P-Classification appear to scale better with the sample size. Going from 1K to 5K, AdaBoost's run time roughly doubles, on average from 0.08 to 0.14 seconds, whereas RankBoost takes over 27 times longer ($29.19/1.08 \approx 27$). Similarly, P-Classification's run time on the 5K data set is slightly more than twice the run time on the 1K data set, as opposed to approximately 22 times longer ($72/3.32 \approx 21.7$) for P-Norm Push on the 5K data set. Thus, the equivalence relationship between classification and ranking algorithms enables us to pass the efficiency of classification algorithms to their ranking counterparts, which leads to significant speed improvement for ranking tasks.

6. Experiments on Prediction Performance

When evaluating the prediction performance of the P-Classification algorithm, we chose *precision* as our performance metric, motivated by a specific relationship between precision and P-Classification's objective that we derive in this section. In Information Retrieval (IR) contexts, precision is defined as the number of relevant instances retrieved as a result of a query, divided by the total number of instances retrieved. Similarly, in a classification task the precision is defined as

$$\text{Precision} := \frac{\text{TP}}{\text{TP} + \text{FP}}$$

where TP (true positives) are the number of instances correctly labeled as belonging to the positive class and FP (false positives) are the number of instances incorrectly labeled as belonging to the positive class. In a classification task, 100% precision means that every instance labeled as belonging to the positive class does indeed belong to the positive class, whereas 0% precision means that all positive instances are misclassified.

In order to derive the relationship between precision and P-Classification's objective, consider P-Classification's objective:

$$\mathcal{R}^{PC}(\boldsymbol{\lambda}) = \sum_{i=1}^{I} e^{-f_{\boldsymbol{\lambda}}(x_i)} + \frac{1}{p} \sum_{k=1}^{K} e^{f_{\boldsymbol{\lambda}}(\tilde{x}_k)p}. \tag{18}$$

There is a potential for the second term to be much larger than the first term when $p > 1$, so we consider:

$$\mathcal{R}^{PC}(\boldsymbol{\lambda}) \geq \frac{1}{p}\sum_k e^{f_{\boldsymbol{\lambda}}(\tilde{x}_k)p} \tag{19}$$

$$\geq \frac{1}{p}\sum_k e^{\gamma p}\mathbb{1}_{[f_{\boldsymbol{\lambda}}(\tilde{x}_k)\geq\gamma]} = \frac{1}{p}e^{\gamma p}\sum_k \mathbb{1}_{[f_{\boldsymbol{\lambda}}(\tilde{x}_k)\geq\gamma]}. \tag{20}$$

Transitioning from (19) to (20) uses the fact that $e^{f_{\boldsymbol{\lambda}}(\tilde{x}_k)} > e^{\gamma p}$ when $f_{\boldsymbol{\lambda}}(\tilde{x}_k) \geq \gamma,\ \forall\gamma$. Let $I_{f_{\boldsymbol{\lambda}}>\gamma}, K_{f_{\boldsymbol{\lambda}}>\gamma}$ denote the number of positive and negative instances that score higher than the cutoff threshold γ, respectively. Then,

$$\begin{aligned}\sum_k \mathbb{1}_{[f_{\boldsymbol{\lambda}}(\tilde{x}_k)\geq\gamma]} &= K_{f_{\boldsymbol{\lambda}}\geq\gamma} \\ &= \left(I_{f_{\boldsymbol{\lambda}}\geq\gamma} + K_{f_{\boldsymbol{\lambda}}\geq\gamma}\right)\left(1 - \frac{I_{f_{\boldsymbol{\lambda}}\geq\gamma}}{I_{f_{\boldsymbol{\lambda}}\geq\gamma} + K_{f_{\boldsymbol{\lambda}}\geq\gamma}}\right) \\ &= \left(I_{f_{\boldsymbol{\lambda}}\geq\gamma} + K_{f_{\boldsymbol{\lambda}}\geq\gamma}\right)\left(1 - \text{Precision@}(f_{\boldsymbol{\lambda}} = \gamma)\right). \end{aligned} \tag{21}$$

Note that $I_{f_{\boldsymbol{\lambda}}\geq\gamma} + K_{f_{\boldsymbol{\lambda}}\geq\gamma}$ is simply the number of all instances with scores greater than γ. Plugging (21) into (20) yields

$$\mathcal{R}^{PC}(\lambda) \geq \frac{1}{p}e^{\gamma p}\left(I_{f_{\boldsymbol{\lambda}}\geq\gamma} + K_{f_{\boldsymbol{\lambda}}\geq\gamma}\right)\left(1 - \text{Precision@}(f_{\boldsymbol{\lambda}} = \gamma)\right)$$

which indicates that minimizing P-Classification's objective may yield solutions that have high precision. Through the equivalence of the P-Norm Push and P-Classification, a similar relationship with precision also exists for the P-Norm Push.

6.1 Effect of p:

In this section, we explore the prediction performance of the P-Classification algorithm with respect to p, for various levels of γ, where γ is the cutoff threshold for calculating precision. We have three hypotheses that we want to investigate, regarding the relationship between p and precision.

Hypothesis 1: The presence of exponent p in the second term in (18) enables P-Classification to achieve improved prediction performance.

The first term of (18) can be much smaller than the second term, due mainly to the presence of p in the exponent. This means that the bound in (19) becomes tighter with the presence of p. This may indicate that the exponent p can influence the algorithm's performance with respect to precision. The empirical analysis that we present later in this section investigates the influence and impact of the exponent p on precision accuracy.

Hypothesis 2: Increasing p in P-Classification's objective yields improved prediction performance.

As p increases, the bound in (19) becomes tighter and the largest terms in $\mathcal{R}^{PC}$ correspond to the highest scoring negative examples. Minimizing $\mathcal{R}^{PC}$ thus "pushes" these negative examples down the ranked list, potentially leading to higher values of precision.

Hypothesis 3: P-Classification can achieve better performance than AdaBoost.

P-Classification is a generalized version of AdaBoost. We hypothesize that as p increases, it will be possible to obtain better prediction performance.

We are able to make the hypotheses above since we have chosen precision to be the performance metric. Another metric for evaluating the performance of a ranking function is *recall*, which is defined as the number of true positives divided by the total number of positive examples. A perfect recall of 100% indicates that all positive examples are above the cutoff threshold. Therefore, if our goal was to optimize recall instead of precision, we would want to put the exponent p on the first term of $\mathcal{R}^{PC}$ rather than the second term, since it will create the effect of pushing the positive examples from bottom to top of the list. As the goal is to concentrate on the correct rankings at the top of the list, we particularly aim at achieving higher precision, rather than higher recall. In many IR systems, including web search, what matters most is how many relevant (positive) results there are on the first page or the first few pages—this is reflected directly by precision. Recall does not accurately represent the performance at the top of the list, since it concerns the performance across *all* of the positives; this would require us to go much farther down the list than is reasonable to consider for these applications, in order to span all of the positive examples.

We will now describe the experimental setup. We chose training sets as follows: for MAGIC, 1000 randomly chosen examples, for Yeast, 500 randomly chosen examples, for Banana, 1000 randomly chosen examples and for Letter Recognition, 1200 examples with 200 positives and 1000 positives to achieve an imbalance ratio of 1:5. Increasing the number of positive examples in the training set of Letter Recognition enabled us to keep the "*x-ege*" attribute, but discard only the "*x2bar*" attribute, due to RankBoost's requirement discussed in Section 4.1. For all data sets except Yeast, we randomly selected 4000 examples as the test set. For Yeast, we selected the remaining 983 examples as the test set. The experiments were conducted for three cutoff thresholds γ, to consider the top 50%, 25% and 10% of the list. The algorithms were run until they had converged (hundreds of iterations).

Table 3 presents the precision results on the test sets from all four data sets. In order to investigate the hypotheses above, we redefine Cost-Sensitive AdaBoost as an algorithm that minimizes the following objective:

$$\mathcal{R}^{AB}_{es}(\boldsymbol{\lambda}) = \sum_{i=1}^{I} e^{-y_i f_{\boldsymbol{\lambda}}(x_i)} + C\frac{1}{p}\sum_{k=1}^{K} e^{-y_k f_{\boldsymbol{\lambda}}(\tilde{x}_k)}. \tag{22}$$

In order to test the first hypothesis, we fix $C = 1$. When $C = 1$, (22) resembles P-Classification's objective in (7), the only difference is that p is missing in the exponent. When $C = 1$ and $p = 1$, (22) reduces to AdaBoost's objective (3). In that case, P-Classification and AdaBoost give the same performance trivially. As shown in Table 3, for fixed values of p, where $p > 1$, our experiments indicate that P-Classification yields higher precision than Cost-Sensitive AdaBoost, which agrees with our first hypothesis. To see this, compare element-wise the "AdaB.CS" rows $(C = 1, p > 1)$ in the table with the corresponding "P-Classification" rows; this is 36 comparisons that all show P-Classification giving higher precision than AdaBoost. This was a controlled experiment in which the treatment was the presence of the exponent p. Our results indicate that the presence of the exponent p can be highly effective in achieving higher precision values.

In order to test the second hypothesis, we investigated the impact of increasing p in P-Classification's objective. We considered four values of p for each of three cutoff thresholds. Table 3 shows that increasing p in P-Classification's objective leads to higher precision values. To see this, consider the P-Classification rows in Table 3. Within each column of the table, performance improves as p increases. With increasing p, P-Classification focuses more on pushing down the high-scored negative instances from top of the list, yielding higher precision.

		MAGIC			Letter Recognition		
		50%	25%	10%	50%	25%	10%
AdaB.$_{CS}$	p=1,C=1	63.94±3.26	69.40±2.75	94.75±1.03	66.88±8.13	66.88±8.13	72.78±7.20
	p=2,C=1	55.92±2.68	69.40±2.75	94.75±1.03	61.76±8.14	61.76±8.14	72.60±6.99
	p=3,C=1	54.25±1.41	69.40±2.75	94.75±1.03	58.91±7.61	58.91±7.61	72.82±7.24
	p=4,C=1	54.25±1.41	69.40±2.75	94.75±1.03	55.61±7.23	56.02±6.82	72.78±7.20
	p=4,C=$\frac{\#neg}{\#pos}$	55.20±2.35	69.40±2.75	94.75±1.03	68.42±7.09	68.42±7.09	72.88±7.33
P-Class.	p=1,C=1	63.94±3.26	69.40±2.75	94.75±1.03	66.88±8.13	66.88±8.13	72.78±7.20
	p=2,C=1	64.45±3.13	70.22±2.77	94.97±0.97	68.98±8.02	68.98±8.02	77.20±7.62
	p=3,C=1	65.21±2.99	70.46±2.60	95.15±0.91	70.09±7.65	70.09±7.65	78.45±8.63
	p=4,C=1	65.13±2.96	70.60±2.62	95.15±0.94	70.38±7.43	70.38±7.43	78.93±8.41
	p=4,C=$\frac{\#neg}{\#pos}$	82.27±7.68	82.49±7.08	95.15±0.94	90.96±3.39	90.96±3.39	90.96±3.39
LR		65.31±2.90	69.51±2.65	94.78±1.08	69.49±7.63	69.49±7.63	77.10±8.22
LR$_{CS}$	C=$\frac{\#neg}{\#pos}$	84.54±8.64	84.54±8.64	95.00±0.96	90.62±4.18	90.62±4.18	90.62±4.18
		Banana			**Yeast**		
		50%	25%	10%	50%	25%	10%
AdaB.$_{CS}$	p=1,C=1	75.56±0.39	88.77±1.75	90.35±2.49	52.77±4.59	52.77±4.59	56.27±3.87
	p=2,C=1	75.56±0.39	88.77±1.75	90.35±2.49	44.24±2.54	49.11±3.91	56.06±3.85
	p=3,C=1	74.91±0.94	88.77±1.75	90.35±2.49	42.66±1.56	49.11±3.91	56.06±3.85
	p=4,C=1	74.15±1.09	88.77±1.75	90.35±2.49	42.66±1.56	49.11±3.91	56.06±3.85
	p=4,C=$\frac{\#neg}{\#pos}$	75.56±0.39	88.77±1.75	90.35±2.49	42.66±1.56	49.11±3.91	56.06±3.85
P-Class.	p=1,C=1	75.56±0.39	88.77±1.75	90.35±2.49	52.77±4.59	52.77±4.59	56.27±3.87
	p=2,C=1	75.57±0.39	89.16±0.60	96.70±2.79	53.20±4.23	53.20±4.23	56.37±4.66
	p=3,C=1	75.57±0.39	89.16±0.60	97.80±0.86	53.25±4.22	53.25±4.22	56.47±5.38
	p=4,C=1	75.57±0.39	89.16±0.60	98.55±1.02	53.26±4.22	53.26±4.22	56.98±4.74
	p=4,C=$\frac{\#neg}{\#pos}$	88.75±9.45	92.86±2.86	98.55±1.02	56.37±6.26	56.37±6.26	58.12±6.20
LR		75.57±0.39	89.16±0.60	90.35±2.49	53.23±4.24	53.23±4.24	56.57±5.15
LR$_{CS}$	C=$\frac{\#pos}{\#neg}$	88.75±9.45	92.86±2.86	90.35±2.49	55.88±6.09	55.88±6.09	57.59±6.57

Table 3: Precision values at the top 50%, 25% and 10% of the ranked list.

Evaluating the third hypothesis, Table 3 shows that P-Classification for $p > 1$ yields superior precision than AdaBoost in all comparisons (36 of them in total, from 4 data sets, 3 γ levels and 3 values of $p > 1$).

6.2 Effect of Parameter C

Our next set of experiments focus on the behavior that we observe in Cost-Sensitive AdaBoost's results, which is that increasing p has a detrimental effect on precision, whereas in P-Classification, increasing p leads to higher precision. Given that the only difference between $\mathcal{R}^{PC}$ and $\mathcal{R}^{AB}_{cs}$ is the presence of the exponent p in $\mathcal{R}^{PC}$, the behavior that we observe can be explained by the hypothesis that the exponent p in P-Classification is the dominant factor in determining the misclassification penalty on the negative examples, overwhelming the effect of the $\frac{1}{p}$ factor.

This leaves room for the possibility that altering the $\frac{1}{p}$ factor could lead to improved performance. We tested this possibility as follows: first, we varied the coefficient C in Cost-Sensitive AdaBoost's objective in (22); second, we introduced the same C into P-Classification's objective,

and thus defined a Cost-Sensitive P-Classification algorithm that minimizes the following loss:

$$\mathcal{R}_{cs}^{PC}(\boldsymbol{\lambda}) = \sum_{i=1}^{I} e^{-f_{\boldsymbol{\lambda}}(x_i)} + C\frac{1}{p}\sum_{k=1}^{K} e^{f_{\boldsymbol{\lambda}}(\tilde{x}_k)p}.$$

In the experiments, we heuristically set $C = \frac{\#neg}{\#pos}$ in order to reduce, and possibly eliminate, the detrimental effect of the $\frac{1}{p}$ term. Our data sets share characteristics similar to many other real world data sets in that the number of positive examples is less than the number of negative examples; therefore $C > 1$ for all four data sets. The $\frac{\#neg}{\#pos}$ ratios averaged over 10 splits for each data set are 354/646, 200/1000, 400/600 and 155/345 for MAGIC, Letter Recognition, Banana and Yeast, respectively. The last row in Table 3 for Cost-Sensitive AdaBoost, and also the last row for P-Classification, contains performance results with $C = \frac{\#neg}{\#pos}$. As seen, for a fixed p ($p = 4$ in this case), using this new value for C dramatically improves precision for P-Classification in most cases (10 out of 12 comparisons) and for AdaBoost in some cases (5 out of 12 comparisons). To see this, compare the $p = 4, C = 1$ row with the $p = 4, C = \frac{\#neg}{\#pos}$ row for each algorithm. Using $C > 1$ is equivalent to giving higher misclassification penalty to the negative examples, which can result in a stronger downward push on these examples, raising precision.

6.3 Comparison of P-Classification With Logistic Regression

Table 3 also presents results for logistic regression, both using its original formulation (5) as well as its cost-sensitive variant; the objective for cost-sensitive logistic regression is formulated by multiplying the second term of logistic regression's objective in (5) with the coefficient C. In comparing P-Classification (p=4) with logistic regression, P-Classification performed worse than logistic regression in only one comparison out of 12 (3 γ levels and 4 data sets). Considering their cost-sensitive variants with $C = \frac{\#neg}{\#pos}$, P-Classification and logistic regression generally outperformed their original (non-cost-sensitive) formulations (10 out of 12 comparisons for P-Classification vs. Cost-Sensitive P-Classification with p=4, and 11 out of 12 comparisons for logistic regression vs cost-sensitive logistic regression). Furthermore, Cost-Sensitive P-Classification performed worse than cost-sensitive logistic regression in only 2 out of 12 comparisons.

7. A Hybrid Approach for Logistic Regression

As we discussed in Section 4.4, logistic regression and LRR do not seem to exhibit the equivalence property that we have established for boosting-style algorithms. Consequently, neither logistic regression or LRR may have the benefit of low classification loss and ranking loss simultaneously. This limitation can be mitigated to some degree, through combining the benefits of logistic regression and LRR into a single hybrid algorithm that aims to solve both classification and ranking problems simultaneously. We define the hybrid loss function as:

$$\mathcal{R}^{LR+LRR} = \mathcal{R}^{LR} + \beta\mathcal{R}^{LRR}$$

where β denotes a non-negative regularization factor. $\beta = 0$ reduces the hybrid loss to that of logistic regression, whereas increasing β tilts the balance towards LRR. The trade-off between classification and ranking accuracy is shown explicitly in Figure 7, which presents the 0-1 classification loss and 0-1 ranking loss of this hybrid approach at various β settings. For comparison, we have included

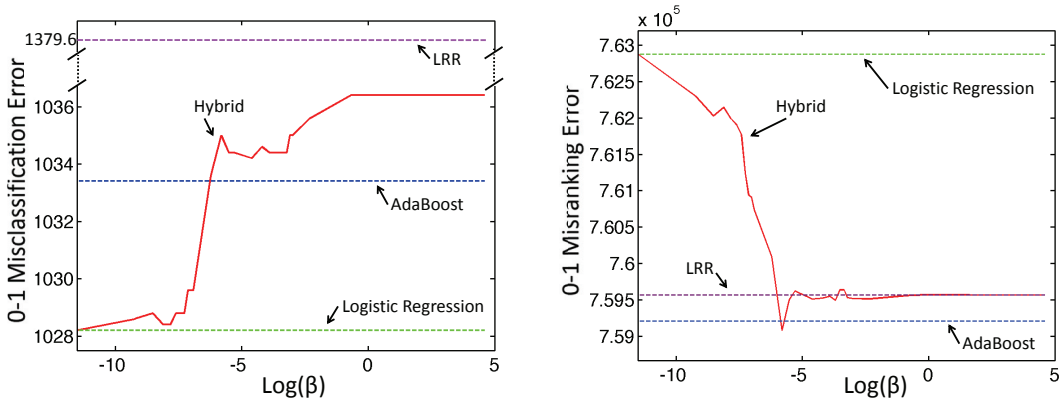


Figure 7: Effect of β on the misclassification error and misranking error rates for MAGIC data set.

the baseline performance of AdaBoost, logistic regression and LRR. For this particular experiment, logistic regression was able to achieve a better misclassification result than AdaBoost (see Figure 7(a)), but at the expense of a very large misranking error (see Figure 7(b)). As β increases, Figure 7(b) shows that the misranking error decreases almost to the level of AdaBoost's, whereas Figure 7(a) shows that the classification error increases to be higher than AdaBoost's. The value of β should be chosen based on the desired performance criteria for the specific application, determining the balance between desired classification vs ranking accuracy.

8. Conclusion

We showed an equivalence relationship between two algorithms for two different tasks, based on a relationship between the minimizers of their objective functions. This equivalence relationship provides an explanation for why these algorithms perform well with respect to multiple evaluation metrics, it allows us to compute conditional probability estimates for ranking algorithms, and permits the solution of ranking problems an order of magnitude faster. The two algorithms studied in this work are generalizations of well-known algorithms AdaBoost and RankBoost. We showed that our new classification algorithm is related to a performance metric used for ranking, and studied empirically how aspects of our new classification algorithm influence ranking performance. This allowed us to suggest improvements to the algorithm in order to boost performance. Finally, we presented a new algorithm inspired by logistic regression that solves a task that is somewhere between classification and ranking, with the goal of providing solutions to both problems. This suggests many avenues for future work. For instance, it may be possible to directly relate either the objective of P-Classification or the P-Norm Push to other performance metrics (see also the discussion in Rudin, 2009). It may also be interesting to vary the derivation of P-Classification to include an exponent on both terms in order to handle, for instance, both precision and recall.

Acknowledgments

This material is based upon work supported by the National Science Foundation under Grant No IIS-1053407. We also gratefully acknowledge support from the MIT Energy Initiative. Thanks to David McAllester and Vivek Farias for helpful discussions.

References

Maria-Florina Balcan, Nikhil Bansal, Alina Beygelzimer, Don Coppersmith, John Langford, and Gregory B. Sorkin. Robust reductions from ranking to classification. *Machine Learning*, 72: 139–153, 2008.

Leo Breiman. Arcing the edge. Technical Report 486, Statistics Department, University of California at Berkeley, 1997.

Chris Burges, Tal Shaked, Erin Renshaw, Ari Lazier, Matt Deeds, Nicole Hamilton, and Greg Hullender. Learning to rank using gradient descent. In *Proceedings of the 22nd International Conference on Machine Learning (ICML)*, pages 89–96, 2005.

Rich Caruana and Alexandru Niculescu-Mizil. An empirical comparison of supervised learning algorithms. In *Proceedings of the 23rd International Conference on Machine Learning (ICML)*, pages 161–168, 2006.

Michael Collins, Robert E. Schapire, and Yoram Singer. Logistic regression, AdaBoost and Bregman distances. *Machine Learning*, 48:253–285, 2002.

Nigel Duffy and David P. Helmbold. A geometric approach to leveraging weak learners. In *European Conference on Computational Learning Theory (EuroCOLT)*. Springer-Verlag, 1999.

Wei Fan, Salvatore J. Stolfo, Junxin Zhang, and Philip K. Chan. Adacost: Misclassification cost-sensitive boosting. In *Proceedings of the 16th International Conference on Machine Learning (ICML)*, pages 97–105, 1999.

Andrew Frank and Arthur Asuncion. UCI machine learning repository, 2010. URL `http://archive.ics.uci.edu/ml`.

Yoav Freund and Robert E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences*, 55:119–139, 1997.

Yoav Freund, Raj Iyer, Robert E. Schapire, and Yoram Singer. An efficient boosting algorithm for combining preferences. *Journal of Machine Learning Research (JMLR)*, 4:933–969, 2003.

Jerome Friedman, Trevor Hastie, and Robert Tibshirani. Additive logistic regression: A statistical view of boosting. *Annals of Statistics*, 28(2):337–374, 2000.

Wojciech Kotlowski, Krzysztof Dembczynski, and Eyke Huellermeier. Bipartite ranking through minimization of univariate loss. In *Proceedings of the 28th International Conference on Machine Learning (ICML)*, pages 1113–1120, 2011.

Aurélie C. Lozano and Naoki Abe. Multi-class cost-sensitive boosting with p-norm loss functions. In *Proceeding of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 506–514, 2008.

Llew Mason, Jonathan Baxter, Peter Bartlett, and Marcus Frean. Boosting algorithms as gradient descent. In *Proceedings of Neural Information Processing Systems (NIPS)*, 2000.

David Mease, Abraham J. Wyner, and Andreas Buja. Boosted classification trees and class probability/quantile estimation. *Journal of Machine Learning Research (JMLR)*, 8:409–439, 2007.

Indraneel Mukherjee, Cynthia Rudin, and Robert Schapire. The rate of convergence of AdaBoost. In *Proceedings of the 24th Annual Conference on Learning Theory (COLT)*, 2011.

Gunnar Rätsch, Takashi Onoda, and Klaus-Robert Müller. Soft margins for AdaBoost. *Machine Learning*, 42(3):287–320, 2001.

Cynthia Rudin. The P-Norm Push: A simple convex ranking algorithm that concentrates at the top of the list. *Journal of Machine Learning Research (JMLR)*, 10:2233–2271, 2009.

Cynthia Rudin and Robert E. Schapire. Margin-based ranking and an equivalence between AdaBoost and RankBoost. *Journal of Machine Learning Research (JMLR)*, 10:2193–2232, 2009.

Robert E. Schapire and Yoav Freund. *Boosting: Foundations and Algorithms*. MIT Press, 2011. In Preparation.

Yanmin Sun, Mohamed S. Kamel, Andrew K. C. Wong, and Yang Wang. Cost-sensitive boosting for classification of imbalanced data. *Pattern Recognition*, 40:3358–3378, 2007.

David H. Wolpert and William G. Macready. No free lunch theorems for optimization. *IEEE Transactions on Evolutionary Computation*, 1(1):67–82, 1997.

Journal of Machine Learning Research 12 (2011) 2931-2974 Submitted 11/09; Revised 3/11; Published 10/11

Hierarchical Knowledge Gradient for Sequential Sampling

Martijn R.K. Mes M.R.K.MES@UTWENTE.NL
Department of Operational Methods for Production and Logistics
University of Twente
Enschede, The Netherlands

Warren B. Powell POWELL@PRINCETON.EDU
Department of Operations Research and Financial Engineering
Princeton University
Princeton, NJ 08544, USA

Peter I. Frazier PF98@CORNELL.EDU
Department of Operations Research and Information Engineering
Cornell University
Ithaca, NY 14853, USA

Editor: Ronald Parr

Abstract

We propose a sequential sampling policy for noisy discrete global optimization and ranking and selection, in which we aim to efficiently explore a finite set of alternatives before selecting an alternative as best when exploration stops. Each alternative may be characterized by a multi-dimensional vector of categorical and numerical attributes and has independent normal rewards. We use a Bayesian probability model for the unknown reward of each alternative and follow a fully sequential sampling policy called the knowledge-gradient policy. This policy myopically optimizes the expected increment in the value of sampling information in each time period. We propose a hierarchical aggregation technique that uses the common features shared by alternatives to learn about many alternatives from even a single measurement. This approach greatly reduces the measurement effort required, but it requires some prior knowledge on the smoothness of the function in the form of an aggregation function and computational issues limit the number of alternatives that can be easily considered to the thousands. We prove that our policy is consistent, finding a globally optimal alternative when given enough measurements, and show through simulations that it performs competitively with or significantly better than other policies.

Keywords: sequential experimental design, ranking and selection, adaptive learning, hierarchical statistics, Bayesian statistics

1. Introduction

We address the problem of maximizing an unknown function θ_x where $x = (x_1, \ldots, x_D)$, $x \in \mathcal{X}$, is a discrete multi-dimensional vector of categorical and numerical attributes. We have the ability to sequentially choose a set of measurements to estimate θ_x, after which we choose the value of x with the largest estimated value of θ_x. Our challenge is to design a measurement policy which produces fast convergence to the optimal solution, evaluated using the expected objective function after a specified number of iterations. Many applications in this setting involve measurements that are time consuming and/or expensive. This problem is equivalent to the ranking and selection (R&S)

problem, where the difference is that the number of alternatives $|\mathcal{X}|$ is extremely large relative to the measurement budget.

We do not make any explicit structural assumptions about θ_x, but we do assume that we are given an ordered set $\mathcal{G}$ and a family of aggregation functions $G^g : \mathcal{X} \to \mathcal{X}^g$, $g \in \mathcal{G}$, each of which maps $\mathcal{X}$ to a region $\mathcal{X}^g$, which is successively smaller than the original set of alternatives. After each observation $\hat{y}_x^n = \theta_x + \varepsilon^n$, we update a family of statistical estimates of θ at each level of aggregation. After n observations, we obtain a family of estimates $\mu_x^{g,n}$ of the function at different levels of aggregation, and we form an estimate μ_x^n of θ_x using

$$\mu_x^n = \sum_{g \in \mathcal{G}} w_x^{g,n} \mu_x^{g,n}, \tag{1}$$

where the weights $w_x^{g,n}$ sum to one over all the levels of aggregation for each point x. The estimates $\mu_x^{g,n}$ at more aggregate levels have lower statistical variance since they are based upon more observations, but exhibit aggregation bias. The estimates $\mu_x^{g,n}$ at more disaggregate levels will exhibit greater variance but lower bias. We design our weights to strike a balance between variance and bias.

Our goal is to create a measurement policy π that leads us to find the alternative x that maximizes θ_x. This problem arises in a wide range of problems in stochastic search including (i) which settings of several parameters of a simulated system has the largest mean performance, (ii) which combination of chemical compounds in a drug would be the most effective to fight a particular disease, and (iii) which set of features to include in a product would maximize profits. We also consider problems where x is a multi-dimensional set of continuous parameters.

A number of measurement policies have been proposed for the ranking and selection problem when the number of alternatives is not too large, and where our beliefs about the value of each alternative are independent. We build on the work of Frazier et al. (2009) which proposes a policy, the knowledge-gradient policy for correlated beliefs, that exploits correlations in the belief structure, but where these correlations are assumed known.

This paper makes the following contributions. First, we propose a version of the knowledge gradient policy that exploits aggregation structure and similarity between alternatives, without requiring that we specify an explicit covariance matrix for our belief. Instead, we develop a belief structure based on the weighted estimates given in (1). We estimate the weights using a Bayesian model adapted from frequentist estimates proposed by George et al. (2008). In addition to eliminating the difficulty of specifying an a priori covariance matrix, this avoids the computational challenge of working with large covariance matrices. Second, we show that a learning policy based on this method is optimal in the limit, that is, eventually it always discovers the best alternative. Our method requires that a family of aggregation functions be provided, but otherwise does not make any specific assumptions about the structure of the function or set of alternatives.

The remainder of this paper is structured as follows. In Section 2 we give a brief overview of the relevant literature. In Section 3, we present our model, the aggregation techniques we use, and the Bayesian updating approach. We present our measurement policy in Section 4 and a proof of convergence of this policy in Section 5. We present numerical experiments in Section 6 and 7. We close with conclusions, remarks on generalizations, and directions for further research in Section 8.

2. Literature

There is by now a substantial literature on the general problem of finding the maximum of an unknown function where we depend on noisy measurements to guide our search. Spall (2003) provides a thorough review of the literature that traces its roots to stochastic approximation methods first introduced by Robbins and Monro (1951). This literature considers problems with vector-valued decisions, but its techniques require many measurements to find maxima precisely, which is a problem when measurements are expensive.

Our problem originates from the ranking and selection (R&S) literature, which begins with Bechhofer (1954). In the R&S problem, we have a collection of alternatives whose value we can learn through sampling, and from which we would like to select the one with the largest value. This problem has been studied extensively since its origin, with much of this work reviewed by Bechhofer et al. (1995), more recent work reviewed in Kim and Nelson (2006), and research continuing actively today. The R&S problem has also been recently and independently considered within computer science (Even-Dar et al., 2002; Madani et al., 2004; Bubeck et al., 2009b).

There is also a related literature on online learning and multi-armed bandits, in which an algorithm is faced with a collection of noisy options of unknown value, and has the opportunity to engage these options sequentially. In the online learning literature, an algorithm is measured according to the *cumulative* value of the options engaged, while in the problem that we consider an algorithm is measured according to its ability to select the best at the end of experimentation. Rather than value, researchers often consider the regret, which is the loss compared to optimal sequence of decisions in hindsight. Cumulative value or regret is appropriate in settings such as dynamic pricing of a good sold online (learning while doing), while terminal value or regret is appropriate in settings such as optimizing a transportation network in simulation before building it in the real world (learn then do). Strong theoretical bounds on cumulative and average regret have been developed in the online setting (see, e.g., Auer et al., 2002; Flaxman et al., 2005; Abernethy et al., 2008).

General-purpose online-to-batch conversion techniques have been developed, starting with Littlestone (1989), for transforming online-learning methods with bounds on cumulative regret into methods with bounds on terminal regret (for a summary and literature review see Shalev-Shwartz, 2007, Appendix B). While these techniques are easy to apply and immediately produce methods with theoretical bounds on the rate at which terminal regret converges to zero, methods created in this way may not have the best achievable bounds on terminal regret: Bubeck et al. (2009b) shows that improving the upper bound on the cumulative regret of an online learning method causes a corresponding lower bound on the terminal regret to get worse. This is indicative of a larger difference between what is required in the two types of problems. Furthermore, as as example of the difference between cumulative and terminal performance, Bubeck et al. (2009b) notes that with finitely many unrelated arms, achieving optimal cumulative regret requires sampling suboptimal arms no more than a logarithmic number of times, while achieving optimal terminal regret requires sampling every arm a linear number of times.

Despite the difference between cumulative and terminal value, a number of methods have been developed that are often applied to both online learning and R&S problems in practice, as well as to more complex problems in reinforcement learning and Markov decision processes. These heuristics include Boltzmann exploration, interval estimation, upper confidence bound policies, and hybrid exploration-exploitation policies such as epsilon-greedy. See Powell and Frazier (2008) for

a review of these. Other policies include the Explicit Explore or Exploit (E^3) algorithm of Kearns and Singh (2002) and R-MAX of Brafman and Tennenholtz (2003).

Researchers from the online learning and multi-armed bandit communities have also directly considered R&S and other related problems in which one is concerned with terminal rather than cumulative value (Even-Dar et al., 2002, 2003; Madani et al., 2004; Mnih et al., 2008; Bubeck et al., 2009b). Most work that directly considers terminal value assumes no a-priori relationship between alternatives. One exception is Srinivas et al. (2010), which considers a problem with a Gaussian process prior on the alternatives, and uses a standard online-to-batch conversion to obtain bounds on terminal regret. We are aware of no work in the online learning community, however, whether considering cumulative value or terminal value, that considers the type of hierarchical aggregation structures that we consider here. A number of researchers have considered other types of dependence between alternatives, such as online convex and linear optimization (Flaxman et al., 2005; Kleinberg, 2005; Abernethy et al., 2008; Bartlett et al., 2008), general metric spaces with a Lipschitz or locally-Lipschitz condition (Kleinberg et al., 2008; Bubeck et al., 2009a), and Gaussian process priors (Grünewälder et al., 2010; Srinivas et al., 2010).

A related line of research has focused on finding the alternative which, if measured, will have the greatest impact on the final solution. This idea was originally introduced in Mockus (1975) for a one-dimensional continuous domain with a Wiener process prior, and in Gupta and Miescke (1996) in the context of the independent normal R&S problem as also considered in this paper. The latter policy was further analyzed in Frazier et al. (2008) under the name knowledge-gradient (KG) policy, where it was shown that the policy is myopically optimal (by construction) and asymptotically optimal. An extension of the KG policy when the variance is unknown is presented in Chick et al. (2010) under the name $\mathcal{LL}_1$, referring to the one-step linear loss, an alternative name when we are minimizing expected opportunity cost. A closely related idea is given in Chick and Inoue (2001) where samples are allocated to maximize an approximation to the expected value of information. Related search methods have also been developed within the simulation-optimization community, which faces the problem of determining the best of a set of parameters, where evaluating a set of parameters involves running what is often an expensive simulation. One class of methods evolved under the name optimal computing budget allocation (OCBA) (Chen et al., 1996; He et al., 2007).

The work in ranking and selection using ideas of expected incremental value is similar to work on Bayesian global optimization of continuous functions. In Bayesian global optimization, one would place a Bayesian prior belief on the unknown function θ. Generally the assumption is that unknown function θ is a realization from a Gaussian process. Wiener process priors, a special case of the Gaussian process prior, were common in early work on Bayesian global optimization, being used by techniques introduced in Kushner (1964) and Mockus (1975). Surveys of Bayesian global optimization may be found in Sasena (2002); Lizotte (2008) and Brochu et al. (2009).

While algorithms for Bayesian global optimization usually assume noise-free function evaluations (e.g., the EGO algorithm of Jones et al., 1998), some algorithms allow measurement noise (Huang et al., 2006; Frazier et al., 2009; Villemonteix et al., 2009). We compare the performance of HKG against two of these: Sequential Kriging Optimization (SKO) from Huang et al. (2006) and the knowledge-gradient policy for correlated normal beliefs (KGCB) from Frazier et al. (2009). The latter policy is an extension of the knowledge-gradient algorithm in the presence of correlated beliefs, where measuring one alternative updates our belief about other alternatives. This method was shown to significantly outperform methods which ignore this covariance structure, but the algorithm requires the covariance matrix to be known. The policies SKO and KGCB are further explained in

Section 6. Like the consistency results that we provide for HKG, consistency results are known for some algorithms: consistency of EGO is shown in Vazquez and Bect (2010), and lower bounds on the convergence rate of an algorithm called GP-UCB are shown in Srinivas et al. (2010).

An approach that is common in optimization of continuous functions, and which accounts for dependencies, is to fit a continuous function through the observations. In the area of Bayesian global optimization, this is usually done using Gaussian process priors. In other approaches, like the Response Surface Methodology (RSM) (Barton and Meckesheimer, 2006) one normally would fit a linear regression model or polynomials. An exception can be found in Brochu et al. (2009) where an algorithm is presented that uses random forests instead, which is reminiscent of the hierarchical prior that we employ in this paper. When we are dealing with nominal categorical dimensions, fitting a continuous function is less appropriate as we will show in this paper. Moreover, the presence of categorical dimensions might give a good indication for the aggregation function to be used. The inclusion of categorical variables in Bayesian global optimization methods, via both random forests and Gaussian processes, as well as a performance comparison between these two, is addressed in Hutter (2009).

There is a separate literature on aggregation and the use of mixtures of estimates. Aggregation, of course, has a long history as a method of simplifying models (see Rogers et al., 1991). Bertsekas and Castanon (1989) describes adaptive aggregation techniques in the context of dynamic programming, while Bertsekas and Tsitsiklis (1996) provides a good presentation of state aggregation methods used in value iteration. In the machine learning community, there is an extensive literature on the use of weighted mixtures of estimates, which is the approach that we use. We refer the reader to LeBlanc and Tibshirani (1996); Yang (2001) and Hastie et al. (2001). In our work, we use a particular weighting scheme proposed by George et al. (2008) due to its ability to easily handle state dependent weights, which typically involves estimation of many thousands of weights since we have a weight for each alternative at each level of aggregation.

3. Model

We consider a finite set $\mathcal{X}$ of distinct alternatives where each alternative $x \in \mathcal{X}$ might be a multi-dimensional vector $x = (x_1, \ldots, x_D)$. Each alternative $x \in \mathcal{X}$ is characterized by an independent normal sampling distribution with unknown mean θ_x and known variance $\lambda_x > 0$. We use M to denote the number of alternatives $|\mathcal{X}|$ and use θ to denote the column vector consisting of all θ_x, $x \in \mathcal{X}$.

Consider a sequence of N sampling decisions, $x^0, x^1, \ldots, x^{N-1}$. The sampling decision x^n selects an alternative to sample at time n from the set $\mathcal{X}$. The sampling error $\varepsilon_x^{n+1} \sim \mathcal{N}(0, \lambda_x)$ is independent conditioned on $x^n = x$, and the resulting sample observation is $\hat{y}_x^{n+1} = \theta_x + \varepsilon_x^{n+1}$. Conditioned on θ and $x^n = x$, the sample has conditional distribution

$$\hat{y}_x^{n+1} \sim \mathcal{N}(\theta_x, \lambda_x).$$

Because decisions are made sequentially, x^n is only allowed to depend on the outcomes of the sampling decisions $x^0, x^1, \ldots, x^{n-1}$. In the remainder of this paper, a random variable indexed by n means it is measurable with respect to $\mathcal{F}^n$ which is the sigma-algebra generated by $x^0, \hat{y}^1_{x^0}, x^1, \ldots, x^{n-1}, \hat{y}^n_{x^{n-1}}$.

In this paper, we derive a method based on Bayesian principles which offers a way of formalizing a priori beliefs and of combining them with the available observations to perform statistical

inference. In this Bayesian approach we begin with a prior distribution on the unknown values θ_x, $x \in \mathcal{X}$, and then use Bayes' rule to recursively to derive the posterior distribution at time $n+1$ from the posterior at time n and the observed data. Let μ^n be our estimate of θ after n measurements. This estimate will either be the Bayes estimate, which is the posterior mean $\mathbb{E}[\theta \mid \mathcal{F}^n]$, or an approximation to this posterior mean as we will use later on. Later, in Sections 3.1 and 3.2, we describe the specific prior and posterior that we use in greater detail. Under most sampling models and prior distributions, including the one we treat here, we may intuitively understand the learning that occurs from sampling as progressive concentration of the posterior distribution on θ, and as the tendency of μ^n, the mean of this posterior distribution, to move toward θ as n increases.

After taking N measurements, we make an implementation decision, which we assume is given by the alternative x^N that has the highest expected reward under the posterior, that is, $x^N \in \arg\max_{x\in\mathcal{X}} \mu_x^N$. Although we could consider policies making implementation decisions in other ways, this implementation decision is optimal when μ^N is the exact posterior mean and when performance is evaluated by the expected value under the prior of the true value of the implemented alternative. Our goal is to choose a sampling policy that maximizes the expected value of the implementation decision x^N. Therefore we define Π to be the set of sampling policies that satisfies the requirement $x^n \in \mathcal{F}^n$ and introduce $\pi \in \Pi$ as a policy that produces a sequence of decisions $\left(x^0,\ldots,x^{N-1}\right)$. We further write $\mathbb{E}^\pi$ to indicate the expectation with respect to the prior over both the noisy outcomes and the truth θ when the sampling policy is fixed to π. Our objective function can now be written as

$$\sup_{\pi\in\Pi} \mathbb{E}^\pi \left[\max_{x\in\mathcal{X}} \mathbb{E}[\theta_x \mid \mathcal{F}^N] \right].$$

If μ^N is the exact posterior mean, rather than an approximation, this can be written as

$$\sup_{\pi\in\Pi} \mathbb{E}^\pi \left[\max_{x\in\mathcal{X}} \mu_x^N \right].$$

As an aid to the reader, the notation defined throughout the next subsections is summarized in Table 1.

3.1 Model Specification

In this section we describe our statistical model, beginning first by describing the aggregation structure upon which it relies, and then describing our Bayesian prior on the sampling means θ_x. Later, in Section 3.2, we describe the Bayesian inference procedure. Throughout these sections we make the following assumptions: (i) we assume independent beliefs across different levels of aggregation and (ii) we have two quantities which we assume are fixed parameters of our model whereas we estimate them using the empirical Bayes approach. Even though these are serious approximations, we show that posterior inference from the prior results in the same estimators as presented in George et al. (2008) derived using frequestist methods.

Aggregation is performed using a set of aggregation functions $G^g : \mathcal{X} \to \mathcal{X}^g$, where $\mathcal{X}^g$ represents the g^{th} level of aggregation of the original set $\mathcal{X}$. We denote the set of all aggregation levels by $\mathcal{G} = \{0,1,\ldots,G\}$, with $g = 0$ being the lowest aggregation level (which might be the finest discretization of a continuous set of alternatives), $g = G$ being the highest aggregation level, and $G = |\mathcal{G}| - 1$.

The aggregation functions G^g are typically problem specific and involve a certain amount of domain knowledge, but it is possible to define generic forms of aggregation. For example, numeric

Variable	Description
G	highest aggregation level
$G^g(x)$	aggregated alternative of alternative x at level g
$\mathcal{G}$	set of all aggregation levels
$\mathcal{G}(x,x')$	set of aggregation levels that alternatives x and x' have in common
$\mathcal{X}$	set of alternatives
$\mathcal{X}^g$	set of aggregated alternatives $G^g(x)$ at the g^{th} aggregation level
$\mathcal{X}^g(x)$	set of alternatives sharing aggregated alternative $G^g(x)$ at aggregation level g
N	maximum number of measurements
M	number of alternatives, that is, $M = \|\mathcal{X}\|$
θ_x	unknown true sampling mean of alternative x
θ_x^g	unknown true sampling mean of aggregated alternative $G^g(x)$
λ_x	measurement variance of alternative x
x^n	n^{th} measurement decision
$\hat{y}_x^n$	n^{th} sample observation of alternative x
ε_x^n	measurement error of the sample observation $\hat{y}_x^n$
μ_x^n	estimate of θ_x after n measurements
$\mu_x^{g,n}$	estimate of aggregated alternative $G^g(x)$ on aggregation level g after n measurements
$w_x^{g,n}$	contribution (weight) of the aggregate estimate $\mu_x^{g,n}$ to the overall estimate μ_x^n of θ_x
$m_x^{g,n}$	number of measurements from the aggregated alternative $G^g(x)$
β_x^n	precision of μ_x^n, with $\beta_x^n = 1/(\sigma_x^n)^2$,
$\beta_x^{g,n}$	precision of $\mu_x^{g,n}$, with $\beta_x^{g,n} = 1/(\sigma_x^{g,n})^2$
$\beta_x^{g,n,\varepsilon}$	measurement precision from alternatives $x' \in \mathcal{X}^g(x)$, with $\beta_x^{g,n,\varepsilon} = 1/(\sigma_x^{g,n,\varepsilon})^2$
$\delta_x^{g,n}$	estimate of the aggregation bias
$\tilde{g}_x^n$	lowest level g for which $m_x^{g,n} > 0$.
$\nu_x^{g,n}$	variance of $\theta_x^g - \theta_x$
$\underline{\delta}$	lower bound on $\delta_x^{g,n}$

Table 1: Notation used in this paper.

data can be defined over a range, allowing us to define a series of aggregations which divide this range by a factor of two at each additional level of aggregation. For vector valued data, we can aggregate by simply ignoring dimensions, although it helps if we are told in advance which dimensions are likely to be the most important.

Using aggregation, we create a sequence of sets $\{\mathcal{X}^g, g = 0,1,\ldots,G\}$, where each set has fewer alternatives than the previous set, and where $\mathcal{X}^0$ equals the original set $\mathcal{X}$. We introduce the following notation and illustrate its value using the example of Figure 1:

$\mathcal{G}(x,x')$ Set of all aggregation levels that the alternatives x and x' have in common, with $\mathcal{G}(x,x') \subseteq \mathcal{G}$. In the example we have $\mathcal{G}(2,3) = \{1,2\}$.

$\mathcal{X}^g(x)$ Set of all alternatives that share the same aggregated alternative $G^g(x)$ at the g^{th} aggregation level, with $\mathcal{X}^g(x) \subseteq \mathcal{X}$. In the example we have $\mathcal{X}^1(4) = \{4,5,6\}$.

$g=2$	13								
$g=1$	10			11			12		
$g=0$	1	2	3	4	5	6	7	8	9

Figure 1: Example with nine alternatives and three aggregation levels.

Given this aggregation structure, we now define our Bayesian model. Define latent variables θ_x^g, where $g \in \mathcal{G}$ and $x \in \mathcal{X}$. These variables satisfy $\theta_x^g = \theta_{x'}^g$ when $G^g(x) = G^g(x')$. Also, $\theta_x^0 = \theta_x$ for all $x \in \mathcal{X}$. We have a belief about these θ_x^g, and the posterior mean of the belief about θ_x^g is $\mu_x^{g,n}$. We see that, roughly speaking, θ_x^g is the best estimate of θ_x that we can make from aggregation level g, given perfect knowledge of this aggregation level, and that $\mu_x^{g,n}$ may be understood to be an estimator of the value of θ_x^g for a particular alternative x at a particular aggregation level g.

We begin with a normal prior on θ_x that is independent across different values of x, given by

$$\theta_x \sim \mathcal{N}(\mu_x^0, (\sigma_x^0)^2).$$

The way in which θ_x^g relates to θ_x is formalized by the probabilistic model

$$\theta_x^g \sim \mathcal{N}(\theta_x, \nu_x^g),$$

where ν_x^g is the variance of $\theta_x^g - \theta_x$ under our prior belief.

The values $\theta_x^g - \theta_x$ are independent across different values of g, and between values of x that differ at aggregation level g, that is, that have different values of $\mathcal{G}^g(x)$. The value ν_x^g is currently a fixed parameter of the model. In practice this parameter is unknown, and while we could place a prior on it (e.g., inverse gamma), we later employ an empirical Bayes approach instead, first estimating it from data and then using the estimated value as if it were given a priori.

When we measure alternative $x^n = x$ at time n, we observe a value $\hat{y}_x^{n+1}$. In reality, this observation has distribution $\mathcal{N}(\theta_x, \lambda_x)$. But in our model, we make the following approximation. We suppose that we observe a value $\hat{y}_x^{g,n+1}$ for each aggregation level $g \in \mathcal{G}$. These values are independent and satisfy

$$\hat{y}_x^{g,n+1} \sim \mathcal{N}(\theta_x^g, 1/\beta_x^{g,n,\varepsilon}),$$

where again $\beta_x^{g,n,\varepsilon}$ is, for the moment, a fixed known parameter, but later will be estimated from data and used as if it were known a priori. In practice we set $\hat{y}_x^{g,n+1} = \hat{y}_x^{n+1}$. It is only a modeling assumption that breaks this equality and assumes independence in its place. This approximation allows us to recover the estimators derived using other techniques in George et al. (2008).

This probabilistic model for $\hat{y}_x^{g,n+1}$ in terms of θ_x^g induces a posterior on θ_x^g, whose calculation is discussed in the next section. This model is summarized in Figure 2.

3.2 Bayesian Inference

We now derive expressions for the posterior belief on the quantities of interest within the model. We begin by deriving an expression for the posterior belief on θ_x^g for a given g.

We define $\mu_x^{g,n}$, $(\sigma_x^{g,n})^2$, and $\beta_x^{g,n} = (\sigma_x^{g,n})^{-2}$ to be the mean, variance, and precision of the belief that we would have about θ_x^g if we had a noninformative prior on θ_x^g and then observed $\hat{y}_{x^{m-1}}^{g,m}$ for *only* those $m < n$ satisfying $G^g(x^m) = G^g(x)$ and *only* for the given value of g. These are the observations from level g pertinent to alternative x. The quantities $\mu_x^{g,n}$ and $\beta_x^{g,n}$ can be obtained recursively

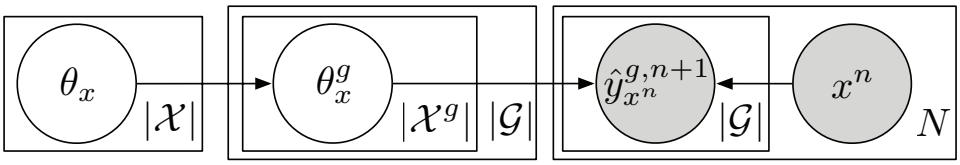


Figure 2: Probabilistic graphical model used by HKG. The dependence of x^n upon the past induced because HKG chooses its measurements adaptively is not pictured.

by considering two cases. When $G^g(x^n) \neq G^g(x)$, we let $\mu_x^{g,n+1} = \mu_x^{g,n}$ and $\beta_x^{g,n+1} = \beta_x^{g,n}$. When $G^g(x^n) = G^g(x)$ we let

$$\mu_x^{g,n+1} = \left[\beta_x^{g,n}\mu_x^{g,n} + \beta_x^{g,n,\varepsilon}\hat{y}_x^{n+1}\right]/\beta_x^{g,n+1}, \tag{2}$$

$$\beta_x^{g,n+1} = \beta_x^{g,n} + \beta_x^{g,n,\varepsilon}, \tag{3}$$

where $\beta_x^{g,0} = 0$ and $\mu_x^{g,0} = 0$.

Using these quantities, we may obtain an expression for the posterior belief on θ_x. We define μ_x^n, $(\sigma_x^n)^2$ and $\beta_x^n = (\sigma_x^n)^{-2}$ to be the mean, variance, and precision of this posterior belief. By Proposition 4 (Appendix B), the posterior mean and precision are

$$\mu_x^n = \frac{1}{\beta_x^n}\left[\beta_x^0\mu_x^0 + \sum_{g\in\mathcal{G}}\left((\sigma_x^{g,n})^2 + \nu_x^g\right)^{-1}\mu_x^{g,n}\right], \tag{4}$$

$$\beta_x^n = \beta_x^0 + \sum_{g\in\mathcal{G}}\left((\sigma_x^{g,n})^2 + \nu_x^g\right)^{-1}. \tag{5}$$

We generally work with a noninformative prior on θ_x in which $\beta_x^0 = 0$. In this case, the posterior variance is given by

$$(\sigma_x^n)^2 = \left(\sum_{g\in\mathcal{G}}\left((\sigma_x^{g,n})^2 + \nu_x^g\right)^{-1}\right)^{-1}, \tag{6}$$

and the posterior mean μ_x^n is given by the weighted linear combination

$$\mu_x^n = \sum_{g\in\mathcal{G}} w_x^{g,n}\mu_x^{g,n}, \tag{7}$$

where the weights $w_x^{g,n}$ are

$$w_x^{g,n} = \left((\sigma_x^{g,n})^2 + \nu_x^g\right)^{-1}\left(\sum_{g'\in\mathcal{G}}\left(\left(\sigma_x^{g',n}\right)^2 + \nu_x^{g'}\right)^{-1}\right)^{-1}. \tag{8}$$

Now, we assumed that we knew ν_x^g and $\beta_x^{g,n,\varepsilon}$ as part of our model, while in practice we do not. We follow the empirical Bayes approach, and estimate these quantities, and then plug in the estimates as if we knew these values a priori. The resulting estimator μ_x^n of θ_x will be identical to the estimator of θ_x derived using frequentist techniques in George et al. (2008).

First, we estimate ν_x^g. Our estimate will be $(\delta_x^{g,n})^2$, where $\delta_x^{g,n}$ is an estimate of the aggregation bias that we define here. At the unaggregated level ($g=0$), the aggregation bias is clearly 0, so we set $\delta_x^{0,n}=0$. If we have measured alternative x and $g>0$, then we set $\delta_x^{g,n}=\max(|\mu_x^{g,n}-\mu_x^{0,n}|,\underline{\delta})$, where $\underline{\delta}\geq 0$ is a constant parameter of the inference method. When $\underline{\delta}>0$, estimates of the aggregation bias are prevented from falling below some minimum threshold, which prevents the algorithm from placing too much weight on a frequently measured aggregate level when estimating the value of an infrequently measured disaggregate level. The convergence proof assumes $\underline{\delta}>0$, although in practice we find that the algorithm works well even when $\underline{\delta}=0$.

To generalize this estimate to include situations when we have not measured alternative x, we introduce a base level $\tilde{g}_x^n$ for each alternative x, being the lowest level g for which $m_x^{g,n}>0$. We then define $\delta_x^{g,n}$ as

$$\delta_x^{g,n}=\begin{cases}0 & \text{if } g=0 \text{ or } g<\tilde{g}_x^n,\\ \max(|\mu_x^{\tilde{g}_x^n}-\mu_x^{g,n}|,\underline{\delta}) & \text{if } g>0 \text{ and } g\geq\tilde{g}_x^n.\end{cases} \tag{9}$$

In addition, we set $w_x^{g,n}=0$ for all $g<\tilde{g}_x^n$.

Second, we estimate $\beta_x^{g,n,\varepsilon}$ using $\beta_x^{g,n,\varepsilon}=\left(\sigma_x^{g,n,\varepsilon}\right)^{-2}$ where $(\sigma_x^{g,n,\varepsilon})^2$ is the group variance (also called the population variance). The group variance $(\sigma_x^{0,n,\varepsilon})^2$ at the disaggregate ($g=0$) level equals λ_x, and we may use analysis of variance (see, e.g., Snijders and Bosker, 1999) to compute the group variance at $g>0$. The group variance over a number of subgroups equals the variance within each subgroup plus the variance between the subgroups. The variance within each subgroup is a weighted average of the variance $\lambda_{x'}$ of measurements of each alternative $x'\in\mathcal{X}^g(x)$. The variance between subgroups is given by the sum of squared deviations of the disaggregate estimates and the aggregate estimates of each alternative. The sum of these variances gives the group variance as

$$(\sigma_x^{g,n,\varepsilon})^2=\frac{1}{m_x^{g,n}}\left(\sum_{x'\in\mathcal{X}^g(x)}m_{x'}^{0,n}\lambda_{x'}+\sum_{x'\in\mathcal{X}^g(x)}m_{x'}^{0,n}\left(\mu_{x'}^{0,n}-\mu_x^{g,n}\right)^2\right),$$

where $m_x^{g,n}$ is the number of measurements from the aggregated alternative $G^g(x)$ at the g^{th} aggregation level, that is, the total number of measurements from alternatives in the set $\mathcal{X}^g(x)$, after n measurements. For $g=0$ we have $\left(\sigma_x^{g,n,\varepsilon}\right)^2=\lambda_x$.

In the computation of $\left(\sigma_x^{g,n,\varepsilon}\right)^2$, the numbers $m_{x'}^{0,n}$ can be regarded as weights: the sum of the bias and measurement variance of the alternative we measured the most contributes the most to the group variance $\left(\sigma_x^{g,n,\varepsilon}\right)^2$. This is because observations of this alternative also have the biggest impact on the aggregate estimate $\mu_x^{g,n}$. The problem, however, is that we are going to use the group variances $\left(\sigma_x^{g,n,\varepsilon}\right)^2$ to get an idea about the range of possible values of $\hat{y}_{x'}^{n+1}$ for all $x'\in\mathcal{X}^g(x)$. By including the number of measurements $m_{x'}^{0,n}$, this estimate of the range will heavily depend on the measurement policy. We propose to put equal weight on each alternative by setting $m_x^{g,n}=|\mathcal{X}^g(x)|$ (so $m_x^{0,n}=1$). The group variance $\left(\sigma_x^{g,n,\varepsilon}\right)^2$ is then given by

$$(\sigma_x^{g,n,\varepsilon})^2=\frac{1}{|\mathcal{X}^g(x)|}\left(\sum_{x'\in\mathcal{X}^g(x)}\lambda_{x'}+\left(\mu_{x'}^{0,n}-\mu_x^{g,n}\right)^2\right). \tag{10}$$

A summary of the Bayesian inference procedure can be found in Appendix A. Given this method of inference, we formally present in the next section the HKG policy for choosing the measurements x^n.

4. Measurement Decision

Our goal is to maximize the expected reward $\mu^N_{x^N}$ of the implementation decision $x^N = \arg\max_{x\in\mathcal{X}} \mu^N_x$. During the sequence of N sampling decisions, $x^0, x^1, \ldots, x^{N-1}$ we gain information that increases our expected final reward $\mu^N_{x^N}$. We may formulate an equivalent problem in which the reward is given in pieces over time, but the total reward given is identical. Then the reward we gain in a single time unit might be regarded as an increase in knowledge. The knowledge-gradient policy maximizes this single period reward. In Section 4.1 we provide a brief general introduction of the knowledge-gradient policy. In Section 4.2 we summarize the knowledge-gradient policy for independent and correlated multivariate normal beliefs as introduced in Frazier et al. (2008, 2009). Then, in Section 4.3, we adapt this policy to our hierarchical setting. We end with an illustration of how the hierarchical knowledge gradient policy chooses its measurements (Section 4.4).

4.1 The Knowledge-Gradient Policy

The knowledge-gradient policy was first introduced in Gupta and Miescke (1996) under the name $(R_1, \ldots, R_1)$, further analyzed in Frazier et al. (2008), and extended in Frazier et al. (2009) to cope with correlated beliefs. The idea works as follows. Let S^n be the knowledge state at time n. In Frazier et al. (2008, 2009) this is given by $S^n = (\mu^n, \Sigma^n)$, where the posterior on θ is $\mathcal{N}(\mu^n, \Sigma^n)$. If we were to stop measuring now, our final expected reward would be $\max_{x\in\mathcal{X}} \mu^n_x$. Now, suppose we were allowed to make one more measurement x^n. Then, the observation $\hat{y}^{n+1}_{x^n}$ would result in an updated knowledge state S^{n+1} which might result in a higher expected reward $\max_{x\in\mathcal{X}} \mu^{n+1}_x$ at the next time unit. The expected incremental value due to measurement x is given by

$$\upsilon^{KG}_x(S^n) = \mathbb{E}\left[\max_{x'\in\mathcal{X}} \mu^{n+1}_{x'} | S^n, x^n = x\right] - \max_{x'\in\mathcal{X}} \mu^n_{x'}. \tag{11}$$

The knowledge-gradient policy π^{KG} chooses its sampling decisions to maximize this expected incremental value. That is, it chooses x^n as

$$x^n = \arg\max_{x\in\mathcal{X}} \upsilon^{KG}_x(S^n).$$

4.2 Knowledge Gradient For Independent And Correlated Beliefs

In Frazier et al. (2008) it is shown that when all components of θ are independent under the prior and under all subsequent posteriors, the knowledge gradient (11) can be written

$$\upsilon^{KG}_x(S^n) = \tilde{\sigma}_x(\Sigma^n, x)\, f\left(\frac{-|\mu^n_x - \max_{x'\neq x} \mu^n_{x'}|}{\tilde{\sigma}_x(\Sigma^n, x)}\right),$$

where $\tilde{\sigma}_x(\Sigma^n, x) = Var\left(\mu^{n+1}_x | S^n, x^n = x\right) = \Sigma^n_{xx} / \sqrt{\lambda_x + \Sigma^n_{xx}}$, with Σ^n_{xx} the variance of our estimate μ^n_x, and where $f(z) = \varphi(z) + z\Phi(z)$ where $\varphi(z)$ and $\Phi(z)$ are, respectively, the normal density and cumulative distribution functions.

In the case of correlated beliefs, an observation $\hat{y}^{n+1}_x$ of alternative x may change our estimate $\mu^n_{x'}$ of alternatives $x' \neq x$. The knowledge gradient (11) can be written as

$$\upsilon^{KG,n}_x(S^n) = \mathbb{E}\left[\max_{x'\in\mathcal{X}} \mu^n_{x'} + \tilde{\sigma}_{x'}(\Sigma^n, x) Z | S^n, x^n = x\right] - \max_{x'\in\mathcal{X}} \mu^n_{x'}, \tag{12}$$

where Z is a standard normal random variable and $\tilde{\sigma}_{x'}(\Sigma^n, x) = \Sigma^n_{x'x}/\sqrt{\lambda_x + \Sigma^n_{xx}}$ with $\Sigma^n_{x'x}$ the covariance between $\mu^n_{x'}$ and μ^n_x.

Solving (12) involves the computation of the expectation over a piecewise linear convex function, which is given as the maximum of affine functions $\mu^n_{x'} + \tilde{\sigma}_{x'}(\Sigma^n, x)Z$. To do this, Frazier et al. (2009) provides an algorithm (Algorithm 2) which solves $h(a,b) = \mathbb{E}[\max_i a_i + b_i Z] - \max_i a_i$ as a generic function of any vectors a and b. In Frazier et al. (2009), the vectors a and b are given by the elements $\mu^n_{x'}$ and $\tilde{\sigma}_{x'}(\Sigma^n, x)$ for all $x' \in \mathcal{X}$ respectively, and the index i corresponds to a particular x'. The algorithm works as follows. First it sorts the sequence of pairs (a_i, b_i) such that the b_i are in non-decreasing order and ties in b are broken by removing the pair (a_i, b_i) when $b_i = b_{i+1}$ and $a_i \leq a_{i+1}$. Next, all pairs (a_i, b_i) that are dominated by the other pairs, that is, $a_i + b_i Z \leq \max_{j \neq i} a_j + b_j Z$ for all values of Z, are removed. Throughout the paper, we use $\tilde{a}$ and $\tilde{b}$ to denote the vectors that result from sorting a and b by b_i followed by the dropping of the unnecessary elements, producing a smaller $\tilde{M}$. The knowledge gradient can now be computed using

$$\upsilon^{KG}_x = \sum_{i=1,\ldots,\tilde{M}} (\tilde{b}_{i+1} - \tilde{b}_i) f\left(-\left|\frac{\tilde{a}_i - \tilde{a}_{i+1}}{\tilde{b}_{i+1} - \tilde{b}_i}\right|\right).$$

Note that the knowledge gradient algorithm for correlated beliefs requires that the covariance matrix Σ^0 be provided as an input. These correlations are typically attributed to physical relationships among the alternatives.

4.3 Hierarchical Knowledge Gradient

We start by generalizing the definition (11) of the knowledge-gradient in the following way

$$\upsilon^{KG}_x(S^n) = \mathbb{E}\left[\max_{x' \in \mathcal{X}} \mu^{n+1}_{x'} | S^n, x^n = x\right] - \max_{x' \in \mathcal{X}} \mathbb{E}\left[\mu^{n+1}_{x'} | S^n, x^n = x\right], \tag{13}$$

where the knowledge state is given by $S^n = \left\{\mu^{g,n}_x, \beta^{g,n}_x : x \in \mathcal{X}, g \in \mathcal{G}\right\}$.

When using the Bayesian updating equations from the original knowledge-gradient policy, the estimates μ^n_x form a martingale, in which case the conditional expectation of $\mu^{n+1}_{x'}$ given S^n is $\mu^n_{x'}$, and (13) is equivalent to the original definition (11). Because of approximations used in the updating equations derived in Section 3, μ^n_x is not a martingale in our case, and the term subtracted in (13) ensures the non-negativity of the KG factor.

Before working out the knowledge gradient (13), we first focus on the aggregate estimate $\mu^{g,n+1}_x$. We rewrite the updating Equation (2) as

$$\begin{aligned}
\mu^{g,n+1}_x &= \left[\beta^{g,n}_x \mu^{g,n}_x + \beta^{g,n,\varepsilon}_x \hat{y}^{n+1}_x\right] / \beta^{g,n+1}_x \\
&= \mu^{g,n}_x + \frac{\beta^{g,n,\varepsilon}_x}{\beta^{g,n}_x + \beta^{g,n,\varepsilon}_x}\left(\hat{y}^{n+1}_x - \mu^{g,n}_x\right) \\
&= \mu^{g,n}_x + \frac{\beta^{g,n,\varepsilon}_x}{\beta^{g,n}_x + \beta^{g,n,\varepsilon}_x}\left(\hat{y}^{n+1}_x - \mu^n_x\right) + \frac{\beta^{g,n,\varepsilon}_x}{\beta^{g,n}_x + \beta^{g,n,\varepsilon}_x}\left(\mu^n_x - \mu^{g,n}_x\right).
\end{aligned}$$

Now, the new estimate is given by the sum of (i) the old estimate, (ii) the deviation of $\hat{y}^{n+1}_x$ from the weighted estimate μ^n_x times the relative increase in precision, and (iii) the deviation of the estimate $\mu^{g,n}_x$ from the weighted estimate μ^n_x times the relative increase in precision. This means

that even if we observe precisely what we expected $\left(\hat{y}_x^{n+1} = \mu_x^n\right)$, the aggregate estimate $\mu_x^{g,n+1}$ still shrinks towards our current weighted estimate μ_x^n. However, the more observations we have, the less shrinking will occur because the precision of our belief on $\mu_x^{g,n}$ will be higher.

The conditional distribution of $\hat{y}_x^{n+1}$ is $\mathcal{N}\left(\mu_x^n, (\sigma_x^n)^2 + \lambda_x\right)$ where the variance of $\hat{y}_x^{n+1}$ is given by the measurement noise λ_x of the current measurement plus the variance $(\sigma_x^n)^2$ of our belief given by (6). So, $Z = \left(\hat{y}_x^{n+1} - \mu_x^n\right) / \sqrt{(\sigma_x^n)^2 + \lambda_x}$ is a standard normal. Now we can write

$$\mu_x^{g,n+1} = \mu_x^{g,n} + \frac{\beta_x^{g,n,\varepsilon}}{\beta_x^{g,n} + \beta_x^{g,n,\varepsilon}} \left(\mu_x^n - \mu_x^{g,n}\right) + \tilde{\sigma}_x^{g,n} Z, \tag{14}$$

where

$$\tilde{\sigma}_x^{g,n} = \frac{\beta_x^{g,n,\varepsilon} \sqrt{(\sigma_x^n)^2 + \lambda_x}}{\beta_x^{g,n} + \beta_x^{g,n,\varepsilon}}. \tag{15}$$

We are interested in the effect of decision x on the weighted estimates $\left\{\mu_{x'}^{n+1}, \forall x' \in \mathcal{X}\right\}$. The problem here is that the values $\mu_{x'}^n$ for all alternatives $x' \in \mathcal{X}$ are updated whenever they share at least one aggregation level with alternative x, which is to say for all x' for which $\mathcal{G}(x', x)$ is not empty. To cope with this, we break our expression (7) for the weighted estimate $\mu_{x'}^{n+1}$ into two parts

$$\mu_{x'}^{n+1} = \sum_{g \notin \mathcal{G}(x',x)} w_{x'}^{g,n+1} \mu_{x'}^{g,n+1} + \sum_{g \in \mathcal{G}(x',x)} w_{x'}^{g,n+1} \mu_x^{g,n+1}.$$

After substitution of (14) and some rearrangement of terms we get

$$\begin{aligned} \mu_{x'}^{n+1} \quad = \quad & \sum_{g \in \mathcal{G}} w_{x'}^{g,n+1} \mu_{x'}^{g,n} + \sum_{g \in \mathcal{G}(x',x)} w_{x'}^{g,n+1} \frac{\beta_x^{g,n,\varepsilon}}{\beta_x^{g,n} + \beta_x^{g,n,\varepsilon}} \left(\mu_x^n - \mu_x^{g,n}\right) \\ & + Z \sum_{g \in \mathcal{G}(x',x)} w_{x'}^{g,n+1} \tilde{\sigma}_x^{g,n}. \end{aligned} \tag{16}$$

Because the weights $w_{x'}^{g,n+1}$ depend on the unknown observation $\hat{y}_{x'}^{n+1}$, we use an estimate $w_{x'}^{g,n}(x)$ of the updated weights given we are going to sample x. Note that we use the superscript n instead of $n+1$ to denote its $\mathcal{F}^n$ measurability.

To compute $w_{x'}^{g,n}(x)$, we use the updated precision $\beta_x^{g,n+1}$ due to sampling x in the weights (8). However, we use the current biases $\delta_x^{g,n}$ because the updated bias $\delta_x^{g,n+1}$ depends on the $\mu_x^{g,n+1}$ which we aim to estimate. The predictive weights $w_{x'}^{g,n}(x)$ are

$$w_{x'}^{g,n}(x) = \frac{\left(\left(\beta_{x'}^{g,n} + I_{x',x}^g \beta_{x'}^{g,n,\varepsilon}\right)^{-1} + \left(\delta_{x'}^{g,n}\right)^2\right)^{-1}}{\sum_{g' \in \mathcal{G}} \left(\left(\beta_{x'}^{g',n} + I_{x',x}^{g'} \beta_{x'}^{g',n,\varepsilon}\right)^{-1} + \left(\delta_{x'}^{g',n}\right)^2\right)^{-1}}, \tag{17}$$

where

$$I_{x',x}^g = \begin{cases} 1 & \text{if } g \in \mathcal{G}(x', x) \\ 0 & \text{otherwise} \end{cases}.$$

After combining (13) with (16) and (17), we get the following knowledge gradient

$$\upsilon_x^{KG}(S^n) = \mathbb{E}\left[\max_{x' \in \mathcal{X}} a_{x'}^n(x) + b_{x'}^n(x) Z | S^n\right] - \max_{x' \in \mathcal{X}} a_{x'}^n(x), \tag{18}$$

where

$$a^n_{x'}(x) = \sum_{g\in\mathcal{G}} w^{g,n}_{x'}(x)\mu^{g,n}_{x'} + \sum_{g\in\mathcal{G}(x',x)} w^{g,n}_{x'}(x)\frac{\beta^{g,n,\varepsilon}_x}{\beta^{g,n}_x+\beta^{g,n,\varepsilon}_x}\left(\mu^n_x-\mu^{g,n}_x\right), \tag{19}$$

$$b^n_{x'}(x) = \sum_{g\in\mathcal{G}(x',x)} w^{g,n}_{x'}(x)\tilde{\sigma}^{g,n}_x. \tag{20}$$

Note that these equations for the knowledge gradient are quite different from those presented in Frazier et al. (2008) for the knowledge gradient for independent beliefs. However, it can be shown that without aggregation levels they coincide (if $G=0$, then $a^n_{x'}(x)=\mu^{0,n}_{x'}=\mu^n_{x'}$ and $b^n_{x'}(x)=\tilde{\sigma}^{0,n}_{x'}$).

Following the approach of Frazier et al. (2009), which was briefly described in Section 4.2, we define $a^n(x)$ as the vector $\left\{a^n_{x'}(x),\ \forall x'\in\mathcal{X}\right\}$ and $b^n(x)$ as the vector $\left\{b^n_{x'}(x),\ \forall x'\in\mathcal{X}\right\}$. From this we derive the adjusted vectors $\tilde{a}^n(x)$ and $\tilde{b}^n(x)$. The knowledge gradient (18) can now be computed using

$$\upsilon^{KG,n}_x = \sum_{i=1,\ldots,\tilde{M}-1}\left(\tilde{b}^n_{i+1}(x)-\tilde{b}^n_i(x)\right) f\left(-\left|\frac{\tilde{a}^n_i(x)-\tilde{a}^n_{i+1}(x)}{\tilde{b}^n_{i+1}(x)-\tilde{b}^n_i(x)}\right|\right), \tag{21}$$

where $\tilde{a}^n_i(x)$ and $\tilde{b}^n_i(x)$ follow from (19) and (20), after the sort and merge operation as described in Section 4.2.

The form of (21) is quite similar to that of the expression in Frazier et al. (2009) for the correlated knowledge-gradient policy, and the computational complexities of the resulting policies are the same. Thus, like the correlated knowledge-gradient policy, the complexity of the hierarchical knowledge-gradient policy is $O\left(M^2\log M\right)$. An algorithm outline for the hierarchical knowledge-gradient measurement decision can be found in Appendix A.

4.4 Remarks

Before presenting the convergence proofs and numerical results, we first provide the intuition behind the hierarchical knowledge gradient (HKG) policy. As illustrated in Powell and Frazier (2008), the independent KG policy prefers to measure alternatives with a high mean and/or with a low precision. As an illustration, consider Figure 3, where we use an aggregation structure given by a perfect binary tree (see Section 6.3) with 128 alternatives at the disaggregate level. At aggregation level 5, there are four aggregated alternatives. As a result, the first four measurements are chosen such that we have one observation for each of these alternatives. The fifth measurement will be either in an unexplored region one aggregation level lower (aggregation level 4 consisting of eight aggregated alternatives) or at an already explored region that has a high weighted estimate. In this case, HKG chooses to sample from the unexplored region $48 < x \leq 64$ since it has a high weighted estimate and a low precision. The same holds for the sixth measurements which would be either from one of the three remaining unexplored aggregated alternatives from level 4, or from an already explored alternative with high weighted mean. In this case, HKG chooses to sample from the region $32 < x \leq 40$, which corresponds with an unexplored alternative at the aggregation level 3. The last panel shows the results after 20 measurements. From this we see HKG concentrates its measurements around the optimum and we have a good fit in this area.

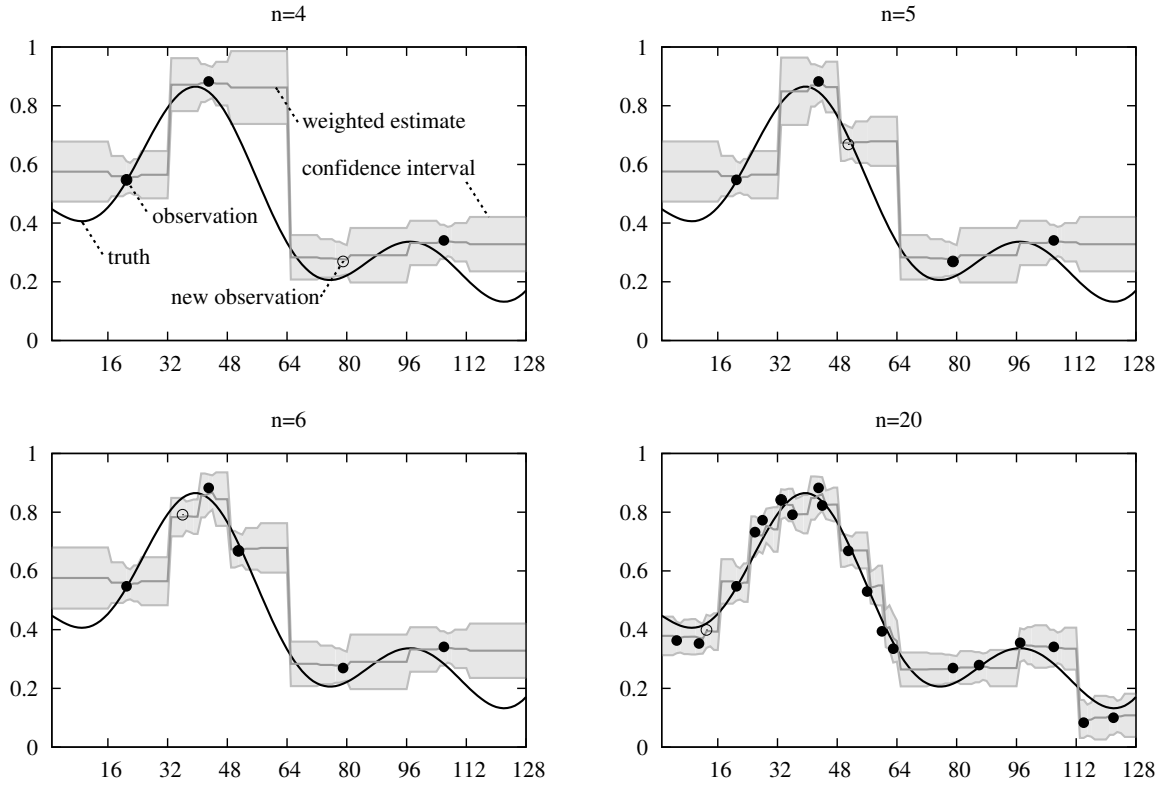


Figure 3: Illustration of the way HKG chooses its measurements.

5. Convergence Results

In this section, we show that the HKG policy measures each alternative infinitely often (Theorem 1). This implies that the HKG policy learns the true values of every alternative as $n \to \infty$ (Corollary 2) and eventually finds a globally optimal alternative (Corollary 3). This final corollary is the main theoretical result of this paper. The proofs of these results depend on lemmas found in Appendix C.

Although the posterior inference and the derivation of the HKG policy assumed that samples from alternative x were normal random variables with known variance λ_x, the theoretical results in this section allow general sampling distributions. We assume only that samples from any fixed alternative x are independent and identically distributed (iid) with finite variance, and that $\underline{\delta} > 0$. These distributions may, of course, differ across x. Thus, even if the true sampling distributions do not meet the assumptions made in deriving the HKG policy, we still enjoy convergence to a globally optimal alternative. We continue to define θ_x to be the true mean of the sampling distribution from alternative x, but the true variance of this distribution can differ from λ_x.

Theorem 1 *Assume that samples from any fixed alternative x are iid with finite variance, and that* $\underline{\delta} > 0$. *Then, the HKG policy measures each alternative infinitely often (i.e.,* $\lim_{n\to\infty} m_x^{0,n} = \infty$ *for each* $x \in \mathcal{X}$*) almost surely.*

Proof Consider what happens as the number of measurements n we make under the HKG policy goes to infinity. Let $\mathcal{X}_\infty$ be the set of all alternatives measured infinitely often under our HKG policy,

and note that this is a random set. Suppose for contradiction that $\mathcal{X}_\infty \neq \mathcal{X}$ with positive probability, that is, there is an alternative that we measure only a finite number of times. Let N_1 be the last time we measure an alternative outside of $\mathcal{X}_\infty$. We compare the KG values $\upsilon_x^{KG,n}$ of those alternatives within $\mathcal{X}_\infty$ to those outside $\mathcal{X}_\infty$.

Let $x \in \mathcal{X}_\infty$. We show that $\lim_{n\to\infty} \upsilon_x^{KG,n} = 0$. Since f is an increasing non-negative function, and $\tilde{b}_{i+1}^n(x) - \tilde{b}_i^n(x) \geq 0$ by the assumed ordering of the alternatives, we have the bounds

$$0 \leq \upsilon_x^{KG,n} \leq \sum_{i=1,\ldots,\tilde{M}-1} \left(\tilde{b}_{i+1}^n(x) - \tilde{b}_i^n(x)\right) f(0).$$

Taking limits, $\lim_{n\to\infty} \upsilon_x^{KG,n} = 0$ follows from $\lim_{n\to\infty} \tilde{b}_i^n(x) = 0$ for $i = 1,\ldots,\tilde{M}$, which follows in turn from $\lim_{n\to\infty} b_{x'}^n(x) = 0 \; \forall x' \in \mathcal{X}$ as shown in Lemma 8.

Next, let $x \notin \mathcal{X}_\infty$. We show that $\liminf_{n\to\infty} \upsilon_x^{KG,n} > 0$. Let $U = \sup_{n,i} |a_i^n(x)|$, which is almost surely finite by Lemma 7. Let $x' \in \mathcal{X}_\infty$. At least one such alternative x' must exist since we allocate an infinite number of measurements and $\mathcal{X}$ is finite. Lemma 9 shows

$$\upsilon_x^{KG,n} \geq \frac{1}{2} |b_{x'}^n(x) - b_x^n(x)| \, f\left(\frac{-4U}{|b_{x'}^n(x) - b_x^n(x)|}\right).$$

From Lemma 8, we know that $\liminf_{n\to\infty} b_x^n(x) > 0$ and $\lim_{n\to\infty} b_{x'}^n(x) = 0$. Thus, $b^* = \liminf_{n\to\infty} |b_x^n(x) - b_{x'}^n(x)| > 0$. Taking the limit inferior of the bound on $\upsilon_x^{KG,n}$ and noting the continuity and monotonicity of f, we obtain

$$\liminf_{n\to\infty} \upsilon_x^{KG,n} \geq \frac{1}{2} b^* f\left(\frac{-4U}{b^*}\right) > 0.$$

Finally, since $\lim_{n\to\infty} \upsilon_x^{KG,n} = 0$ for all $x \in \mathcal{X}_\infty$ and $\liminf_{n\to\infty} \upsilon_{x'}^{KG,n} > 0$ for all $x' \notin \mathcal{X}_\infty$, each $x' \notin \mathcal{X}_\infty$ has an $n > N_1$ such that $\upsilon_{x'}^{KG,n} > \upsilon_x^{KG,n} \; \forall x \in \mathcal{X}_\infty$. Hence we choose to measure an alternative outside $\mathcal{X}_\infty$ at a time $n > N_1$. This contradicts the definition of N_1 as the last time we measured outside $\mathcal{X}_\infty$, contradicting the supposition that $\mathcal{X}_\infty \neq \mathcal{X}$. Hence we may conclude that $\mathcal{X}_\infty = \mathcal{X}$, meaning we measure each alternative infinitely often. ■

Corollary 2 *Assume that samples from any fixed alternative x are iid with finite variance, and that* $\underline{\delta} > 0$. *Then, under the HKG policy,* $\lim_{n\to\infty} \mu_x^n = \theta_x$ *almost surely for each* $x \in \mathcal{X}$.

Proof Fix x. We first consider $\mu_x^{0,n}$, which can be written as

$$\mu_x^{0,n} = \frac{\beta_x^{0,0}\mu_x^{0,0} + m_x^{0,n}(\lambda_x)^{-1} y_x^n}{\beta_x^{0,0} + m_x^{0,n}(\lambda_x)^{-1}},$$

where y_x^n is the average of all observations of alternative x by time n. As $n \to \infty$, $m_x^{0,n} \to \infty$ by Theorem 1. Thus, $\lim_{n\to\infty} \mu_x^{0,n} = \lim_{n\to\infty} y_x^n$, which is equal to θ_x almost surely by the law of large numbers.

We now consider the weights $w_x^{g,n}$. For $g \neq 0$, (8) shows

$$w_x^{g,n} \leq \frac{\left((\sigma_x^{g,n})^2 + (\delta_x^{g,n})^2\right)^{-1}}{(\sigma_x^{0,n})^{-2} + \left((\sigma_x^{g,n})^2 + (\delta_x^{g,n})^2\right)^{-1}}.$$

When n is large enough that we have measured at least one alternative in $\mathcal{X}^g(x)$, then $\delta_x^{g,n} \geq \underline{\delta}$, implying $\left((\sigma_x^{g,n})^2 + (\delta_x^{g,n})^2\right)^{-1} \leq \underline{\delta}^{-2}$ and $w_x^{g,n} \leq \underline{\delta}^{-2}/((\sigma_x^{0,n})^{-2} + \underline{\delta}^{-2})$. As $n \to \infty$, $m_x^{0,n} \to \infty$ by Theorem 1 and $(\sigma_x^{0,n})^{-2} = \beta^{0,0} + m_x^{0,n}(\lambda_x)^{-1} \to \infty$. This implies that $\lim_{n\to\infty} w_x^{g,n} = 0$. Also observe that $w_x^{0,n} = 1 - \sum_{g\neq 0} w_x^{g,n}$ implies $\lim_{n\to\infty} w_x^{0,n} = 1$.

These limits for the weights, the almost sure finiteness of $\sup_n |\mu_x^{g,n}|$ for each g from Lemma 7, and the definition (7) of μ_x^n together imply $\lim_{n\to\infty} \mu_x^n = \lim_{n\to\infty} \mu_x^{0,n}$, which equals θ_x as shown above. ■

Finally, Corollary 3 below states that the HKG policy eventually finds a globally optimal alternative. This is the main result of this section. In this result, keep in mind that $\hat{x}^n = \arg\max_x \mu_x^N$ is the alternative one would estimate to be best at time N, given all the measurements collected by HKG. It is this estimate that converges to the globally optimal alternative, and not the HKG measurements themselves.

Corollary 3 *For each n, let $\hat{x}^n \in \arg\max_x \mu_x^n$. Assume that samples from any fixed alternative x are iid with finite variance, and that $\underline{\delta} > 0$. Then, under the HKG policy, there exists an almost surely finite random variable N' such that $\hat{x}^n \in \arg\max_x \theta_x$ for all $n > N'$.*

Proof Let $\theta^* = \max_x \theta_x$ and $\varepsilon = \min\{\theta^* - \theta_x : x \in \mathcal{X}, \theta^* > \theta_x\}$, where $\varepsilon = \infty$ if $\theta_x = \theta^*$ for all x. Corollary 2 states that $\lim_{n\to\infty} \mu_x^n = \theta_x$ almost surely for all x, which implies the existence of an almost surely finite random variable N' with $\max_x |\mu_x^n - \theta_x| < \varepsilon/2$ for all $n > N'$. On the event $\{\varepsilon = \infty\}$ we may take $N' = 0$. Fix $n > N'$, let $x^* \in \arg\max_x \theta_x$, and let $x' \notin \arg\max_x \theta_x$. Then $\mu_{x^*}^n - \mu_{x'}^n = (\theta_{x^*} - \theta_{x'}) + (-\theta_{x^*} + \mu_{x^*}^n) + (\theta_{x'} - \mu_{x'}^n) > \theta_{x^*} - \theta_{x'} - \varepsilon \geq 0$. This implies that $\hat{x}^n \in \arg\max_x \theta_x$. ■

6. Numerical Experiments

To evaluate the hierarchical knowledge-gradient policy, we perform a number of experiments. Our objective is to find the strengths and weaknesses of the HKG policy. To this end, we compare HKG with some well-known competing policies and study the sensitivity of these policies to various problem settings such as the dimensionality and smoothness of the function, and the measurement noise.

6.1 Competing Policies

We compare the Hierarchical Knowledge Gradient (HKG) algorithm against several ranking and selection policies: the Interval Estimation (IE) rule from Kaelbling (1993), the Upper Confidence Bound (UCB) decision rule from Auer et al. (2002), the Independent Knowledge Gradient (IKG) policy from Frazier et al. (2008), Boltzmann exploration (BOLTZ), and pure exploration (EXPL).

In addition, we compare with the Knowledge Gradient policy for correlated beliefs (KGCB) from Frazier et al. (2009) and, from the field of Bayesian global optimization, we select the Sequential Kriging Optimization (SKO) policy from Huang et al. (2006). SKO is an extension of the well known Efficient Global Optimization (EGO) policy (Jones et al., 1998) to the case with noisy measurements.

We also consider an hybrid version of the HKG algorithm (HHKG) in which we only exploit the similarity between alternatives in the updating equations and not in the measurement decision. As a result, this policy uses the measurement decision of IKG and the updating equations of HKG. The possible advantage of this hybrid policy is that it is able to cope with similarity between alternatives without the computational complexity of HKG.

Since several of the policies require choosing one or more parameters, we provide a brief description of the implementation of these policies in Appendix D. For those policies that require it, we perform tuning using all one-dimensional test functions (see Section 6.2). For the Bayesian approaches, we always start with a non-informative prior.

6.2 Test Functions

To evaluate the policies numerically, we use various test functions with the goal of finding the highest point of each function. Measuring the functions is done with normally distributed noise with variance λ. The functions are chosen from commonly used test functions for similar procedures.

6.2.1 ONE-DIMENSIONAL FUNCTIONS

First we test our approach on one-dimensional functions. In this case, the alternatives x simply represent a single value, which we express by i or j. As test functions we use a Gaussian process with zero mean and power exponential covariance function

$$Cov(i,j) = \sigma^2 \exp\left\{-\left(\frac{|i-j|}{(M-1)\rho}\right)^{\eta}\right\},$$

which results in a stationary process with variance σ^2 and a length scale ρ.

Higher values of ρ result in fewer peaks in the domain and higher values of η result in smoother functions. Here we fix $\eta = 2$ and vary $\rho \in 0.05, 0.1, 0.2, 0.5$. The choice of σ^2 determines the vertical scale of the function. Here we fix $\sigma^2 = 0.5$ and we vary the measurement variance λ.

To generate a truth θ_i, we take a random draw from the Gaussian process (see, e.g., Rasmussen and Williams, 2006) evaluated at the discretized points $i = 1, .., 128$. Figure 4 shows one test function for each value of ρ.

Next, we consider non-stationary covariance functions. We choose to use the Gibbs covariance function (Gibbs, 1997) as it has a similar structure to the exponential covariance function but is non-stationary. The Gibbs covariance function is given by

$$Cov(i,j) = \sigma^2 \sqrt{\frac{2l(i)l(j)}{l(i)^2 + l(j)^2}} \exp\left(-\frac{(i-j)^2}{l(i)^2 + l(j)^2}\right),$$

where $l(i)$ is an arbitrary positive function in i. In our experiments we use a horizontally shifted periodic sine curve for $l(i)$,

$$l(i) = 1 + 10\left(1 + \sin\left(2\pi\left(\frac{i}{128} + u\right)\right)\right),$$

where u is a random number from [0,1] that shifts the curve horizontally across the x-axis. The function $l(i)$ is chosen so that, roughly speaking, the resulting function has one full period, that is, one area with relatively low correlations and one area with relatively high correlations. The area

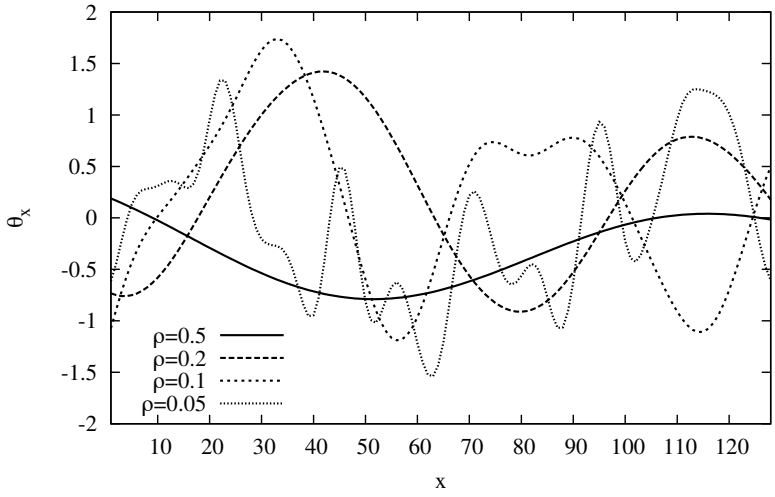


Figure 4: Illustration of one-dimensional test functions.

with low correlations visually resembles the case of having a stationary function with $\rho = 0.05$, whereas the area with high correlations visually resembles the case of having a stationary function with $\rho = 0.5$.

The policies KGCB, SKO and HKG all assume the presence of correlations in function values. To test the robustness of these policies in the absence of any correlation, we consider one last one-dimensional test function. This function has an independent truth generated by $\theta_i = U[0,1], i = 1,...,128$.

6.2.2 TWO-DIMENSIONAL FUNCTIONS

Next, we consider two-dimensional test functions. First, we consider the Six-hump camel back (Branin, 1972) given by

$$f(x) = 4x_1^2 - 2.1x_1^4 + \frac{1}{3}x_1^6 + x_1x_2 - 4x_2^2 + 4x_2^4.$$

Different domains have been proposed for this function. Here we consider the domain $x \in [-1.6, 2.4] \times [-0.8, 1.2]$ as also used in Huang et al. (2006) and Frazier et al. (2009), and a slightly bigger domain $x \in [-2, 3] \times [-1, 1.5]$. The extended part of this domain contains only values far from the optimum. Hence, the extension does not change the value and location of the optimum.

The second function we consider is the Tilted Branin (Huang et al., 2006) given by

$$f(x) = \left(x_2 - \frac{5.1}{4\pi^2}x_1^2 + \frac{5}{\pi}x_1 - 6\right)^2 + 10\left(1 - \frac{1}{8\pi}\right)\cos(x_1) + 10 + \frac{1}{2}x_1,$$

with $x \in [-5, 10] \times [0, 15]$.

The Six-hump camel back and Tilted Branin function are relatively smooth functions in the sense that a Gaussian process can be fitted to the truth relatively well. Obviously, KGCB and SKO benefit from this. To also study more messy functions, we shuffle these functions by placing a 2×2 grid onto the domain and exchange the function values from the lower left quadrant with those from the upper right quadrant.

With the exception of SKO, all policies considered in this paper require problems with a finite number of alternatives. Therefore, we discretize the set of alternatives and use an 32×32 equispaced grid on $\mathbb{R}^2$. We choose this level of discretization because, although our method is theoretically capable of handling any finite number of alternatives, computational issues limit the possible number to the order of thousands. This limit also holds for KGCB, which has the same computational complexity as HKG. For SKO we still use the continuous functions which should give this policy some advantage.

6.2.3 CASE EXAMPLE

To give an idea about the type of practical problems for which HKG can be used, we consider a transportation application (see Simao et al., 2009). Here we must decide where to send a driver described by three attributes: (i) the location to which we are sending him, (ii) his home location (called his domicile) and (iii) to which of six fleets he belongs. The "fleet" is a categorical attribute that describes whether the driver works regionally or nationally and whether he works as a single driver or in a team. The spatial attributes (driver location and domicile) are divided into 100 regions (by the company). However, to reduce computation time, we aggregate these regions into 25 regions. Our problem is to find which of the $25 \times 25 \times 6 = 3750$ is best.

To allow replicability of this experiment, we describe the underlying truth using an adaption of a known function which resembles some of the characteristics of the transportation application. For this purpose we use the Six-hump camel back function, on the smaller domain, as presented earlier. We let x_1 be the location and x_2 be the driver domicile, which are both discretized into 25 pieces to represent regions. To include the dependence on capacity type, we use the following transformation

$$g(x_1, x_2, x_3) = p_1(x_3) - p_2(x_3)(|x_1 - 2x_2|) - f(x_1, x_2),$$

where x_3 denotes the capacity type. We use $p_2(x_3)$ to describe the dependence of capacity type on the distance between the location of the driver and his domicile.

We consider the following capacity types: CAN for Canadian drivers that only serve Canadian loads, WR for western drivers that only serve western loads, US_S for United States (US) solo drivers, US_T for US team drivers, US_IS for US independent contractor solo drivers, and US_IT for US independent contractor team drivers. The parameter values are shown in Table 2. To cope with the fact that some drivers (CAN and WR) cannot travel to certain locations, we set the value to zero for the combinations $\{x_3 = \text{CAN} \wedge x_1 < 1.8\}$ and $\{x_3 = \text{WR} \wedge x_1 > -0.8\}$. The maximum of $g(x_1, x_2, x_3)$ is attained at $g(0, 0, \text{US\_S})$ with value 6.5.

x_3	CAN	WR	US_S	US_T	US_IS	US_IT
$p_1(x_3)$	7.5	7.5	6.5	5.0	2.0	0.0
$p_2(x_3)$	0.5	0.5	2.0	0.0	2.0	0.0

Table 2: Parameter settings.

To provide an indication of the resulting function, we show $\max_{x_3} g(x_1, x_2, x_3)$ in Figure 5. This function has similar properties to the Six-hump camel back, except for the presence of discontinuities due to the capacity types CAN and WR, and a twist at $x_1 = x_2$.

An overview of all test functions can be found in Table 3. Here σ denotes the standard deviation of the function measured over the given discretization.

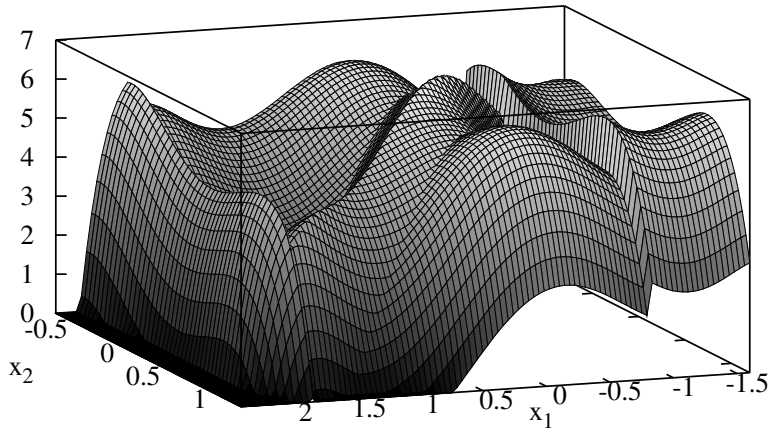


Figure 5: $max_{x_3} g(x_1, x_2, x_3)$.

Type	Function name	σ	Description
One-dimensional	GP1R005	0.32	stationary GP with $\rho = 0.05$
	GP1R01	0.49	stationary GP with $\rho = 0.1$
	GP1R02	0.57	stationary GP with $\rho = 0.2$
	GP1R05	0.67	stationary GP with $\rho = 0.5$
	NSGP	0.71	non-stationary GP
	IT	0.29	independent truth
Two-dimensional	SHCB-DS	2.87	Six-hump camel back on small domain
	SHCB-DL	18.83	Six-hump camel back on large domain
	TBRANIN	51.34	Tilted Branin
	SHCB-DS-SH	2.87	shuffled SHCB-DS
	SHCB-DB-SH	18.83	shuffled SHCB-DL
	TBRANIN-SH	51.34	shuffled TBRANIN
Case example	TA	3.43	transportation application

Table 3: Overview of test functions.

6.3 Experimental Settings

We consider the following experimental factors: the measurement variance λ, the measurement budget N, and for the HKG policy the aggregation structure. Given these factors, together with the nine policies from Section 6.1 and the 15 test functions from Section 6.2, a full factorial design is not an option. Instead, we limit the number of combinations as explained in this section.

As mentioned in the introduction, our interest is primarily in problems where M is larger than the measurement budget N. However, for these problems it would not make sense to compare with the tested versions of IE, UCB and BOLTZ since, in the absence of an informed prior, these methods typically choose one measurement of each of the M alternatives before measuring any alternative a second time. Although we do not do so here, one could consider versions of these policies with informative priors (e.g., the GP-UCB policy of Srinivas et al. (2010), which uses UCB with a Gaussian process prior), which would perform better on problems with M much larger than

N. To obtain meaningful results for the tested versions of IE, UCB and BOLTZ, we start with an experiment with a relatively large measurement budget and relatively large measurement noise. We use all one-dimensional test functions with $N = 500$ and $\sqrt{\lambda} \in \{0.5, 1\}$. We omit the policy HHKG, which will be considered later.

In the remaining experiments we omit the policies IE, UCB, and BOLTZ that use non-informative priors because they would significantly underperform the other policies. This is especially true with the multi-dimensional problems where the number of alternatives after discretization is much bigger then the measurement budget. We start with testing the remaining policies, together with the hybrid policy HHKG, on all one-dimensional test functions using $\sqrt{\lambda} \in \{0.1, 0.5, 1\}$ and $N = 200$. Next, we use the non-stationary function to study (i) the sensitivity of all policies on the value of λ, using $\sqrt{\lambda} \in \{0.1, 0.5, 1, 1.5, 2, 2.5\}$ and (ii) the sensitivity of HKG on the aggregation structure. For the latter, we consider two values for $\sqrt{\lambda}$, namely 0.5 and 1, and five different aggregation structures as presented at the end of this subsection.

For the stationary one-dimensional setting, we generate 10 random functions for each value of ρ. For the non-stationary setting and the random truth setting, we generate 25 random functions each. This gives a total of 90 different functions. We use 50 replications for each experiment and each generated function.

For the multi-dimensional functions we only consider the policies KGCB, SKO, HKG, and HHKG. For the two-dimensional functions we use $N = 200$. For the transportation application we use $N = 500$ and also present the results for intermediate values of n. We set the values for λ by taking into account the standard deviation σ of the functions (see Table 3). For the Six-hump camel back we use $\sqrt{\lambda} \in \{1, 2, 4\}$, for the Tilted Branin we use $\sqrt{\lambda} \in \{2, 4, 8\}$, and for the case example we use $\sqrt{\lambda} \in \{1, 2\}$. For the multi-dimensional functions we use 100 replications.

During the replications we keep track of the opportunity costs, which we define as $OC(n) = (\max_i \theta_i) - \theta_{i^*}$, with $i^* \in \arg\max_x \mu_x^n$, that is, the difference between the true maximum and the value of the best alternative found by the algorithm after n measurements. Our key performance indicator is the mean opportunity costs $\mathbb{E}[OC(n)]$ measured over all replications of one or more experiments. For clarity of exposition, we also group experiments and introduce a set GP1 containing the 40 stationary one-dimensional test functions and a set NS0 containing the 50 non-stationary and independent truth functions. When presenting the $\mathbb{E}[OC(n)]$ in tabular form, we bold and underline the lowest value, and we also bold those values that are not significantly different from the lowest one (using Welch's t test at the 0.05 level).

We end this section with an explanation of the aggregation functions used by HKG. Our default aggregation structure is given by a binary tree, that is, $|\mathcal{X}^g(x)| = 2^g$ for all $x \in \mathcal{X}^g$ and $g \in \mathcal{G}$. As a result, we have 8 $(\ln(128)/\ln(2) + 1)$ aggregation levels for the one-dimensional problems and 6 $(\ln(32)/\ln(2) + 1)$ for the two-dimensional problems. For the experiment with varying aggregation functions, we introduce a variable ω to denote the number of alternatives $G^g(x), g < G$ that should be aggregated in a single alternative $G^{g+1}(x)$ one aggregation level higher. At the end of the domain this might not be possible, for example, if we have an odd number of (aggregated) alternatives. In this case, we use the maximum number possible. We consider the values $\omega \in \{2, 4, 8, 16\}$, where $\omega = 2$ resembles the original situation of using a binary tree. To evaluate the impact of having a difference in the size of aggregated sets, we introduce a fifth aggregation structure where ω alternately takes values 2 and 4.

For the transportation application, we consider five levels of aggregation. At aggregation level 0, we have 25 regions for location and domicile, and 6 capacity types, producing 3750 attribute vectors.

At aggregation level 1, we represent the driver domicile as one of 5 areas. At aggregation level 2, we ignore the driver domicile; at aggregation level 3, we ignore capacity type; and at aggregation level 4, we represent location as one of 5 areas.

An overview of all experiments can be found in Table 4.

Experiment	Number of runs
One-dimensional long	$90 \times 8 \times 2 \times 1 \times 50 = 72{,}000$
One-dimensional normal	$90 \times 6 \times 3 \times 1 \times 50 = 81{,}000$
One-dimensional varying λ	$25 \times 6 \times 6 \times 1 \times 50 = 45{,}000$
One-dimensional varying ω	$25 \times 1 \times 2 \times 5 \times 50 = 12{,}500$
Two-dimensional	$6 \times 3 \times 3 \times 1 \times 100 = 27{,}000$
Transportation application	$2 \times 3 \times 2 \times 1 \times 100 = 6000$

Table 4: Overview of experiments. The number of runs is given by #functions $\times$ #policies $\times$ #λ's $\times$ #ω's $\times$ #replications. The total number of experiments, defined by the number of unique combinations of function, policy, λ, and ω, is 2696.

7. Numerical Results

In this section we present the results of the experiments described in Section 6. We demonstrate that HKG performs best when measured by the average performance across all problems. In particular, it outperforms others on functions for which the use of an aggregation function seems to be a natural choice, but it also performs well on problems for which the other policies are specifically designed. In the following subsections we present the policies, the test functions, and the experimental design.

7.1 One-dimensional Functions

In our first experiment, we focus on the comparison with R&S policies using a relatively large measurement budget. A complete overview of the results, for $n = 500$ and an intermediate value $n = 250$, can be found in Appendix E. To illustrate the sensitivity of the performance of these policies to the number of measurements n, we also provide a graphical illustration in Figure 6. To keep these figures readable, we omit the policies UCB and IKG since their performance is close to that of IE (see Appendix E).

As expected, the R&S policies perform well with many measurements. IE generally performs best, closely followed by UCB. BOLTZ only performs well for few measurements ($n \leq M$) after which it underperforms the other policies with the exception of EXPL, which spends an unnecessary portion of its measurements on less attractive alternatives.

With increasing n, IE eventually outperforms at least one of the advanced policies (KGCB, SKO, and HKG). However, it seems that the number of measurements required for IE to outperform KGCB and HKG increases with increasing measurement variance λ. We further see, from Appendix E, that IE outperforms IKG on most instances. However, keep in mind that we tuned IE using exactly the functions on which we test while IKG does not require any form of tuning. The qualitative change in the performance of IE at $n = 128$ samples is due to the fact that the version of IE against

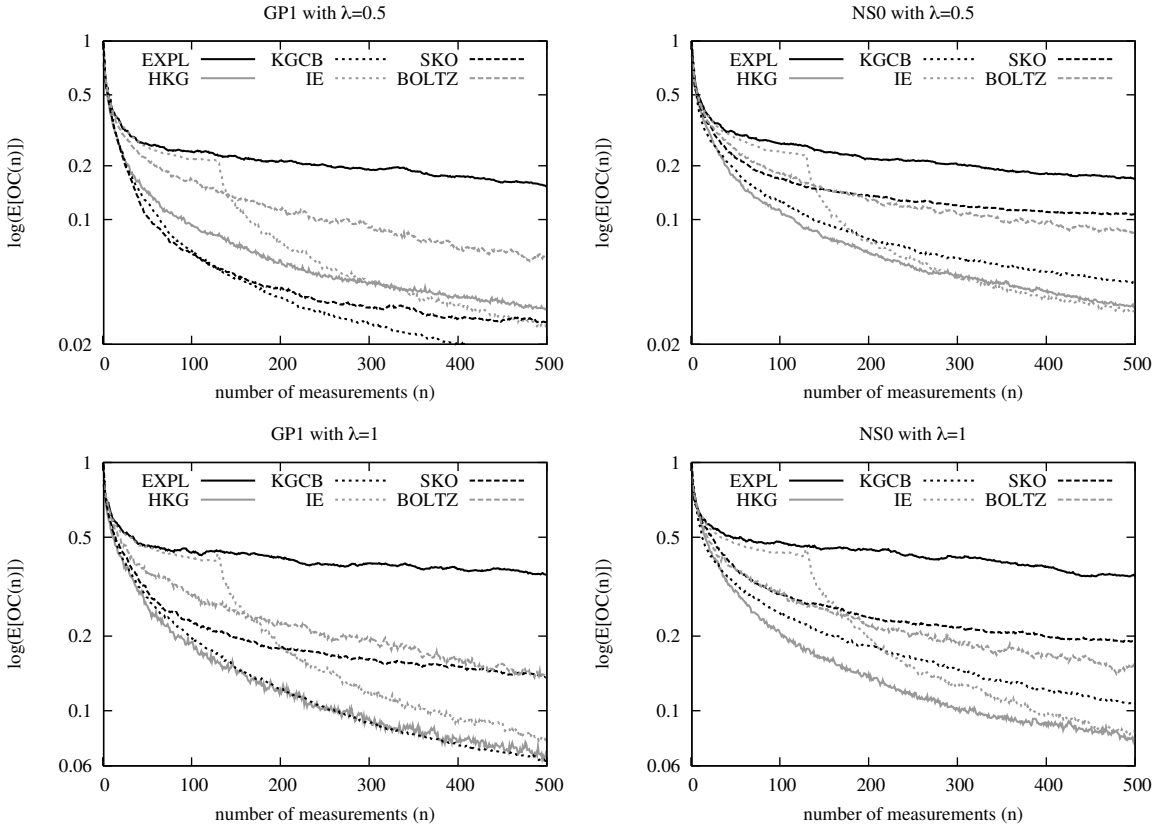


Figure 6: Results for the one-dimensional long experiments.

which we compare uses a non-informative prior, which causes it to measure each alternative exactly once before it can use the IE logic to decide where to allocate future samples.

With respect to the more advanced policies, we see that HKG outperforms the others on the NS0 functions (non-stationary covariance and independent truth) and performs competitively on the stationary GPs in the case of relatively large λ. Obviously, KGCB and SKO are doing well on the latter case since the truths are drawn from a Gaussian process and these policies fit a Gaussian process to the evaluated function values. Apart from the given aggregation function, HKG does not assume any structure and therefore has a slower rate of convergence on these instances. Further, it is remarkable to see that SKO is only competitive on GP1 with $\lambda = 0.5$ but not with $\lambda = 1$. We return to this issue in the next experiment.

For a more detailed comparison between KGCB, SKO and HKG we now focus on smaller measurement budgets. A summary of the results can be found in Table 5. More detailed results in combination with a further analysis can be found in Appendix E. As mentioned before, we bold and underline the lowest value, and we also bold those values that are not significantly different from the lowest one.

On the GP1 functions with $\lambda \leq 0.5$, HKG is outperformed by KGCB and SKO. SKO does particularly well during the early measurements (n=50) after which it is outperformed by KGCB (n=200). On the GP1 functions with $\lambda = 1$, we see HKG becomes more competitive: in almost

Function	$\sqrt{\lambda}$	n	EXPL	IKG	KGCB	SKO	HKG	HHKG
GP1	0.1	50	0.090	0.081	0.010	**0.008**	0.034	0.078
		200	0.051	0.006	**0.002**	0.004	0.008	0.008
	0.5	50	0.265	0.252	0.123	**0.104**	0.141	0.175
		200	0.214	0.075	**0.037**	**0.041**	0.059	0.065
	1	50	0.460	0.441	0.286	0.302	**0.265**	0.305
		200	0.415	0.182	**0.122**	0.181	**0.121**	0.135
NS0	0.1	50	0.111	0.096	0.066	0.093	**0.051**	0.113
		200	0.043	**0.008**	0.017	0.060	**0.009**	0.014
	0.5	50	0.301	0.288	0.189	0.221	**0.170**	0.212
		200	0.219	0.086	0.078	0.136	**0.065**	0.081
	1	50	0.498	0.468	0.323	0.375	**0.306**	0.335
		200	0.446	0.213	0.183	0.238	**0.141**	0.163

Table 5: $\mathbb{E}[OC(n)]$ on the one-dimensional normal experiments.

all cases it outperforms SKO, and with a limited measurement budget (n=50) it also outperforms KGCB.

On the NS0 functions, we see that HKG always outperforms KGCB and SKO with the only exception being the independent truth (IT) function with $\lambda = 1$ and $n = 50$ (see Appendix E). We also see that SKO is always outperformed by KGCB. Especially in the case with low measurement noise ($\lambda = 0.1$) and a large number of measurements ($n = 200$), SKO performs relatively poorly. This is exactly the situation in which one would expect to obtain a good fit, but a fitted Gaussian process prior with zero correlation is of no use. With an increasing number of measurements, we see SKO is even outperformed by EXPL.

In general, HKG seems to be relatively robust in the sense that, whenever it is outperformed by other policies, it still performs well. This claim is also supported by the opportunity costs measured over all functions and values of λ found in Table 6 (note this is not a completely fair comparison since we have slightly more non-stationary functions, and the average opportunity costs over all policies is slightly higher in the non-stationary cases). Even though HKG seems to be quite competitive, HKG seems to have convergence problems in the low noise case ($\lambda = 0.1$). We analyze this issue further in Appendix E. The hybrid policy does not perform well, although it outperforms IKG on most problem instances.

	EXPL	IKG	KGCB	SKO	HKG	HHKG
$\mathbb{E}[OC(50)]$	0.289	0.273	0.169	0.189	**0.163**	0.205
$\mathbb{E}[OC(200)]$	0.232	0.096	0.075	0.114	**0.068**	0.078

Table 6: Aggregate results for the one-dimensional normal experiments.

In the next experiment we vary the measurement variance λ. Figure 7 shows the relative reduction in $\mathbb{E}[OC(50)]$ compared with the performance of EXPL. For clarity of exposition, we omitted the results for $n = 200$ and the performance of IKG. These results confirm our initial conclusions with respect to the measurement variance: increasing λ gives HKG a competitive advantage whereas the opposite holds for SKO. On the GP1R02 functions, HKG is outperformed by SKO and KGCB for $\lambda \leq 0.5$. With $\lambda > 0.5$, the performance of KGCB, HKG, and HHKG is close and they all

outperform SKO. On the NSGP functions, the ordering of policies seem to remain the same for all values of λ, with the exception that with $\lambda \geq 1$, SKO is outperformed by all policies. The difference between KGCB and HKG seems to decline with increasing λ.

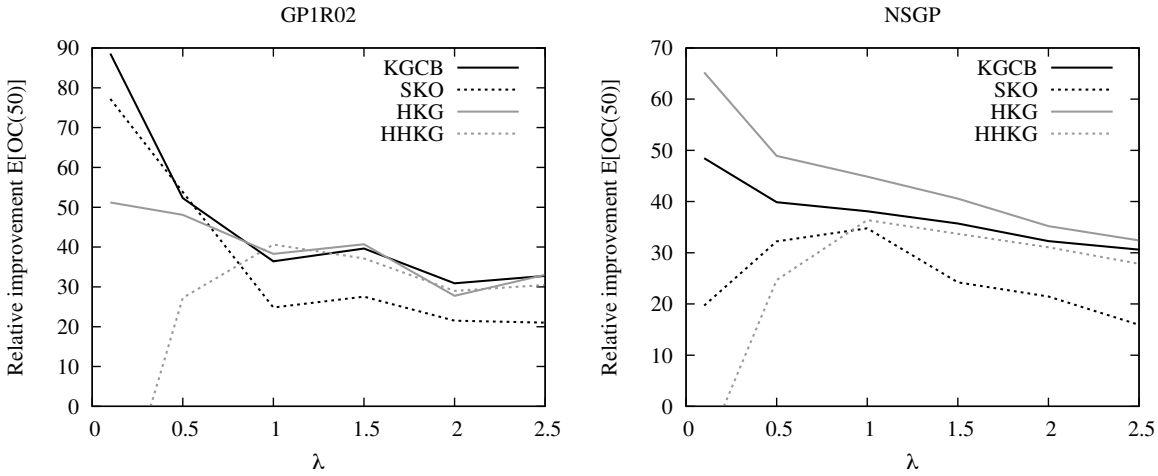


Figure 7: Sensitivity to the measurement noise.

As a final test with one-dimensional functions, we now vary the aggregation structure used by HKG. The results can be found in Figure 8. Obviously, HKG is sensitive to the choice of aggregation structure. The aggregation function with $\omega = 16$ is so coarse that, even on the lowest aggregation level, there exists aggregate alternatives that have local maxima as well as local minima in their aggregated set. We also see that the performance under the $\omega = 2/4$ structure is close to that of $\omega = 4$, which indicates that having some symmetry in the aggregation function is preferable. When comparing the two figures, we see that the impact of the aggregation function decreases with increasing λ. The reason for this is that with higher λ, more weight is given to the more aggregate levels. As a result, the benefit of having more precise lower aggregation levels decreases.

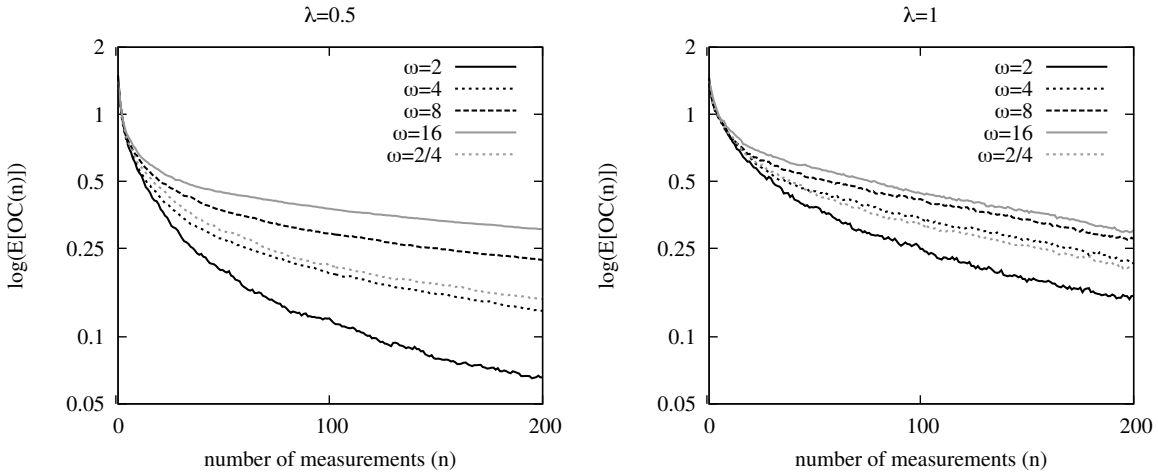


Figure 8: Sensitivity of HKG to the aggregation function.

7.2 Two-dimensional Functions

An overview of results for the two-dimensional functions can be found in Table 7. From these results we draw the following conclusions:

1. On the standard test functions, SHCB-DS and TBRANIN, HKG is outperformed by KGCB and SKO. However, with increasing λ, HKG still outperforms SKO.

2. In case of the Six-hump camel back function, just extending the domain a bit (where the extended part of the domain only contains points with large opportunity costs) has a major impact on the results. With the exception of one outcome (KGCB with $\lambda = 1$), the opportunity costs increase for all policies. This makes sense because there are simply more alternatives with higher opportunity costs. For KGCB and SKO, these extreme values also play a role in fitting the Gaussian process prior. As a result, we have a less reliable fit at the area of interest, something especially SKO suffers from. Obviously, also HKG 'loses' measurements on these extreme values. However, their influence on the fit (via the aggregation function) is limited since HKG automatically puts a low weight on them. As a result, HKG outperforms the other policies in almost all cases.

3. Shuffling the Six-hump camel back has a similar influence to extending the domain. In all cases, HKG outperforms KGCB and SKO. Shuffling the TBRANIN has an especially large impact on the performance of KGCB and SKO. However, not all performance differences with the shuffled TBRANIN are significant due to relatively large variances, especially in the case of $n = 50$.

7.3 Example Case

The results for the transportation application can be found in Figure 9. As mentioned in Section 6, the first two dimensions of this problem are described by the Six-hump camel back function on the small domain. This function is also considered in Huang et al. (2006) and Frazier et al. (2009) where the policies SKO and KGCB respectively are introduced. Compared to HKG, these policies perform relatively well on this standard test function. It is interesting to see that the addition of a third, categorical, dimension changes the situation.

As can be seen from Figure 9, HKG outperforms SKO and KGCB for both values of λ and almost all intermediate values of n. Measured at $n = 100$ and $n = 200$, the differences between HKG and both KGCB and SKO are significant (again using the 0.05 level). The hybrid policy HHKG is doing remarkably well; the differences with HKG at $n = 200$ are not significant, which is partly due to the fact that the variances with HHKG are higher. The performance of HHKG is especially remarkable since this policy requires only a fraction of the computation time of the others. Given, the large number of measurements and alternatives, the running times of KGCB, SKO, and HKG take multiple hours per replication whereas HHKG requires around 10 seconds.

8. Conclusions

We have presented an efficient learning strategy to optimize an arbitrary function that depends on a multi-dimensional vector with numerical and categorical attributes. We do not attempt to fit a

		$\mathbb{E}[OC(50)]$				$\mathbb{E}[OC(100)]$			
Function	$\sqrt{\lambda}$	KGCB	SKO	HKG	HHKG	KGCB	SKO	HKG	HHKG
SHCB-DS	1	**0.28**	**0.35**	**0.37**	0.55	**0.18**	0.30	0.29	0.33
	2	**0.56**	**0.53**	0.72	0.84	**0.38**	**0.41**	**0.48**	0.54
	4	**0.95**	**1.17**	**1.19**	**1.08**	**0.72**	**0.89**	**0.92**	**0.78**
SHCB-DB	1	**0.53**	0.70	**0.57**	**0.58**	**0.12**	0.53	0.41	0.35
	2	1.03	1.11	**0.73**	0.92	0.83	0.95	**0.46**	0.64
	4	1.55	1.50	**1.21**	**1.34**	1.33	1.42	**0.89**	**1.05**
SHCB-DS-SF	1	0.60	0.63	**0.32**	0.51	0.35	0.41	**0.20**	0.31
	2	0.90	0.95	**0.67**	**0.81**	0.69	0.86	**0.42**	**0.51**
	4	**1.17**	1.44	**1.13**	**1.22**	**1.05**	1.23	**0.86**	**0.89**
SHCB-DB_SF	1	1.19	0.75	**0.48**	**0.65**	0.60	0.81	**0.29**	**0.38**
	2	1.66	1.23	**0.69**	0.99	1.08	1.07	**0.48**	0.64
	4	1.85	1.41	**1.00**	**1.14**	1.36	1.43	**0.74**	**0.86**
TBRANIN	2	**0.16**	0.30	2.33	3.30	**0.08**	0.23	0.79	1.57
	4	**0.67**	1.21	2.40	4.12	**0.33**	0.85	1.16	2.27
	8	**3.64**	**2.88**	**3.81**	4.99	**1.29**	2.03	2.12	2.80
TBRANIN-SF	2	21.85	**1.42**	2.18	3.76	7.59	1.42	**0.82**	1.68
	4	10.61	**2.84**	**2.57**	4.55	3.17	1.99	**1.25**	2.22
	8	7.63	**5.01**	**4.07**	**4.50**	6.47	3.46	**2.33**	**2.48**

Table 7: Results for the 2-dimensional test functions.

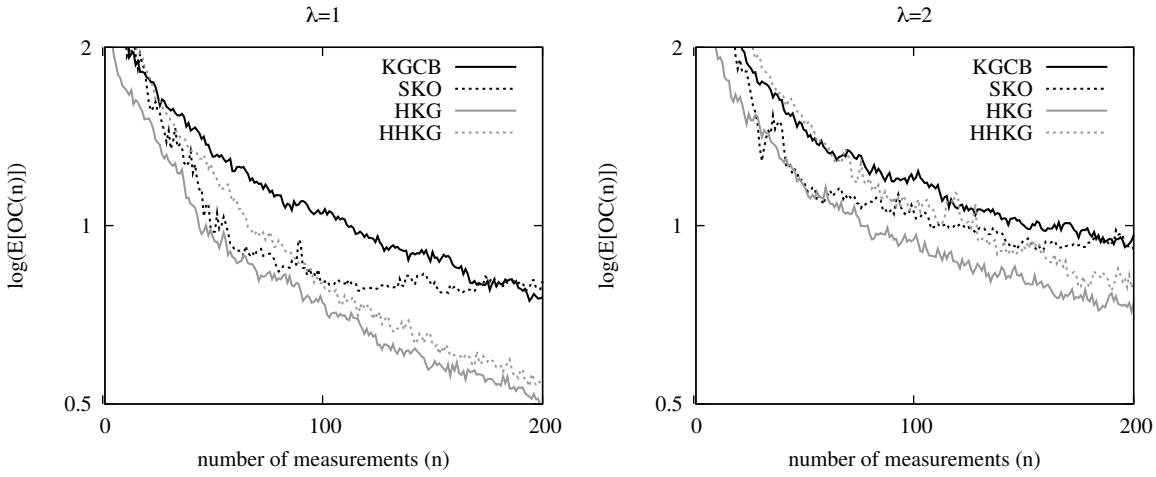


Figure 9: Results for the transportation application.

function to this surface, but we do require a family of aggregation functions. We produce estimates of the value of the function using a Bayesian adaptation of the hierarchical estimation procedure suggested by George et al. (2008). We then present an adaptation of the knowledge-gradient procedure of Frazier et al. (2009) for problems with correlated beliefs. That method requires the use of a known covariance matrix, while in our strategy, we compute covariances from our statistical model.

The hierarchical knowledge-gradient (HKG) algorithm shares the inherent steepest ascent property of the knowledge gradient algorithm, which chooses samples that produce the greatest single-sample improvement in our ability to maximize the function. We also prove that the algorithm is guaranteed to produce the optimal solution in the many-sample limit, since the HKG algorithm measures every alternative infinitely often.

We close with experimental results on a class of one and two dimensional scalar functions and a multi-attribute problem drawn from a transportation application. In these experiments, HKG performs better than all competing policies tested, when measured by average performance across all problems. In particular, it outperforms the other policies on functions for which the use of an aggregation function seems to be a natural choice (e.g., those with categorical dimensions), but it also performs well on problems for which the other policies are specifically designed.

The limitation of the HKG policy is that it requires a given aggregation structure, which means that we depend on having some insight into the problem. When this is the case, the ability to capture this knowledge in an aggregation structure is actually a strength, since we can capture the most important features in the highest levels of aggregation. If we do not have this insight, designing the aggregation functions imposes an additional modeling burden.

We mention two other limitations that give rise to further research. First, we observe convergence problems for HKG in the case of low measurement variance where HKG tends to become to confident about values of alternatives never measured before. We describe this issue in more detail in Appendix E. Second, the HKG policy requires enumerating all possible choices before determining the next measurement. This is appropriate for applications where we need to make good choices with a small number of measurements, typically far smaller than the set of alternatives. However, this limits our approach to handling perhaps thousands of choices, but not millions. A solution here would be to create a limited set of choices for the next measurement. As a starting point we might create this set by running HKG on a higher aggregation level which has fewer elements. Preliminary experiments have shown that this method can drastically reduce computation time without harming the performance too much. Future research could further explore such computational improvements.

We mention one final direction for future research. While we have presented a proof of convergence for the HKG policy, there are no theoretical results currently available that bound the rate at which it converges. Future research could derive such bounds, or could create new techniques appropriate for problems with hierarchical aggregation structures that have bounds on their convergence rates. One approach for creating such techniques would be to begin with an online learning technique with bounds on cumulative regret, and then to use a batch-to-online conversion technique to derive a procedure with a bound on the rate at which its terminal regret converges to zero.

Appendix A.

The overall sampling and updating procedure used for HKG is shown in Algorithm 1 and an outline for the HKG measurement decision is shown in Algorithm 2.

Algorithm 1 Sampling and updating procedure.

Require: Inputs $(G^g)\forall g \in \mathcal{G}$, $(\lambda_x)\forall x \in \mathcal{X}$, and $\underline{\delta}$

Initialize $(\mu_x^0, \beta_x^0, \tilde{g}_x^0)\forall x \in \mathcal{X}$, $(\mu_x^{g,0}, \beta_x^{g,0}, \delta_x^{g,0}, \beta_x^{g,0,\varepsilon})\forall g \in \mathcal{G}, x \in \mathcal{X}$
for $n = 1$ to N **do**
  Use Algorithm 2 to get measurement decision x^*
  Measure x^* and observe $\hat{y}_{x^*}^n$
  Compute $\tilde{g}_x^n \ \forall x \in \mathcal{X}$
  Compute $\mu_x^{g,n}$, $\beta_x^{g,n}$, and $\delta_x^{g,n} \ \forall g \in \mathcal{G}, x \in \mathcal{X}$ using (2), (3), and (9)
  Compute $w_x^{g,n}$ with $(\sigma_x^{g,n})^2 = 1/\beta_x^{g,n} \ \forall g \in \mathcal{G}, x \in \mathcal{X}$ using (8)
  Compute $\beta_x^{g,n,\varepsilon} = (\sigma_x^{g,n,\varepsilon})^{-2} \ \forall g \in \mathcal{G}, x \in \mathcal{X}$ using (10)
  Compute μ_x^n and β_x^n with $(\sigma_x^{g,n})^2 = 1/\beta_x^{g,n} \ \forall x \in \mathcal{X}$ using (4) and (5)
end for
return $x^N \in \arg\max_{x \in \mathcal{X}} \mu_x^N$

Algorithm 2 Hierarchical knowledge-gradient measurement decision.

Require: Inputs $(G^g)\forall g \in \mathcal{G}$, $(\lambda_x, \mu_x^n, \beta_x^n)\forall x \in \mathcal{X}$, $(\mu_x^{g,n}, \beta_x^{g,n}, \delta_x^{g,n}, \beta_x^{g,n,\varepsilon})\forall g \in \mathcal{G}, x \in \mathcal{X}$

for $x = 1$ to M **do**
  Compute $\tilde{\sigma}_x^{g,n} \ \forall g \in \mathcal{G}$ using (15) with $(\sigma_x^n)^2 = 1/\beta_x^n$
  for $x' = 1$ to M **do**
    Compute $w_{x'}^{g,n}(x) \ \forall g \in \mathcal{G}$ using (17)
    Compute $a_{x'}^n(x)$ and $b_{x'}^n(x)$ using (19) and (20)
  end for
  Sort the sequence of pairs $(a_i^n(x), b_i^n(x))_{i=1}^M$ so that the $b_i^n(x)$ are in non-decreasing order and ties are broken so that $a_i^n(x) < a_{i+1}^n(x)$ if $b_i^n(x) = b_{i+1}^n(x)$.
  for $i = 1$ to $M-1$ **do**
    if $b_i^n(x) = b_{i+1}^n(x)$ **then**
      Remove entry i from the sequence $(a_i^n(x), b_i^n(x))_{i=1}^M$
    end if
  end for
  Use Algorithm 1 from Frazier et al. (2009) to compute $\tilde{a}_i^n(x)$ and $\tilde{b}_i^n(x)$
  Compute $\upsilon_x^{KG,n}$ using (21)
  if $x = 1$ or $\upsilon_x^{KG,n} \geq \upsilon^*$ **then**
    $\upsilon^* = \upsilon_x^{KG,n}$, $x^* = x$
  end if
end for
return x^*

Appendix B.

Proposition 4 *The posterior belief on θ_x given observations up to time n for all aggregation levels is normally distributed with mean and precision*

$$\begin{aligned}\mu_x^n &= \frac{1}{\beta_x^n}\left[\beta_x^0\mu_x^0 + \sum_{g\in\mathcal{G}}\left((\sigma_x^{g,n})^2+\nu_x^g\right)^{-1}\mu_x^{g,n}\right],\\ \beta_x^n &= \beta_x^0 + \sum_{g\in\mathcal{G}}\left((\sigma_x^{g,n})^2+\nu_x^g\right)^{-1}.\end{aligned}$$

Proof Let $Y_x^{g,n} = \left\{\hat{y}^{g,m}_{x^{m-1}} : m \le n, G^g(x) = G^g(x^{m-1})\right\}$. This is the set of observations from level g pertinent to alternative x.

Let H be a generic subset of $\mathcal{G}$. We show by induction on the size of the set H that the posterior on θ_x given $Y_x^{g,n}$ for all $g \in H$ is normal with mean and precision

$$\begin{aligned}\mu_x^{H,n} &= \frac{1}{\beta_x^{H,n}}\left[\beta_x^0\mu_x^0 + \sum_{g\in H}\left((\sigma_x^{g,n})^2+\nu_x^g\right)^{-1}\mu_x^{g,n}\right],\\ \beta_x^{H,n} &= \beta_x^0 + \sum_{g\in H}\left((\sigma_x^{g,n})^2+\nu_x^g\right)^{-1}.\end{aligned}$$

Having shown this statement for all H, the proposition follows by taking $H = \mathcal{G}$.

For the base case, when the size of H is 0, we have $H = \emptyset$ and the posterior on θ is the same as the prior. In this case the induction statement holds because $\mu_x^{H,n} = \mu_x^0$ and $\beta_x^{H,n} = \beta_x^0$.

Now suppose the induction statement holds for all H of a size m and consider a set H' with $m+1$ elements. Choose $g \in H'$ and let $H = H' \setminus \{g\}$. Then the induction statement holds for H because it has size m. Let $\mathbb{P}_H$ denote the prior conditioned on $Y_x^{g',n}$ for $g' \in H$, and define $\mathbb{P}_{H'}$ similarly. We show that the induction statement holds for H' by considering two cases: $Y_x^{g,n}$ empty and non-empty.

If $Y_x^{g,n}$ is empty, then the distribution of θ_x is the same under both $\mathbb{P}_H$ and $\mathbb{P}_{H'}$. Additionally, from the fact that $\sigma_x^{g,n} = \infty$ it follows that $\mu_x^{H,n} = \mu_x^{H',n}$ and $\beta_x^{H,n} = \beta_x^{H',n}$. Thus, the induction statement holds for H'.

Now consider the case that $Y_x^{g,n}$ is non-empty. Let φ be the normal density, and let y denote the observed value of $Y_x^{g,n}$. Then, by the definitions of H and H', and by Bayes rule,

$$\mathbb{P}_{H'}\{\theta_x \in du\} = \mathbb{P}_H\{\theta_x \in du \mid Y_x^{g,n} = y\} \propto \mathbb{P}_H\{Y_x^{g,n} \in dy \mid \theta_x = u\}\,\mathbb{P}_H\{\theta_x \in du\}.$$

The second term may be rewritten using the induction statement as $\mathbb{P}_H\{\theta_x \in du\} = \varphi\left((u-\mu_x^{H,n})/\sigma_x^{H,n}\right)$. The first term may be rewritten by first noting that $Y_x^{g,n}$ is independent of $Y_x^{g',n}$ for $g' \in H$, and then conditioning on θ_x^g. This provides

$$\begin{aligned}\mathbb{P}_H\{Y_x^{g,n} \in dy \mid \theta_x = u\} &= \mathbb{P}\{Y_x^{g,n} \in dy \mid \theta_x = u\}\\ &= \int_{\mathbb{R}} \mathbb{P}\{Y_x^{g,n} \in dy \mid \theta_x^g = v\}\,\mathbb{P}\{\theta_x^g = v \mid \theta_x = u\}\,dv\\ &\propto \int_{\mathbb{R}} \varphi\left(\frac{\mu_x^{g,n}-v}{\sigma_x^{g,n}}\right)\varphi\left(\frac{v-u}{\sqrt{\nu_x^g}}\right)dv\\ &\propto \varphi\left(\frac{\mu_x^{g,n}-u}{\sqrt{(\sigma_x^{g,n})^2+\nu_x^g}}\right).\end{aligned}$$

In the third line, we use the fact that $\mathbb{P}_H\left\{Y_x^{g,n} \in dy \mid \theta_x^g = v\right\}$ is proportional (with respect to u) to $\varphi\left((\mu_x^{g,n} - v)/\sigma_x^{g,n}\right)$, which may be shown by induction on n from the recursive definitions for $\mu_x^{g,n}$ and $\beta_x^{g,n}$.

Using this, we write

$$\mathbb{P}_{H'}\{\theta_x \in du\} \propto \varphi\left(\frac{u-\mu_x^{g,n}}{\sqrt{(\sigma_x^{g,n})^2+\nu_x^g}}\right)\varphi\left(\frac{u-\mu_x^{H,n}}{\sigma_x^{H,n}}\right) \propto \varphi\left(\frac{u-\mu_x^{H',n}}{\sigma_x^{H',n}}\right),$$

which follows from an algebraic manipulation that involves completing the square.

This shows that the posterior is normally distributed with mean $\mu_x^{H',n}$ and variance $(\sigma_x^{H',n})^2$, showing the induction statement. ■

Appendix C.

This appendix contains all the lemmas required in the proofs of Theorem 1 and Corollaries 2 and 3.

Lemma 5 *If $z_1, z_2, \ldots$ is a sequence of non-negative real numbers bounded above by a constant $a < \infty$, and $s_n = \sum_{k \le n} z_k$, then $\sum_n (z_n/s_n)^2 \mathbf{1}_{\{s_n > 0\}}$ is finite.*

Proof Let $n_0 = \inf\{n \ge 0 : s_n > 0\}$, and, for each integer k, let $n_k = \inf\{n \ge 0 : s_n > ka\}$. Then, noting that $s_n = 0$ for all $n < n_0$ and that $s_n > 0$ for all $n \ge n_0$, we have

$$\sum_n (z_n/s_n)^2 \mathbf{1}_{\{s_n>0\}} = \left[\sum_{n_0 \le n < n_1} (z_n/s_n)^2\right] + \sum_{k=1}^{\infty}\left[\sum_{n_k \le n < n_{k+1}} (z_n/s_n)^2\right].$$

We show that this sum is finite by showing that the two terms are both finite. The first term may be bounded by

$$\sum_{n_0 \le n < n_1} (z_n/s_n)^2 \le \sum_{n_0 \le n < n_1} (z_n/z_{n_0})^2 \le \left(\sum_{n_0 \le n < n_1} z_n/z_{n_0}\right)^2 \le (a/z_{n_0})^2 < \infty.$$

The second term may be bounded by

$$\begin{aligned}
\sum_{k=1}^{\infty}\sum_{n=n_k}^{n_{k+1}-1} (z_n/s_n)^2 \le \sum_{k=1}^{\infty}\sum_{n=n_k}^{n_{k+1}-1} (z_n/ka)^2 &\le \sum_{k=1}^{\infty}\left(\sum_{n=n_k}^{n_{k+1}-1} z_n/ka\right)^2 \\
= \sum_{k=1}^{\infty}\left(\frac{s_{n_{k+1}-1} - s_{n_k} + z_{n_k}}{ka}\right)^2 &\le \sum_{k=1}^{\infty}\left(\frac{(k+1)a - ka + a}{ka}\right)^2 \\
= \sum_{k=1}^{\infty}(2/k)^2 = \frac{2}{3}\pi^2 &< \infty.
\end{aligned}$$

■

Lemma 6 *Assume that samples from any fixed alternative x are iid with finite variance. Fix $g \in \mathcal{G}$ and $x \in \mathcal{X}$ and let*

$$y_x^n = \left[\sum_{m<n} \beta_x^{g,m,\varepsilon} \hat{y}_x^{m+1} \mathbf{1}_{\{x^m=x\}}\right] \Big/ \left[\sum_{m<n} \beta_x^{g,m,\varepsilon} \mathbf{1}_{\{x^m=x\}}\right]$$

for all those n for which the denominator is strictly positive, and let $y_x^n = 0$ for those n for which the denominator is zero. Then, $\sup_n |y_x^n|$ is finite almost surely.

Proof Let $\alpha^n = \left[\beta_x^{g,n,\varepsilon} \mathbf{1}_{\{x^n=x\}}\right] / \left[\sum_{m\le n} \beta_x^{g,m,\varepsilon} \mathbf{1}_{\{x^m=x\}}\right]$, so that

$$y_x^{n+1} = (1-\alpha^n) y_x^n + \alpha^n \hat{y}_x^{n+1}.$$

Let v_x be the variance of samples from alternative x, which is assumed finite. Let $M^n = (y_x^n - \theta_x)^2 + \sum_{m=n}^{\infty} \mathbf{1}_{\{x^m=x\}} v_x (\alpha^m)^2$, and note that Lemma 5 and the upper bound $(\min_{x'} \lambda_{x'})^{-1}$ on $\beta_x^{g,m,\varepsilon}$ together imply that M^0 is finite. We will show that M^n is a supermartingale with respect to the filtration generated by $(\hat{y}_x^n)_{n=1}^{\infty}$. In this proof, we write $\mathbb{E}^n$ to indicate $\mathbb{E}[\cdot \mid \mathcal{F}^n]$, the conditional expectation taken with respect to $\mathcal{F}^n$.

Consider $\mathbb{E}^n[M^{n+1}]$. On the event $\{x^n \neq x\}$ (which is $\mathcal{F}^n$ measurable), we have $M^{n+1} = M^n$ and $\mathbb{E}^n\left[M^{n+1} - M^n\right] = 0$. On the event $\{x^n = x\}$ we compute $\mathbb{E}^n\left[M^{n+1} - M^n\right]$ by first computing

$$\begin{aligned} M^{n+1} - M^n &= (y_x^{n+1} - \theta_x)^2 - (y_x^n - \theta_x)^2 - v_x(\alpha^n)^2 \\ &= ((1-\alpha^n) y_x^n + \alpha^n \hat{y}_x^{n+1} - \theta_x)^2 - (y_x^n - \theta_x)^2 - v_x(\alpha^n)^2 \\ &= -(\alpha^n)^2 (y_x^n - \theta_x)^2 + 2\alpha^n (1-\alpha^n)(y_x^n - \theta_x)(\hat{y}_x^{n+1} - \theta_x) \\ &\quad + (\alpha^n)^2 \left[(\hat{y}_x^{n+1} - \theta_x)^2 - v_x\right]. \end{aligned}$$

Then, the $\mathcal{F}^n$ measurability of α^n and y_x^n, together with the facts that $\mathbb{E}^n\left[\hat{y}_x^{n+1} - \theta_x\right] = 0$ and $\mathbb{E}^n\left[\left(\hat{y}_x^{n+1} - \theta_x\right)^2\right] = v_x$, imply

$$\mathbb{E}\left[M^{n+1} - M^n\right] = -(\alpha^n)^2 (y_x^n - \theta_x)^2 \le 0.$$

Since $M^n \ge 0$ and $M^0 < \infty$, the integrability of M^n follows. Thus, $(M^n)_n$ is a supermartingale and has a finite limit almost surely. Then,

$$\lim_{n\to\infty} M^n = \lim_{n\to\infty} (y_x^n - \theta_x)^2 + \sum_{m=n}^{\infty} \mathbf{1}_{\{x^m=x\}} v_x (\alpha^m)^2 = \lim_{n\to\infty} (y_x^n - \theta_x)^2.$$

The almost sure existence of a finite limit for $(\hat{y}_x^n - \theta_x)^2$ implies the almost sure existence of a finite limit for $|\hat{y}_x^n - \theta_x|$ as well. Finally, the fact that a sequence with a limit has a finite supremum implies that $\sup_n |y_x^n| \le \sup_n |y_x^n - \theta_x| + |\theta_x| < \infty$ almost surely. ■

Lemma 7 *Assume that samples from any fixed alternative x are iid with finite variance. Let $x, x' \in \mathcal{X}$, $g \in \mathcal{G}$. Then $\sup_n |\mu_x^{g,n}|$ and $\sup_n |a_{x'}^n(x)|$ are almost surely finite.*

Proof We first show $\sup_n |\mu_x^{g,n}| < \infty$ almost surely for fixed x and g. We write $\mu_x^{g,n}$ as

$$\mu_x^{g,n} = \frac{\beta_x^{g,0}\mu_x^{g,0} + \sum_{m<n} \beta_x^{g,m,\varepsilon} \mathbf{1}_{\{x^m \in \mathcal{X}^g(x)\}} \hat{y}_{x^m}^{m+1}}{\beta_x^{g,0} + \sum_{m<n} \beta_x^{g,m,\varepsilon} \mathbf{1}_{\{x^m \in \mathcal{X}^g(x)\}}} = p_0^n \mu_x^{g,0} + \sum_{x' \in \mathcal{X}^g(x)} p_{x'}^n y_{x'}^n,$$

where the $y_{x'}^n$ are as defined in Lemma 6 and the $p_{x'}^n$ are defined for $x' \in \mathcal{X}^g(x)$ by

$$p_0^n = \frac{\beta_x^{g,0}}{\beta_x^{g,0} + \sum_{m<n} \beta_x^{g,m,\varepsilon} \mathbf{1}_{\{x^m \in \mathcal{X}^g(x)\}}}, \qquad p_{x'}^n = \frac{\sum_{m<n} \beta_x^{g,m,\varepsilon} \mathbf{1}_{\{x^m = x'\}}}{\beta_x^{g,0} + \sum_{m<n} \beta_x^{g,m,\varepsilon} \mathbf{1}_{\{x^m \in \mathcal{X}^g(x)\}}}.$$

Note that p_0^n and each of the $p_{x'}^n$ are bounded uniformly between 0 and 1. We then have

$$\sup_n |\mu_x^{g,n}| \le \sup_n \left[|\mu_x^{g,0}| + \sum_{x' \in \mathcal{X}^g(x)} |y_{x'}^n| \right] \le |\mu_x^{0,g}| + \sum_{x' \in \mathcal{X}^g(x)} \sup_n |y_{x'}^n|.$$

By Lemma 6, $\sup_n |y_{x'}^n|$ is almost surely finite, and hence so is $\sup_n |\mu_x^{g,n}|$.

We now turn our attention to $a_{x'}^n(x)$ for fixed x and x'. $a_{x'}^n(x)$ is a weighted linear combinations of the terms $\mu_{x'}^{g,n}$, $g \in \mathcal{G}$ (note that $\mu_{x'}^n$ is itself a linear combination of such terms), where the weights are uniformly bounded. This, together with the almost sure finiteness of $\sup_n |\mu_{x'}^{g,n}|$ for each g, implies that $\sup_n |a_{x'}^n(x)|$ is almost surely finite. ■

Lemma 8 *Assume that $\underline{\delta} > 0$ and samples from any fixed alternative x are iid with finite variance. Let $\mathcal{X}_\infty$ be the (random) set of alternatives measured infinitely often by HKG. Then, for each $x', x \in \mathcal{X}$, the following statements hold almost surely,*

- *If $x \in \mathcal{X}_\infty$ then $\lim_{n\to\infty} b_{x'}^n(x) = 0$ and $\lim_{n\to\infty} b_x^n(x') = 0$.*
- *If $x \notin \mathcal{X}_\infty$ then $\liminf_{n\to\infty} b_x^n(x) > 0$.*

Proof Let x' and x be any pair of alternatives.

First consider the case $x \in \mathcal{X}_\infty$. Let $g \in \mathcal{G}(x',x)$ and $B = \sup_n (\sigma_x^{g,n,\varepsilon})^2$. Lemma 7 and (10) imply that B is almost surely finite. Since $\beta_x^{g,n,\varepsilon} \ge 1/B$ for each n, we have $\beta_x^{g,n} \ge m_x^{g,n} B$. Then $x \in \mathcal{X}_\infty$ implies $\lim_{n\to\infty} m_x^{g,n} = \infty$ and $\lim_{n\to\infty} \beta_x^{g,n} = \infty$. Also, x and x' share aggregation level g, so $\beta_x^{g,n} = \beta_{x'}^{g,n}$ and $\lim_{n\to\infty} \beta_{x'}^{g,n} = \infty$. Then consider $\tilde{\sigma}_x^{g,n}$ for n large enough that we have measured alternative x at least once. From (10), $(\sigma_x^{g,n,\varepsilon})^2 \ge \lambda_x / |\mathcal{X}^g(x)|$, which gives a uniform upper bound $\beta_x^{g,n,\varepsilon} \le |\mathcal{X}^g(x)|/\lambda_x$. Also, the definition (6) implies $(\sigma_x^n)^2 \le (\sigma_x^{g,n})^2 \le 1/B$. This, the definition (15), and $\lim_{n\to\infty} \beta_x^{g,n} = \infty$ together imply $\lim_{n\to\infty} \tilde{\sigma}_x^{g,n} = 0$. The limit $\lim_{n\to\infty} \tilde{\sigma}_{x'}^{g,n} = 0$ follows similarly from the bounds $\beta_{x'}^{g,n,\varepsilon} \le |\mathcal{X}^g(x)|/\lambda_{x'}$ and $(\sigma_{x'}^n)^2 \le (\sigma_x^{n,g})^2 \le 1/B$, and $\lim_{n\to\infty} \beta_{x'}^{g,n} = \infty$. Hence, (20) and the boundedness of the weights $w_{x'}^{g,n}$ and $w_x^{g,n}$ imply $\lim_{n\to\infty} b_{x'}^n(x) = \lim_{n\to\infty} b_x^n(x') = 0$.

Now consider the case $x \notin \mathcal{X}_\infty$. We show that $\liminf_{n\to\infty} b_x^n(x) > 0$. From (20) and $0 \in \mathcal{G}(x,x)$,

$$b_x^n(x) \ge w_x^{0,n}(x) \frac{(\lambda_x)^{-1} \sqrt{\left(\sum_{g' \in \mathcal{G}} \beta_x^{g',n}\right)^{-1} + \lambda_x}}{\beta_x^{0,n} + (\lambda_x)^{-1}}.$$

Because $x \notin \mathcal{X}_\infty$, there is some random time $N_1 < \infty$ after which we do not measure x, and $\beta_x^{0,n} \leq \beta_x^{N_1,0}$ for all n.

$$b_x^n(x) \geq w_x^{0,n}(x) \frac{(\lambda_x)^{-1}\sqrt{\lambda_x}}{\beta_x^{0,N_1} + (\lambda_x)^{-1}},$$

where the weights are given by

$$w_x^{0,n}(x) = \frac{\left(\beta_x^{0,n} + (\lambda_x)^{-1}\right)^{-1}}{\left(\beta_x^{0,n} + (\lambda_x)^{-1}\right)^{-1} + \sum_{g \in \mathcal{G}\setminus\{0\}} \psi_x^{g,n}},$$

with

$$\psi_x^{g,n} = \left((\beta_x^{g,n} + \beta_x^{g,n,\varepsilon})^{-1} + (\delta_x^{g,n})^2\right)^{-1}.$$

We now show $\limsup_n \psi_x^{g,n} < \infty$ for all $g \in \mathcal{G}\setminus\{0\}$. We consider two cases for g. In the first case, suppose that an alternative in $\mathcal{X}^g(x)$ is measured at least once. Then, for all n after this measurement, $m_x^{g,n} > 0$ and $\delta_x^{g,n} \geq \underline{\delta}$ (by (9)), implying $\psi_x^{g,n} \leq \underline{\delta}^{-2}$ and $\limsup_n \psi_x^{g,n} \leq \underline{\delta}^{-2} < \infty$. In the second case, suppose no alternative in $\mathcal{X}^g(x)$ is ever measured. Then, $\limsup_n \psi_x^{g,n} \leq \limsup_n \beta_x^{g,n} + \beta_x^{g,n,\varepsilon} < \infty$.

Finally, $\limsup_n \psi_x^{g,n} < \infty$ and $\left(\beta_x^{0,n} + (\lambda_x)^{-1}\right)^{-1} \geq \left(\beta_x^{0,N_1} + (\lambda_x)^{-1}\right)^{-1} > 0$ together imply $\liminf_{n\to\infty} w_x^{0,n}(x) > 0$. This shows $\liminf_{n\to\infty} b_x^n(x) > 0$. ■

Lemma 9 *Let $a \in \mathbb{R}^d$ with $\max_i |a_i| \leq c$, $b \in \mathbb{R}^d$, and let Z be a standard normal random variable. If $x \neq x'$, then,*

$$\mathbb{E}\left[\max_i a_i + b_i Z\right] - \max_i a_i \geq \frac{|b_{x'} - b_x|}{2} f\left(\frac{-4c}{|b_{x'} - b_x|}\right),$$

where this expression is understood to be 0 if $b_{x'} = b_x$.

Proof Let $x^* \in \arg\max_i a_i$ and $a^* = \max_i a_i$. Then adding and subtracting $a_{x^*} + b_{x^*}Z = a^* + b_{x^*}Z$ and observing $\mathbb{E}[b_{x^*}Z] = 0$ provides

$$\begin{aligned}\mathbb{E}\left[\max_i a_i + b_i Z\right] - a^* &= \mathbb{E}\left[\left(\max_i (a_i - a^*) + (b_i - b_{x^*})Z\right) + a^* + b_{x^*}Z\right] - a^* \\ &= \mathbb{E}\left[\max_i (a_i - a^*) + (b_i - b_{x^*})Z\right].\end{aligned}$$

Let $j \in \arg\max_{i\in\{x,x'\}} |b_i - b^*|$. Then, by taking the maximum in the previous expression over only j and x^*, we obtain the lower bound

$$\begin{aligned}\mathbb{E}\left[\max_i a_i + b_i Z\right] - a^* &\geq \mathbb{E}\left[\max\left(0, a_j - a^* + (b_j - b_{x^*})Z\right)\right] \\ &\geq \mathbb{E}\left[\max\left(0, -2c + (b_j - b_{x^*})Z\right)\right] \\ &= |b_j - b_{x^*}| f\left(\frac{-2c}{|b_j - b_{x^*}|}\right) \geq \frac{|b_{x'} - b_x|}{2} f\left(\frac{-4c}{|b_{x'} - b_x|}\right).\end{aligned}$$

The second line follows from the bound $\max_i |a_i| \leq c$. The equality in the third line can be verified by evaluating the expectation analytically (see, e.g., Frazier et al., 2008), where the expression is taken to be 0 if $b_j = b_{x^*}$. The inequality in the third line then follows from $|b_j - b^*| \geq |b_x - b_{x'}|/2$ and from f being an increasing non-negative function. ■

Appendix D.

Here we provide a brief description of the implementation of the policies considered in our numerical experiments.

Interval estimation (IE) The IE decision rule by Kaelbling (1993) is given by

$$x^n = \arg\max_{x \in \mathcal{X}} \left(\mu_x^n + z_{\alpha/2} \cdot \sigma_x^n\right)$$

where $z_{\alpha/2}$ is a tunable parameter. Kaelbling (1993) suggests that values of 2, 2.5 or 3 often works best. The IE policy is quite sensitive to this parameter. For example, we observe that the following cases require higher values for $z_{\alpha/2}$: more volatile functions (low values for ρ, see Section 6.2), a higher measurement variance λ, and higher measurement budget N. To find a value that works reasonably well on most problem instances, we tested values between 0.5 and 4 with increments of .1 and found that $z_{\alpha/2} = 2.3$ works best on average. Since we assume the measurement noise is known, we use $\sigma_x^n = \sqrt{\frac{\lambda}{m_x^n}}$, where m_x^n is the number of times x has been measured up to and including time n.

UCB1-Normal (UCB1) The study by Auer et al. (2002) proposes different variations of the Upper Confidence Bound (UCB) decision rule originally proposed by Lai (1987). The UCB1-Normal policy is proposed for problems with Gaussian rewards and is given by

$$x^n = \arg\max_{x \in \mathcal{X}} \left(\mu_x^n + 4\sqrt{\frac{\lambda \log n}{N_x^n}}\right).$$

The original presentation of the policy uses a frequentist estimate of the measurement variance λ, which we replace by the known value. We improve the performance of UCB1 by treating the coefficient 4 as a tunable parameter. As with IE, we observe that the performance is quite sensitive to the value of this parameter. Using a setup similar to IE, we found that a value of 0.9 produced the best results on average.

Independent KG (IKG) This is the knowledge-gradient policy as presented in Section 4.1 of this paper.

Boltzmann exploration (BOLTZ) Boltzmann exploration chooses its measurements by

$$\mathbb{P}(x^n = x) = \left(\frac{e^{\mu_x^n / T^n}}{\sum_{x' \in \mathcal{X}} e^{\mu_{x'}^n / T^n}}\right),$$

where the policy is parameterized by a decreasing sequence of "temperature" coefficients $(T^n)_{n=0}^{N-1}$. We tune this temperature sequence within the set of exponentially decreasing sequences defined by $T^{n+1} = \gamma T^n$ for some constant $\gamma \in (0,1]$. The set of all such sequences is parameterized by γ and T^N. We tested combinations of $\gamma \in \{.1,.2,...,1\}$ and $T^N \in \{.1,.5,1,2\}$ and found that the combination $\gamma = 1$ and $T^N = .3$ produces the best results on average.

Pure exploration (EXPL) The pure exploration policy measures each alternative x with the same probability, that is, $\mathbb{P}(x^n = x) = 1/M$.

Sequential Kriging Optimization (SKO) This is a blackbox optimization method from Huang et al. (2006) that fits a Gaussian process onto the observed variables. The hyperparameters of the Gaussian process prior are estimated using an initial Latin hypercube design with $2p+2$ measurements, with p being the number of dimensions, as recommended by Huang et al. (2006). After this initial phase we continue to update the hyperparameters, using maximum likelihood estimation, during the first 50 measurements. The parameters are updated at each iteration.

KG for Correlated Beliefs (KGCB) This is the knowledge-gradient policy for correlated beliefs as presented in Section 4.1. We estimate the hyperparameters in the same way as done with SKO.

Hierarchical KG (HKG) This is the hierarchical knowledge-gradient policy as presented in this paper. This policy only requires an aggregation function as input. We present these functions in Section 6.3.

Hybrid HKG (HHKG) In this hybrid policy, we only exploit the similarity between alternatives in the updating equations and not in the measurement decision. As a result, this policy uses the measurement decision of IKG and the updating equations of HKG. The possible advantage of this hybrid policy is that it is able to cope with similarity between alternatives without the computational complexity of HKG.

Appendix E.

Here we show more detailed results for the experiments on one-dimensional problems. A complete overview of the results for the one-dimensional experiments with $N = 500$ can be found in Table 8 and with $N = 200$ in Table 9.

Besides the conclusions from the main text, we mention a few additional observations based on the more detailed results.

First, from Table 9 we see that the relative performance of KGCB and SKO depends on the value of ρ. On relatively smooth functions with $\rho \geq 2$, SKO outperforms KGCB, whereas the opposite holds for $\rho < 2$.

Second, it is remarkable to see that in the independent truth case (IT), the policies that exploit correlation (KGCB and HKG) are doing so well and outperform IKG. The explanation is the following. After M measurements, IKG has sampled each alternative once and the implementation decision is the one with the highest value observed so far. Obviously, this is not a reliable estimate, especially with $\lambda \geq 0.5$. The policies KGCB and HKG tend to resample promising alternatives. So, after M measurements, they have a more reliable estimate for their implementation decision.

Function	$\sqrt{\lambda}$	N	EXPL	IKG	KGCB	SKO	HKG	IE	UCB	BOLTZ
GP1R05	0.5	250	0.206	0.090	0.061	**0.029**	0.072	0.077	0.073	0.133
		500	0.169	0.044	0.037	**0.027**	0.053	0.038	0.040	0.075
	1	250	0.344	0.170	0.131	0.142	**0.111**	0.174	0.183	0.242
		500	0.332	0.108	**0.093**	0.111	**0.092**	0.106	0.113	0.155
GP1R02	0.5	250	0.152	0.041	**0.024**	**0.024**	0.032	0.046	0.043	0.069
		500	0.106	0.022	**0.014**	0.019	**0.017**	0.024	0.025	0.048
	1	250	0.308	0.103	**0.084**	0.129	**0.077**	0.112	0.111	0.151
		500	0.298	0.057	**0.050**	0.120	**0.044**	0.062	0.061	0.113
GP1R01	0.5	250	0.196	0.057	**0.019**	0.038	0.043	0.043	0.053	0.088
		500	0.158	0.033	**0.009**	0.024	0.027	0.022	0.024	0.058
	1	250	0.424	0.162	**0.107**	0.218	**0.114**	0.138	0.166	0.192
		500	0.348	0.084	**0.064**	0.165	**0.069**	0.069	0.088	0.143
GP1R005	0.5	250	0.253	0.065	**0.017**	0.047	0.049	0.053	0.058	0.100
		500	0.183	0.027	**0.008**	0.037	0.031	0.019	0.019	0.070
	1	250	0.483	0.162	**0.093**	0.189	**0.100**	0.145	0.178	0.210
		500	0.432	0.084	**0.046**	0.147	0.061	0.073	0.080	0.143
NSGP	0.5	250	0.249	0.052	0.070	0.146	**0.049**	**0.046**	**0.043**	0.122
		500	0.186	0.024	0.044	0.121	0.026	**0.019**	**0.019**	0.076
	1	250	0.539	0.193	0.184	0.240	**0.124**	0.150	0.175	0.220
		500	0.443	0.092	0.113	0.194	**0.067**	**0.068**	**0.073**	0.141
IT	0.5	250	0.182	0.075	0.066	0.107	**0.060**	0.075	0.074	0.113
		500	0.153	0.047	0.045	0.092	**0.040**	**0.042**	0.046	0.093
	1	250	0.306	0.155	0.144	0.207	**0.108**	0.151	0.162	0.188
		500	0.253	0.097	0.101	0.188	**0.087**	0.094	0.099	0.168
GP1	0.5	250	0.202	0.063	**0.030**	0.034	0.049	0.055	0.057	0.098
		500	0.154	0.032	**0.017**	0.027	0.032	0.026	0.027	0.063
	1	250	0.390	0.149	**0.104**	0.170	**0.101**	0.143	0.160	0.198
		500	0.352	0.083	**0.063**	0.136	**0.066**	0.078	0.086	0.138
NS0	0.5	250	0.215	0.064	0.068	0.126	**0.055**	0.060	**0.059**	0.118
		500	0.169	0.035	0.044	0.106	**0.033**	**0.031**	**0.032**	0.085
	1	250	0.423	0.174	0.164	0.224	**0.116**	0.150	0.168	0.204
		500	0.348	0.094	0.107	0.191	**0.077**	**0.081**	0.086	0.154

Table 8: Results for the one-dimensional long experiments.

Function	$\sqrt{\lambda}$	N	EXPL	IKG	KGCB	SKO	HKG	HHKG
GP1R05	0.1	50	0.149	0.131	0.020	**0.001**	0.033	0.036
		200	0.102	0.008	0.006	**0.001**	0.008	0.008
	0.5	50	0.261	0.231	0.165	**0.078**	0.171	0.169
		200	0.216	0.097	0.075	**0.036**	0.085	0.080
	1	50	0.390	0.411	0.277	**0.210**	0.258	0.278
		200	0.359	0.222	0.150	0.148	**0.129**	0.162
GP1R02	0.1	50	0.039	0.038	0.010	**0.005**	0.026	0.050
		200	0.025	0.008	**0.003**	**0.002**	0.007	0.006
	0.5	50	0.203	0.187	0.079	**0.063**	0.092	0.126
		200	0.169	0.055	**0.029**	**0.029**	0.037	0.044
	1	50	0.396	0.389	**0.233**	**0.230**	**0.224**	0.257
		200	0.332	0.142	**0.096**	0.138	**0.097**	**0.087**
GP1R01	0.1	50	0.062	0.056	**0.007**	0.014	0.030	0.083
		200	0.036	0.006	**0.001**	0.008	0.008	0.005
	0.5	50	0.254	0.253	**0.121**	**0.117**	**0.132**	0.184
		200	0.218	0.065	**0.022**	0.043	0.055	0.054
	1	50	0.477	0.482	**0.303**	0.358	**0.294**	**0.283**
		200	0.441	0.182	**0.124**	0.235	**0.136**	**0.128**
GP1R005	0.1	50	0.111	0.099	**0.003**	0.011	0.047	0.144
		200	0.043	0.004	**0.000**	0.003	0.008	0.011
	0.5	50	0.342	0.336	**0.127**	0.157	0.170	0.222
		200	0.254	0.082	**0.021**	0.054	0.061	0.080
	1	50	0.577	0.482	0.329	0.411	**0.286**	0.401
		200	0.530	0.182	**0.118**	0.204	**0.123**	0.164
NSGP	0.1	50	0.168	0.143	0.087	0.135	**0.059**	0.184
		200	0.047	**0.003**	0.021	0.094	**0.005**	0.017
	0.5	50	0.391	0.373	0.235	0.265	**0.200**	0.294
		200	0.263	0.082	0.084	0.156	**0.066**	0.082
	1	50	0.692	0.627	0.428	0.451	**0.381**	0.440
		200	0.580	0.249	0.208	0.260	**0.153**	0.176
IT	0.1	50	0.053	0.050	**0.046**	0.052	**0.044**	**0.042**
		200	0.039	0.013	**0.012**	0.027	0.013	**0.011**
	0.5	50	0.212	0.203	0.144	0.178	0.141	**0.130**
		200	0.175	0.091	0.072	0.116	**0.065**	0.079
	1	50	0.305	0.310	**0.218**	0.298	**0.230**	**0.231**
		200	0.312	0.177	0.157	0.217	**0.128**	0.150

Table 9: Results for the one-dimensional normal experiments.

Obviously, there is a probability that KGCB and HKG do not measure the true optimal alternative after M measurements. However, given the way we generated this function, there are multiple alternatives close the the optimal one (we may expect 10% of the alternatives to be less then 0.1 from the optimum).

Finally, even though HKG seems to be quite competitive, there are some results that suggest future extensions of HKG. Specifically, HKG seems to have convergence problems in the low noise case ($\lambda = 0.1$). We see this from (i) the settings with $\lambda = 0.1$ and $n = 200$ where HKG underperforms IKG on three cases (two of them with significant differences), (ii) the settings with the one-dimensional long experiments where HKG is outperformed by IKG in three cases, each of them having a low value for λ and a large number of measurements, and (iii) the hybrid policy HHKG is outperformed by IKG on most of the $\lambda = 0.1$ cases. We believe that the source of this problem lies in the use of the base level $\tilde{g}_x^n$, that is, the lowest level g for which we have at least one observation on an aggregate alternative that includes alternative x ($m_x^{g,n} > 0$). We introduced this base level because we need the posterior mean μ_x^n and the posterior variance $(\sigma_x^n)^2$ for all alternatives, including those we have not measured. When λ is relatively small, the posterior variance on the aggregate levels $(\sigma_x^{g,n})^2$ increases relatively quickly; especially because the squared bias $(\delta_x^{g,n})^2$, which we use as an estimate for ν_x^g, is small at the base level (equal to the lower bound $\underline{\delta}$). As a result, we may become too confident about the value of an alternative we never measured. We may be able to resolve this by adding a prior on these functions, which obviously requires prior knowledge about the truth or additional measurements, or by tuning $\underline{\delta}$.

References

Jacob Abernethy, Elad Hazan, and Alexander Rakhlin. Competing in the dark: An efficient algorithm for bandit linear optimization. In *Proceedings of the 21st Annual Conference on Learning Theory (COLT)*, pages 263–274, 2008.

Peter Auer, Nicolò Cesa-Bianchi, and Paul Fischer. Finite-time analysis of the multiarmed bandit problem. *Machine Learning*, 47(2-3):235–256, 2002.

Peter L. Bartlett, Varsha Dani, Thomas P. Hayes, Sham Kakade, Alexander Rakhlin, and Ambuj Tewari. High-probability regret bounds for bandit online linear optimization. In *Proceedings of the 21st Annual Conference on Learning Theory (COLT)*, pages 335–342, 2008.

Russell R. Barton and Martin Meckesheimer. Metamodel-based simulation optimization. In Shane G. Henderson and Barry L. Nelson, editors, *Simulation*, volume 13 of *Handbooks in Operations Research and Management Science*, pages 535 – 574. Elsevier, 2006.

Robert E. Bechhofer. A single-sample multiple decision procedure for ranking means of normal populations with known variances. *The Annals of Mathematical Statistics*, 25(1):16–39, 1954.

Robert E. Bechhofer, Thomas J. Santner, and David M. Goldsman. *Design and Analysis of Experiments for Statistical Selection, Screening and Multiple Comparisons*. John Wiley & Sons, New York, NY, 1995.

Dimitri P. Bertsekas and David A. Castanon. Adaptive aggregation methods for infinite horizon dynamic programming. *IEEE Transactions on Automatic Control*, 34(6):589–598, 1989.

Dimitri P. Bertsekas and John N. Tsitsiklis. *Neuro-Dynamic Programming*. Athena Scientific, Belmont, MA, 1996.

Ronen I. Brafman and Moshe Tennenholtz. R-MAX - a general polynomial time algorithm for near-optimal reinforcement learning. *Journal of Machine Learning Research*, 3:213–231, 2003.

Franklin H. Branin. Widely convergent method for finding multiple solutions of simultaneous nonlinear equations. *IBM Journal of Reseach and Development*, 16(5):504–522, 1972.

Erik Brochu, Mike Cora, and Nando de Freitas. A tutorial on Bayesian optimization of expensive cost functions, with application to active user modeling and hierarchical reinforcement learning. Technical Report TR-2009-023, Department of Computer Science, University of British Columbia, 2009.

Sébastien Bubeck, Réemi Munos, Gilles Stoltz, and Csaba Szepesvári. Online optimization in X-armed bandits. In *Advances in Neural Information Processing Systems*, pages 201–208, 2009a.

Sébastien Bubeck, Rémi Munos, and Gilles Stoltz. Pure exploration in multi-armed bandits problems. In *Proceedings of the 20th International Conference on Algorithmic Learning Theory*, pages 23–37, 2009b.

Chun-Hung Chen, Hsiao-Chang Chen, and Liyi Dai. A gradient approach for smartly allocating computing budget for discrete event simulation. In *Proceedings of the 28th Conference on Winter Simulation*, pages 398–405, 1996.

Stephen E. Chick and Koichiro Inoue. New two-stage and sequential procedures for selecting the best simulated system. *Operations Research*, 49(5):732–743, 2001.

Stephen E. Chick, Jurgen Branke, and Christian Schmidt. Sequential sampling to myopically maximize the expected value of information. *INFORMS Journal on Computing*, 22(1):71–80, 2010.

Eyal Even-Dar, Shie Mannor, and Yishay Mansour. PAC bounds for multi-armed bandit and Markov decision processes. In *Proceedings of the 15th Annual Conference on Computational Learning Theory (COLT)*, pages 193–209, 2002.

Eyal Even-Dar, Shie Mannor, and Yishay Mansour. Action elimination and stopping conditions for reinforcement learning. In *Proceedings of the 20th International Conference on Machine Learning*, pages 162–169, 2003.

Abraham D. Flaxman, Adam Tauman Kalai, and H. Brendan McMahan. Online convex optimization in the bandit setting: gradient descent without a gradient. In *Proceedings of the Sixteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '05, pages 385–394, 2005.

Peter I. Frazier, Warren B. Powell, and Savas Dayanik. A knowledge-gradient policy for sequential information collection. *SIAM Journal on Control and Optimization*, 47(5):2410–2439, 2008.

Peter I. Frazier, Warren B. Powell, and Savas Dayanik. The knowledge-gradient policy for correlated normal beliefs. *INFORMS Journal on Computing*, 21(4):599–613, 2009.

Abraham George, Warren B. Powell, and Sanjeev R. Kulkarni. Value function approximation using multiple aggregation for multiattribute resource management. *Journal of Machine Learning Research*, 9:2079–2111, 2008.

Mark N. Gibbs. *Bayesian Gaussian Processes for Regression and Classification*. PhD thesis, University of Cambridge, 1997.

Steffen Grünewälder, Jean-Yves Audibert, Manfred Opper, and John Shawe-Taylor. Regret bounds for gaussian process bandit problems. In *Proceedings of the 14th International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2010.

Shanti S. Gupta and Klaus J. Miescke. Bayesian look ahead one-stage sampling allocations for selection of the best population. *Journal of Statistical Planning and Inference*, 54(2):229–244, 1996.

Trevor Hastie, Robert Tibshirani, and Jerome H. Friedman. *The Elements of Statistical Learning*. Springer series in Statistics, New York, NY, 2001.

Donghai He, Stephen E. Chick, and Chun-Hung Chen. Opportunity cost and OCBA selection procedures in ordinal optimization for a fixed number of alternative systems. *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews*, 37(5):951–961, 2007.

Deng Huang, Theodore T. Allen, William I. Notz, and Ning Zheng. Global optimization of stochastic black-box systems via sequential kriging meta-models. *Journal of Global Optimization*, 34 (3):441–466, 2006.

Frank Hutter. *Automated Configuration of Algorithms for Solving Hard Computational Problems*. PhD thesis, University of British Columbia, 2009.

Donald R. Jones, Matthias Schonlau, and William J. Welch. Efficient global optimization of expensive black-box functions. *Journal of Global Optimization*, 13(4):455–492, 1998.

Leslie P. Kaelbling. *Learning In Embedded Systems*. MIT Press, Cambridge, MA, 1993.

Michael Kearns and Satinder Singh. Near-optimal reinforcement learning in polynomial time. *Machine Learning*, 49(2-3):209–232, 2002.

Seong-Hee Kim and Barry L. Nelson. *Handbook in Operations Research and Management Science: Simulation*, chapter Selecting the best system. Elsevier, Amsterdam, 2006.

Robert Kleinberg, Aleksandrs Slivkins, and Eli Upfal. Multi-armed bandits in metric spaces. In *Proceedings of the 40th Annual ACM symposium on Theory of Computing*, pages 681–690, 2008.

Robert D. Kleinberg. *Online Decision Problems With Large Strategy Sets*. PhD thesis, MIT, 2005.

Harold J. Kushner. A new method of locating the maximum of an arbitrary multipeak curve in the presence of noise. *Journal of Basic Engineering*, 86:97–106, 1964.

Tze L. Lai. Adaptive treatment allocation and the multi-armed bandit problem. *The Annals of Statistics*, 15(3):1091–1114, 1987.

Michael LeBlanc and Robert Tibshirani. Combining estimates in regression and classification. *Journal of the American Statistical Association*, 91(436):1641–1650, 1996.

Nick Littlestone. From on-line to batch learning. In *Proceedings of the Second Annual Workshop on Computational learning theory*, pages 269–284, 1989.

Daniel J. Lizotte. *Practical Bayesian Optimization*. PhD thesis, University of Alberta, 2008.

Omid Madani, Daniel J. Lizotte, and Russell Greiner. The budgeted multi-armed bandit problem. In *Proceedings of the 17th Annual Conference on Computational Learning Theory (COLT)*, pages 643–645, 2004.

Volodymyr Mnih, Csaba Szepesvári, and Jean-Yves Audibert. Empirical Bernstein stopping. In *Proceedings of the 25th International Conference on Machine Learning*, pages 672–679, 2008.

Jonas Mockus. On Bayesian methods for seeking the extremum. In G. Marchuk, editor, *Optimization Techniques IFIP Technical Conference Novosibirsk, July 17, 1974*, volume 27 of *Lecture Notes in Computer Science*, pages 400–404. Springer Berlin / Heidelberg, 1975.

Warren B. Powell and Peter I. Frazier. Optimal learning. In *TutORials in Operations Research*, pages 213–246. INFORMS, 2008.

Carl E. Rasmussen and Christopher Williams. *Gaussian Processes for Machine Learning*. MIT Press, 2006.

Herbert Robbins and Sutton Monro. A stochastic approximation method. *Annals of Mathematical Statistics*, 22(3):400–407, 1951.

David F. Rogers, Robert D. Plante, Richard T. Wong, and James R. Evans. Aggregation and disaggregation techniques and methodology in optimization. *Operations Research*, 39(4):553–582, 1991.

Michael J. Sasena. *Flexibility and Efficiency Enhancements for Constrained Global Design Optimization with Kriging Approximations*. PhD thesis, University of Michigan, 2002.

Shai Shalev-Shwartz. *Online learning: Theory, algorithms, and applications*. PhD thesis, The Hebrew University of Jerusalem, 2007.

Hugo P. Simao, Jeff Day, Abraham P. George, Ted Gifford, John Nienow, and Warren B. Powell. An approximate dynamic programming algorithm for large-scale fleet management: A case application. *Transportation Science*, 43(2):178–197, 2009.

Tom A.B. Snijders and Roel J. Bosker. *Multilevel Analysis: An Introduction To Basic And Advanced Multilevel Modeling*. Sage Publications Ltd, 1999.

James C. Spall. *Introduction to Stochastic Search and Optimization*. Wiley-Interscience, Hoboken, NJ, 2003.

Niranjan Srinivas, Andreas Krause, Sham M. Kakade, and Matthias Seeger. Gaussian process optimization in the bandit setting: No regret and experimental design. In *Proceedings International Conference on Machine Learning (ICML)*, 2010.

Emmanuel Vazquez and Julien Bect. Convergence properties of the expected improvement algorithm with fixed mean and covariance functions. *Journal of Statistical Planning and Inference*, 140(11):3088–3095, 2010.

Julien Villemonteix, Emmanuel Vazquez, and Eric Walter. An informational approach to the global optimization of expensive-to-evaluate functions. *Journal of Global Optimization*, 44(4):509–534, 2009.

Yuhong Yang. Adaptive regression by mixing. *Journal of American Statistical Association*, 96 (454):574–588, 2001.

Journal of Machine Learning Research 12 (2011) 2975-3026 Submitted 9/10; Revised 6/11; Published 10/11

High-dimensional Covariance Estimation Based On Gaussian Graphical Models

Shuheng Zhou SHUHENGZ@UMICH.EDU
Department of Statistics
University of Michigan
Ann Arbor, MI 48109-1041, USA

Philipp Rütimann RUTIMANN@STAT.MATH.ETHZ.CH
Seminar for Statistics
ETH Zürich
8092 Zürich, Switzerland

Min Xu MINX@CS.CMU.EDU
Machine Learning Department
Carnegie Mellon University
Pittsburgh, PA 15213-3815, USA

Peter Bühlmann BUHLMANN@STAT.MATH.ETHZ.CH
Seminar for Statistics
ETH Zürich
8092 Zürich, Switzerland

Editor: Hui Zou

Abstract

Undirected graphs are often used to describe high dimensional distributions. Under sparsity conditions, the graph can be estimated using ℓ_1-penalization methods. We propose and study the following method. We combine a multiple regression approach with ideas of thresholding and refitting: first we infer a sparse undirected graphical model structure via thresholding of each among many ℓ_1-norm penalized regression functions; we then estimate the covariance matrix and its inverse using the maximum likelihood estimator. We show that under suitable conditions, this approach yields consistent estimation in terms of graphical structure and fast convergence rates with respect to the operator and Frobenius norm for the covariance matrix and its inverse. We also derive an explicit bound for the Kullback Leibler divergence.

Keywords: graphical model selection, covariance estimation, Lasso, nodewise regression, thresholding

1. Introduction

There have been a lot of recent activities for estimation of high-dimensional covariance and inverse covariance matrices where the dimension p of the matrix may greatly exceed the sample size n. High-dimensional covariance estimation can be classified into two main categories, one which relies on a natural ordering among the variables [Wu and Pourahmadi, 2003; Bickel and Levina, 2004; Huang et al., 2006; Furrer and Bengtsson, 2007; Bickel and Levina, 2008; Levina et al., 2008] and one where no natural ordering is given and estimators are permutation invariant with respect to indexing the variables [Yuan and Lin, 2007; Friedman et al., 2007; d'Aspremont et al., 2008; Banerjee et al., 2008; Rothman et al., 2008]. We focus here on the latter class with permutation invariant estimation and we aim for an estimator which is accurate for both the covariance matrix Σ and its inverse, the precision matrix Σ^{-1}. A popular approach for obtaining a permutation invariant estimator which is sparse in the estimated precision matrix $\widehat{\Sigma}^{-1}$ is given by the ℓ_1-norm regularized maximum-likelihood estimation, also known as the GLasso [Yuan and Lin, 2007; Friedman et al., 2007; Banerjee et al., 2008]. The GLasso approach is simple to use, at least when relying on publicly available software such as the `glasso` package in `R`. Further improvements have been reported when using some SCAD-type penalized maximum-likelihood estimator [Lam and Fan, 2009] or an adaptive GLasso procedure [Fan et al., 2009], which can be thought of as a two-stage procedure. It is well-known from linear regression that such two- or multi-stage methods effectively address some bias problems which arise from ℓ_1-penalization [Zou, 2006; Candès and Tao, 2007; Meinshausen, 2007; Zou and Li, 2008; Bühlmann and Meier, 2008; Zhou, 2009, 2010a].

In this paper we develop a new method for estimating graphical structure and parameters for multivariate Gaussian distributions using a multi-step procedure, which we call G**e**lat**o** (Graph **e**stimation with Lasso and Thresh**o**lding). Based on an ℓ_1-norm regularization and thresholding method in a first stage, we infer a sparse undirected graphical model, that is, an estimated Gaussian conditional independence graph, and we then perform unpenalized maximum likelihood estimation (MLE) for the covariance Σ and its inverse Σ^{-1} based on the estimated graph. We make the following theoretical contributions: (i) Our method allows us to select a graphical structure which is sparse. In some sense we select only the important edges even though there may be many non-zero edges in the graph. (ii) Secondly, we evaluate the quality of the graph we have selected by showing consistency and establishing a fast rate of convergence with respect to the operator and Frobenius norm for the estimated inverse covariance matrix; under sparsity constraints, the latter is of lower order than the corresponding results for the GLasso [Rothman et al., 2008] and for the SCAD-type estimator [Lam and Fan, 2009]. (iii) We show predictive risk consistency and provide a rate of convergence of the estimated covariance matrix. (iv) Lastly, we show general results for the MLE, where only *approximate* graph structures are given as input. Besides these theoretical advantages, we found empirically that our graph based method performs better in general, and sometimes substantially better than the GLasso, while we never found it clearly worse. Moreover, we compare it with an adaptation of the method Space [Peng et al., 2009]. Finally, our algorithm is simple and is comparable to the GLasso both in terms of computational time and implementation complexity.

There are a few key motivations and consequences for proposing such an approach based on graphical modeling. We will theoretically show that there are cases where our graph based method can accurately estimate conditional independencies among variables, that is, the zeroes of Σ^{-1}, in situations where GLasso fails. The fact that GLasso easily fails to estimate the zeroes of Σ^{-1} has been recognized by Meinshausen [2008] and it has been discussed in more details in Ravikumar et al. [2011]. Closer relations to existing work are primarily regarding our first stage of estimating the structure of the graph. We follow the nodewise regression approach from Meinshausen and Bühlmann [2006] but we make use of recent results for variable selection in linear models assuming the much weaker restricted eigenvalue condition [Bickel et al., 2009; Zhou, 2010a] instead of the restrictive neighborhood stability condition [Meinshausen and Bühlmann, 2006] or the equivalent irrepresentable condition [Zhao and Yu, 2006]. In some sense, the novelty of our theory extending beyond Zhou [2010a] is the analysis for covariance and inverse covariance estimation and for risk consistency based on an estimated sparse graph as we mentioned above. Our regression and thresholding results build upon analysis of the thresholded Lasso estimator as studied in Zhou [2010a]. Throughout our analysis, the sample complexity is one of the key focus point, which builds upon results in Zhou [2010b]; Rudelson and Zhou [2011]. Once the zeros are found, a constrained maximum likelihood estimator of the covariance can be computed, which was shown in Chaudhuri et al. [2007]; it was unclear what the properties of such a procedure would be. Our theory answers such questions. As a two-stage method, our approach is also related to the adaptive Lasso [Zou, 2006] which has been analyzed for high-dimensional scenarios in Huang et al. [2008], Zhou et al. [2009] and van de Geer et al. [2011]. Another relation can be made to the method by Rütimann and Bühlmann [2009] for covariance and inverse covariance estimation based on a directed acyclic graph. This relation has only methodological character: the techniques and algorithms used in Rütimann and Bühlmann [2009] are very different and from a practical point of view, their approach has much higher degree of complexity in terms of computation and implementation, since estimation of an equivalence class of directed acyclic graphs is difficult and cumbersome. There has also been work that focuses on estimation of sparse directed Gaussian graphical model. Verzelen [2010] proposes a multiple regularized regression procedure for estimating a precision matrix with sparse Cholesky factors, which correspond to a sparse directed graph. He also computes nonasymptotic Kullback Leibler risk bound of his procedure for a class of regularization functions. It is important to note that directed graph estimation requires a fixed good ordering of the variables a priori.

1.1 Notation

We use the following notation. Given a graph $G = (V, E_0)$, where $V = \{1, \ldots, p\}$ is the set of vertices and E_0 is the set of undirected edges. we use s^i to denote the degree for node i, that is, the number of edges in E_0 connecting to node i. For an edge set E, we let $|E|$ denote its size. We use $\Theta_0 = \Sigma_0^{-1}$ and Σ_0 to refer to the true precision and covariance matrices respectively from now on. We denote the number of non-zero elements of Θ by $\mathrm{supp}(\Theta)$. For any matrix $W = (w_{ij})$, let $|W|$ denote the

determinant of W, $\mathrm{tr}(W)$ the trace of W. Let $\varphi_{\max}(W)$ and $\varphi_{\min}(W)$ be the largest and smallest eigenvalues, respectively. We write $\mathrm{diag}(W)$ for a diagonal matrix with the same diagonal as W and $\mathrm{offd}(W) = W - \mathrm{diag}(W)$. The matrix Frobenius norm is given by $\|W\|_F = \sqrt{\sum_i \sum_j w_{ij}^2}$. The operator norm $\|W\|_2^2$ is given by $\varphi_{\max}(WW^T)$. We write $|\cdot|_1$ for the ℓ_1 norm of a matrix vectorized, that is, for a matrix $|W|_1 = \|\mathrm{vec}W\|_1 = \sum_i \sum_j |w_{ij}|$, and sometimes write $\|W\|_0$ for the number of non-zero entries in the matrix. For an index set T and a matrix $W = [w_{ij}]$, write $W_T \equiv (w_{ij} I((i,j) \in T))$, where $I(\cdot)$ is the indicator function.

2. The Model and the Method

We assume a multivariate Gaussian model

$$X = (X_1, \ldots, X_p) \sim \mathcal{N}_p(0, \Sigma_0), \quad \text{where } \Sigma_{0,ii} = 1. \tag{1}$$

The data is generated by $X^{(1)}, \ldots, X^{(n)}$ i.i.d. $\sim \mathcal{N}_p(0, \Sigma_0)$. Requiring the mean vector and all variances being equal to zero and one respectively is not a real restriction and in practice, we can easily center and scale the data. We denote the concentration matrix by $\Theta_0 = \Sigma_0^{-1}$.

Since we will use a nodewise regression procedure, as described below in Section 2.1, we consider a regression formulation of the model. Consider many regressions, where we regress one variable against all others:

$$X_i = \sum_{j \neq i} \beta_j^i X_j + V_i \ (i = 1, \ldots, p), \quad \text{where} \tag{2}$$

$$V_i \sim \mathcal{N}(0, \sigma_{V_i}^2) \text{ independent of } \{X_j; j \neq i\} \ (i = 1, \ldots, p). \tag{3}$$

There are explicit relations between the regression coefficients, error variances and the concentration matrix $\Theta_0 = (\theta_{0,ij})$:

$$\beta_j^i = -\theta_{0,ij}/\theta_{0,ii}, \ \mathrm{Var}(V_i) := \sigma_{V_i}^2 = 1/\theta_{0,ii} \ (i, j = 1, \ldots, p). \tag{4}$$

Furthermore, it is well known that for Gaussian distributions, conditional independence is encoded in Θ_0, and due to (4), also in the regression coefficients:

$$\begin{aligned} & X_i \text{ is conditionally dependent of } X_j \text{ given } \{X_k;\ k \in \{1, \ldots, p\} \setminus \{i, j\}\} \\ \iff \quad & \theta_{0,ij} \neq 0 \iff \beta_i^j \neq 0 \text{ and } \beta_j^i \neq 0. \end{aligned} \tag{5}$$

For the second equivalence, we assume that $\mathrm{Var}(V_i) = 1/\theta_{0,ii} > 0$ and $\mathrm{Var}(V_j) = 1/\theta_{0,jj} > 0$. Conditional (in-)dependencies can be conveniently encoded by an undirected graph, the conditional independence graph which we denote by $G = (V, E_0)$. The set of vertices is $V = \{1, \ldots, p\}$ and the set of undirected edges $E_0 \subseteq V \times V$ is defined as follows:

$$\begin{aligned} & \text{there is an undirected edge between nodes } i \text{ and } j \\ \iff \quad & \theta_{0,ij} \neq 0 \iff \beta_i^j \neq 0 \text{ and } \beta_j^i \neq 0. \end{aligned} \tag{6}$$

Note that on the right hand side of the second equivalence, we could replace the word "and" by "or". For the second equivalence, we assume $\mathrm{Var}(V_i), \mathrm{Var}(V_j) > 0$ following the remark after (5).

We now define the sparsity of the concentration matrix Θ_0 or the conditional independence graph. The definition is different than simply counting the non-zero elements of Θ_0, for which we have $\mathrm{supp}(\Theta_0) = p + 2|E_0|$. We consider instead the number of elements which are sufficiently large. For each i, define the number $s^i_{0,n}$ as the smallest integer such that the following holds:

$$\sum_{j=1, j\neq i}^{p} \min\{\theta^2_{0,ij}, \lambda^2 \theta_{0,ii}\} \quad \leq \quad s^i_{0,n} \lambda^2 \theta_{0,ii}, \text{ where } \lambda = \sqrt{2\log(p)/n}, \tag{7}$$

where *essential sparsity* $s^i_{0,n}$ at row i describes the number of "sufficiently large" non-diagonal elements $\theta_{0,ij}$ relative to a given (n,p) pair and $\theta_{0,ii}, i = 1,\ldots,p$. The value $S_{0,n}$ in (8) is summing *essential sparsity* across all rows of Θ_0,

$$S_{0,n} \quad := \quad \sum_{i=1}^{p} s^i_{0,n}. \tag{8}$$

Due to the expression of λ, the value of $S_{0,n}$ depends on p and n. For example, if all non-zero non-diagonal elements $\theta_{0,ij}$ of the ith row are larger in absolute value than $\lambda\sqrt{\theta_{0,ii}}$, the value $s^i_{0,n}$ coincides with the node degree s^i. However, if some (many) of the elements $|\theta_{0,ij}|$ are non-zero but small, $s^i_{0,n}$ is (much) smaller than its node degree s^i; As a consequence, if some (many) of $|\theta_{0,ij}|, \forall i,j, i \neq j$ are non-zero but small, the value of $S_{0,n}$ is also (much) smaller than $2|E_0|$, which is the "classical" sparsity for the matrix $(\Theta_0 - \mathrm{diag}(\Theta_0))$. See Section A for more discussions.

2.1 The Estimation Procedure

The estimation of Θ_0 and $\Sigma_0 = \Theta_0^{-1}$ is pursued in two stages. We first estimate the undirected graph with edge set E_0 as in (6) and we then use the maximum likelihood estimator based on the estimate $\widehat{E}_n$, that is, the non-zero elements of $\widehat{\Theta}_n$ correspond to the estimated edges in $\widehat{E}_n$. Inferring the edge set E_0 can be based on the following approach as proposed and theoretically justified in Meinshausen and Bühlmann [2006]: perform p regressions using the Lasso to obtain p vectors of regression coefficients $\widehat{\beta}^1, \ldots, \widehat{\beta}^p$ where for each i, $\widehat{\beta}^i = \{\widehat{\beta}^i_j;\ j \in \{1,\ldots,p\} \setminus i\}$; Then estimate the edge set by the "OR" rule,

$$\text{estimate an edge between nodes } i \text{ and } j \Longleftrightarrow \widehat{\beta}^i_j \neq 0 \text{ or } \widehat{\beta}^j_i \neq 0. \tag{9}$$

2.1.1 NODEWISE REGRESSIONS FOR INFERRING THE GRAPH

In the present work, we use the Lasso in combination with thresholding [Zhou, 2009, 2010a]. Consider the Lasso for each of the nodewise regressions

$$\beta^i_{\mathrm{init}} = \mathrm{argmin}_{\beta^i} \sum_{r=1}^{n} (X^{(r)}_i - \sum_{j\neq i} \beta^i_j X^{(r)}_j)^2 + \lambda_n \sum_{j\neq i} |\beta^i_j| \quad \text{for } i = 1,\ldots,p, \tag{10}$$

where $\lambda_n > 0$ is the same regularization parameter for all regressions. Since the Lasso typically estimates too many components with non-zero estimated regression coefficients, we use thresholding to get rid of variables with small regression coefficients from solutions of (10):

$$\widehat{\beta}^i_j(\lambda_n, \tau) = \beta^i_{j,\text{init}}(\lambda_n) I(|\beta^i_{j,\text{init}}(\lambda_n)| > \tau), \tag{11}$$

where $\tau > 0$ is a thresholding parameter. We obtain the corresponding estimated edge set as defined by (9) using the estimator in (11) and we use the notation

$$\widehat{E}_n(\lambda_n, \tau). \tag{12}$$

We note that the estimator depends on two tuning parameters λ_n and τ.

The use of thresholding has clear benefits from a theoretical point of view: the number of false positive selections may be much larger without thresholding (when tuned for good prediction). and a similar statement would hold when comparing the adaptive Lasso with the standard Lasso. We refer the interested reader to Zhou [2009, 2010a] and van de Geer et al. [2011].

2.1.2 MAXIMUM LIKELIHOOD ESTIMATION BASED ON GRAPHS

Given a conditional independence graph with edge set E, we estimate the concentration matrix by maximum likelihood. Denote by $\widehat{S}_n = n^{-1}\sum_{r=1}^n X^{(r)}(X^{(r)})^T$ the sample covariance matrix (using that the mean vector is zero) and by

$$\widehat{\Gamma}_n = \text{diag}(\widehat{S}_n)^{-1/2}(\widehat{S}_n)\text{diag}(\widehat{S}_n)^{-1/2}$$

the sample correlation matrix. The estimator for the concentration matrix in view of (1) is:

$$\begin{aligned}
&\widehat{\Theta}_n(E) = \text{argmin}_{\Theta \in \mathcal{M}_{p,E}} \left(\text{tr}(\Theta\widehat{\Gamma}_n) - \log|\Theta| \right), \text{ where} \\
&\mathcal{M}_{p,E} = \{\Theta \in \mathbb{R}^{p\times p};\ \Theta \succ 0 \text{ and } \theta_{ij} = 0 \text{ for all } (i,j) \notin E, \text{ where } i \neq j\}
\end{aligned} \tag{13}$$

defines the constrained set for positive definite Θ. If $n \geq q^*$ where q^* is the maximal clique size of a minimal chordal cover of the graph with edge set E, the MLE exists and is unique, see, for example Uhler [2011, Corollary 2.3]. We note that our theory guarantees that $n \geq q^*$ holds with high probability for $G = (V, E)$, where $E = \widehat{E}_n(\lambda_n, \tau))$, under Assumption (A1) to be introduced in the next section. The definition in (13) is slightly different from the more usual estimator which uses the sample covariance $\widehat{S}_n$ rather than $\widehat{\Gamma}_n$. Here, the sample correlation matrix reflects the fact that we typically work with standardized data where the variables have empirical variances equal to one. The estimator in (13) is constrained leading to zero-values corresponding to $E^c = \{(i,j) : i,j = 1,\ldots,p, i \neq j, (i,j) \notin E\}$.

If the edge set E is sparse having relatively few edges only, the estimator in (13) is already sufficiently regularized by the constraints and hence, no additional penalization is used at this stage. Our final estimator for the concentration matrix is the combination of (12) and (13):

$$\widehat{\Theta}_n = \widehat{\Theta}_n(\widehat{E}_n(\lambda_n, \tau)). \tag{14}$$

2.1.3 CHOOSING THE REGULARIZATION PARAMETERS

We propose to select the parameter λ_n via cross-validation to minimize the squared test set error among all p regressions:

$$\widehat{\lambda}_n = \text{argmin}_\lambda \sum_{i=1}^{p} (\text{CV-score}(\lambda) \text{ of } i\text{th regression}),$$

where CV-score(λ) of ith regression is with respect to the squared error prediction loss. Sequentially proceeding, we then select τ by cross-validating the multivariate Gaussian log-likelihood, from (13). Regarding the type of cross-validation, we usually use the 10-fold scheme. Due to the sequential nature of choosing the regularization parameters, the number of candidate estimators is given by the number of candidate values for λ plus the number of candidate value for τ. In Section 4, we describe the grids of candidate values in more details. We note that for our theoretical results, we do not analyze the implications of our method using estimated $\widehat{\lambda}_n$ and $\widehat{\tau}$.

3. Theoretical Results

In this section, we present in Theorem 1 convergence rates for estimating the precision and the covariance matrices with respect to the Frobenius norm; in addition, we show a risk consistency result for an oracle risk to be defined in (16). Moreover, in Proposition 4, we show that the model we select is sufficiently sparse while at the same time, the bias term we introduce via sparse approximation is sufficiently bounded. These results illustrate the classical bias and variance tradeoff. Our analysis is non-asymptotic in nature; however, we first formulate our results from an asymptotic point of view for simplicity. To do so, we consider a triangular array of data generating random variables

$$X^{(1)}, \ldots, X^{(n)} \text{ i.i.d.} \sim \mathcal{N}_p(0, \Sigma_0), \; n = 1, 2, \ldots \tag{15}$$

where $\Sigma_0 = \Sigma_{0,n}$ and $p = p_n$ change with n. Let $\Theta_0 := \Sigma_0^{-1}$. We make the following assumptions.

(A0) The size of the neighborhood for each node $i \in V$ is upper bounded by an integer $s < p$ and the sample size satisfies for some constant C

$$n \geq Cs \log(p/s).$$

(A1) The dimension and number of sufficiently strong non-zero edges $S_{0,n}$ as in (8) satisfy: dimension p grows with n following $p = o(e^{cn})$ for some constant $0 < c < 1$ and

$$S_{0,n} = o(n/\log \max(n, p)) \; (n \to \infty).$$

(A2) The minimal and maximal eigenvalues of the true covariance matrix Σ_0 are bounded: for some constants $M_{\text{upp}} \geq M_{\text{low}} > 0$, we have

$$\varphi_{\min}(\Sigma_0) \geq M_{\text{low}} > 0 \text{ and } \varphi_{\max}(\Sigma_0) \leq M_{\text{upp}} < \infty.$$

Moreover, throughout our analysis, we assume the following. There exists $v^2 > 0$ such that for all i, and V_i as defined in (3): $\mathrm{Var}(V_i) = 1/\theta_{0,ii} \geq v^2$.

Before we proceed, we need some definitions. Define for $\Theta \succ 0$

$$R(\Theta) = \mathrm{tr}(\Theta\Sigma_0) - \log|\Theta|, \tag{16}$$

where minimizing (16) without constraints gives Θ_0. Given (8), (7), and Θ_0, define

$$C^2_{\mathrm{diag}} := \min\{\max_{i=1,\ldots p} \theta^2_{0,ii}, \max_{i=1,\ldots,p} \left(s^i_{0,n}/S_{0,n}\right) \cdot \|\mathrm{diag}(\Theta_0)\|^2_F\}. \tag{17}$$

We now state the main results of this paper. We defer the specification on various tuning parameters, namely, λ_n, τ to Section 3.2, where we also provide an outline for Theorem 1.

Theorem 1 *Consider data generating random variables as in (15) and assume that (A0), (A1), and (A2) hold. We assume $\Sigma_{0,ii} = 1$ for all i. Then, with probability at least $1 - d/p^2$, for some small constant $d > 2$, we obtain under appropriately chosen λ_n and τ, an edge set $\widehat{E}_n$ as in (12), such that*

$$|\widehat{E}_n| \leq 2S_{0,n}, \;\; \textit{where} \;\; |\widehat{E}_n \setminus E_0| \leq S_{0,n}; \tag{18}$$

and for $\widehat{\Theta}_n$ and $\widehat{\Sigma}_n = (\widehat{\Theta}_n)^{-1}$ as defined in (14)*, the following holds,*

$$\begin{aligned}
\left\|\widehat{\Theta}_n - \Theta_0\right\|_2 \leq \|\widehat{\Theta}_n - \Theta_0\|_F &= O_P\left(\sqrt{S_{0,n}\log\max(n,p)/n}\right), \\
\left\|\widehat{\Sigma}_n - \Sigma_0\right\|_2 \leq \|\widehat{\Sigma}_n - \Sigma_0\|_F &= O_P\left(\sqrt{S_{0,n}\log\max(n,p)/n}\right), \\
R(\widehat{\Theta}_n) - R(\Theta_0) &= O_P\left(S_{0,n}\log\max(n,p)/n\right),
\end{aligned}$$

where the constants hidden in the $O_P()$ notation depend on τ, $M_{\mathrm{low}}, M_{\mathrm{upp}}$, C_{diag} as in (17)*, and constants concerning sparse and restrictive eigenvalues of Σ_0 (cf. Section 3.2 and B).*

We note that convergence rates for the estimated covariance matrix and for predictive risk depend on the rate in Frobenius norm of the estimated inverse covariance matrix. The predictive risk can be interpreted as follows. Let $X \sim \mathcal{N}(0, \Sigma_0)$ with f_{Σ_0} denoting its density. Let $f_{\widehat{\Sigma}_n}$ be the density for $\mathcal{N}(0, \widehat{\Sigma}_n)$ and $D_{\mathrm{KL}}(\Sigma_0\|\widehat{\Sigma}_n)$ denotes the Kullback Leibler (KL) divergence from $\mathcal{N}(0, \Sigma_0)$ to $\mathcal{N}(0, \widehat{\Sigma}_n)$. Now, we have for $\Sigma, \widehat{\Sigma}_n \succ 0$,

$$R(\widehat{\Theta}_n) - R(\Theta_0) := 2\mathbf{E}_0\left[\log f_{\Sigma_0}(X) - \log f_{\widehat{\Sigma}_n}(X)\right] := 2D_{\mathrm{KL}}(\Sigma_0\|\widehat{\Sigma}_n) \geq 0. \tag{19}$$

Actual conditions and non-asymptotic results that are involved in the **Gelato** estimation appear in Sections B, C, and D respectively.

Remark 2 *Implicitly in (A1), we have specified a lower bound on the sample size to be $n = \Omega\left(S_{0,n}\log\max(n,p)\right)$. For the interesting case of $p > n$, a sample size of*

$$n = \Omega\left(\max(S_{0,n}\log p, s\log(p/s))\right)$$

is sufficient in order to achieve the rates in Theorem 1. As to be shown in our analysis, the lower bound on n is slightly different for each Frobenius norm bound to hold from a non-asymptotic point of view (cf. Theorem 19 and 20).

Theorem 1 can be interpreted as follows. First, the cardinality of the estimated edge set exceeds $S_{0,n}$ at most by a factor 2, where $S_{0,n}$ as in (8) is the number of sufficiently strong edges in the model, while the number of false positives is bounded by $S_{0,n}$. Note that the factors 2 and 1 can be replaced by some other constants, while achieving the same bounds on the rates of convergence (cf. Section D.1). We emphasize that we achieve these two goals by sparse model selection, where only important edges are selected even though there are many more non-zero edges in E_0, under conditions that are much weaker than (A2). More precisely, (A2) can be replaced by conditions on sparse and restrictive eigenvalues (RE) of Σ_0. Moreover, the bounded neighborhood constraint (A0) is required only for regression analysis (cf. Theorem 15) and for bounding the bias due to sparse approximation as in Proposition 4. This is shown in Sections B and C. Analysis follows from Zhou [2009, 2010a] with earlier references to Candès and Tao [2007], Meinshausen and Yu [2009] and Bickel et al. [2009] for estimating sparse regression coefficients.

We note that the conditions that we use are indeed similar to those in Rothman et al. [2008], with (A1) being much more relaxed when $S_{0,n} \ll |E_0|$. The convergence rate with respect to the Frobenius norm should be compared to the rate $O_P(\sqrt{|E_0| \log \max(n,p)/n})$ in case $\mathrm{diag}(\Sigma_0)$ is known, which is the rate in Rothman et al. [2008] for the GLasso and for SCAD [Lam and Fan, 2009]. In the scenario where $|E_0| \gg S_{0,n}$, that is, there are many weak edges, the rate in Theorem 1 is better than the one established for GLasso [Rothman et al., 2008] or for the SCAD-type estimator [Lam and Fan, 2009]; hence we require a smaller sample size in order to yield an accurate estimate of Θ_0.

Remark 3 *For the general case where $\Sigma_{0,ii}, i = 1, \ldots, p$ are not assumed to be known, we could achieve essentially the same rate as stated in Theorem 1 for $\|\widehat{\Theta}_n - \Theta_0\|_2$ and $\|\widehat{\Sigma}_n - \Sigma_0\|_2$ under $(A_0), (A_1)$ and (A_2) following analysis in the present work (cf. Theorem 6) and that in Rothman et al. [2008, Theorem 2]. Presenting full details for such results are beyond the scope of the current paper. We do provide the key technical lemma which is essential for showing such bounds based on estimating the inverse of the correlation matrix in Theorem 6; see also Remark 7 which immediately follows.*

In this case, for the Frobenius norm and the risk to converge to zero, a too large value of p is not allowed. Indeed, for the Frobenius norm and the risk to converge, $(A1)$ is to be replaced by:

$(A3)$ $p \asymp n^c$ *for some constant* $0 < c < 1$ *and* $p + S_{0,n} = o(n/\log \max(n,p))$ *as* $n \to \infty$.

In this case, we have

$$\begin{aligned} \|\widehat{\Theta}_n - \Theta_0\|_F &= O_P\left(\sqrt{(p + S_{0,n}) \log \max(n,p)/n}\right), \\ \|\widehat{\Sigma}_n - \Sigma_0\|_F &= O_P\left(\sqrt{(p + S_{0,n}) \log \max(n,p)/n}\right), \\ R(\widehat{\Theta}_n) - R(\Theta_0) &= O_P\left((p + S_{0,n}) \log \max(n,p)/n\right). \end{aligned}$$

Moreover, in the refitting stage, we could achieve these rates with the maximum likelihood estimator based on the sample covariance matrix $\widehat{S}_n$ as defined in (20)*:*

$$\widehat{\Theta}_n(E) = argmin_{\Theta \in \mathcal{M}_{p,E}} \left(\mathrm{tr}(\Theta \widehat{S}_n) - \log|\Theta| \right), \text{ where}$$
$$\mathcal{M}_{p,E} = \{\Theta \in \mathbb{R}^{p \times p};\ \Theta \succ 0 \text{ and } \theta_{ij} = 0 \text{ for all } (i,j) \notin E, \text{ where } i \neq j\}. \tag{20}$$

A real high-dimensional scenario where $p \gg n$ is excluded in order to achieve Frobenius norm consistency. This restriction comes from the nature of the Frobenius norm and when considering, for example, the operator norm, such restrictions can indeed be relaxed as stated above.

It is also of interest to understand the bias of the estimator caused by using the estimated edge set $\widehat{E}_n$ instead of the true edge set E_0. This is the content of Proposition 4. For a given $\widehat{E}_n$, denote by

$$\widetilde{\Theta}_0 = \mathrm{diag}(\Theta_0) + (\Theta_0)_{\widehat{E}_n} = \mathrm{diag}(\Theta_0) + \Theta_{0,\widehat{E}_n \cap E_0},$$

where the second equality holds since $\Theta_{0,E_0^c} = 0$. Note that the quantity $\widetilde{\Theta}_0$ is identical to Θ_0 on $\widehat{E}_n$ and on the diagonal, and it equals zero on $\widehat{E}_n^c = \{(i,j) : i,j = 1,\ldots,p, i \neq j, (i,j) \notin \widehat{E}_n\}$. Hence, the quantity $\Theta_{0,\mathcal{D}} := \widetilde{\Theta}_0 - \Theta_0$ measures the bias caused by a potentially wrong edge set $\widehat{E}_n$; note that $\widetilde{\Theta}_0 = \Theta_0$ if $\widehat{E}_n = E_0$.

Proposition 4 *Consider data generating random variables as in expression (15). Assume that (A0), (A1), and (A2) hold. Then we have for choices on λ_n, τ as in Theorem 1 and $\widehat{E}_n$ in (12),*

$$\left\|\Theta_{0,\mathcal{D}}\right\|_F := \|\widetilde{\Theta}_0 - \Theta_0\|_F = O_P\left(\sqrt{S_{0,n} \log \max(n,p)/n}\right).$$

We note that we achieve essentially the same rate for $\|(\widetilde{\Theta}_0)^{-1} - \Sigma_0\|_F$; see Remark 27. We give an account on how results in Proposition 4 are obtained in Section 3.2, with its non-asymptotic statement appearing in Corollary 17.

3.1 Discussions and Connections to Previous Work

It is worth mentioning that consistency in terms of operator and Frobenius norms does not depend too strongly on the property to recover the true underlying edge set E_0 in the refitting stage. Regarding the latter, suppose we obtain with high probability the screening property

$$E_0 \subseteq E, \tag{21}$$

when assuming that all non-zero regression coefficients $|\beta_j^i|$ are sufficiently large (E might be an estimate and hence random). Although we do not intend to make precise the exact conditions and choices of tuning parameters in regression and thresholding in order to achieve (21), we state Theorem 5, in case (21) holds with the following condition: the number of false positives is bounded as $|E \setminus E_0| = O(S)$.

Theorem 5 *Consider data generating random variables as in expression (15) and assume that (A1) and (A2) hold, where we replace $S_{0,n}$ with $S := |E_0| = \sum_{i=1}^{p} s^i$. We assume $\Sigma_{0,ii} = 1$ for all i. Suppose on some event $\mathcal{E}$, such that $\mathbb{P}(\mathcal{E}) \geq 1 - d/p^2$ for a small constant d, we obtain an edge set E such that $E_0 \subseteq E$ and $|E \setminus E_0| = O(S)$. Let $\widehat{\Theta}_n(E)$ be the minimizer as defined in* (13)*. Then, we have* $\|\widehat{\Theta}_n(E) - \Theta_0\|_F = O_P\left(\sqrt{S \log \max(n,p)/n}\right)$.

It is clear that this bound corresponds to exactly that of Rothman et al. [2008] for the GLasso estimation under appropriate choice of the penalty parameter for a general $\Sigma \succ 0$ with $\Sigma_{ii} = 1$ for all i (cf. Remark 3). We omit the proof as it is more or less a modified version of Theorem 19, which proves the stronger bounds as stated in Theorem 1. We note that the maximum node-degree bound in (A0) is not needed for Theorem 5.

We now make some connections to previous work. First, we note that to obtain with high probability the exact edge recovery, $E = E_0$, we need again sufficiently large non-zero edge weights and some restricted eigenvalue (RE) conditions on the covariance matrix as defined in Section A even for the multi-stage procedure. An earlier example is shown in Zhou et al. [2009], where the second stage estimator $\widehat{\beta}$ corresponding to (11) is obtained with nodewise regressions using adaptive Lasso [Zou, 2006] rather than thresholding as in the present work in order to recover the edge set E_0 with high probability under an assumption which is stronger than (A0). Clearly, given an accurate $\widehat{E}_n$, under (A1) and (A2) one can then apply Theorem 5 to accurately estimate $\widehat{\Theta}_n$. On the other hand, it is known that GLasso necessarily needs more restrictive conditions on Σ_0 than the nodewise regression approach with the Lasso, as discussed in Meinshausen [2008] and Ravikumar et al. [2011] in order to achieve exact edge recovery.

Furthermore, we believe it is straightforward to show that Gelato works under the RE conditions on Σ_0 and with a smaller sample size than the analogue without the thresholding operation in order to achieve *nearly exact recovery* of the support in the sense that $E_0 \subseteq \widehat{E}_n$ and $\max_i |\widehat{E}_{n,i} \setminus E_{0,i}|$ is small, that is, the number of extra estimated edges at each node i is bounded by a small constant. This is shown essentially in Zhou [2009, Theorem1.1] for a single regression. Given such properties of $\widehat{E}_n$, we can again apply Theorem 5 to obtain $\widehat{\Theta}_n$ under (A1) and (A2). Therefore, Gelato requires relatively weak assumptions on Σ_0 in order to achieve the best sparsity and bias tradeoff as illustrated in Theorem 1 and Proposition 4 when many signals are weak, and Theorem 5 when all signals in E_0 are strong.

Finally, it would be interesting to derive a tighter bound on the operator norm for the Gelato estimator. Examples of such bounds have been recently derived for a restricted class of inverse covariance matrices in Yuan [2010] and Cai et al. [2011].

3.2 An Outline for Theorem 1

Let $s_0 = \max_{i=1,\ldots,p} s^i_{0,n}$. We note that although sparse eigenvalues $\rho_{\max}(s), \rho_{\max}(3s_0)$ and restricted eigenvalue for Σ_0 (cf. Section A) are parameters that are unknown, we only need them to appear in

the lower bounds for d_0, D_4, and hence also that for λ_n and t_0 that appear below. We simplify our notation in this section to keep it consistent with our theoretical non-asymptotic analysis to appear toward the end of this paper.

3.2.1 REGRESSION

We choose for some $c_0 \geq 4\sqrt{2}$, $0 < \theta < 1$, and $\lambda = \sqrt{2\log(p)/n}$,

$$\lambda_n = d_0\lambda, \text{ where } d_0 \geq c_0(1+\theta)^2\sqrt{\rho_{\max}(s)\rho_{\max}(3s_0)}.$$

Let $\beta^i_{\text{init}}, i = 1,\ldots,p$ be the optimal solutions to (10) with λ_n as chosen above. We first prove an oracle result on nodewise regressions in Theorem 15.

3.2.2 THRESHOLDING

We choose for some constants D_1, D_4 to be defined in Theorem 15,

$$t_0 = f_0\lambda := D_4 d_0\lambda \quad \text{where } D_4 \geq D_1$$

and D_1 depends on restrictive eigenvalue of Σ_0; Apply (11) with $\tau = t_0$ and $\beta^i_{\text{init}}, i = 1,\ldots,p$ for thresholding our initial regression coefficients. Let

$$\mathcal{D}^i = \{j : j \neq i, \ \left|\beta^i_{j,\text{init}}\right| < t_0 = f_0\lambda\},$$

where bounds on $\mathcal{D}^i, i = 1,\ldots,p$ are given in Lemma 16. In view of (9), we let

$$\mathcal{D} = \{(i,j) : i \neq j : (i,j) \in \mathcal{D}^i \cap \mathcal{D}^j\}. \tag{22}$$

3.2.3 SELECTING EDGE SET E

Recall for a pair (i,j) we take the *OR rule* as in (9) to decide if it is to be included in the edge set E: for $\mathcal{D}$ as defined in (22), define

$$E := \{(i,j) : i,j = 1,\ldots,p, i \neq j, (i,j) \notin \mathcal{D}\}. \tag{23}$$

to be the subset of pairs of non-identical vertices of G which do not appear in $\mathcal{D}$; Let

$$\widetilde{\Theta}_0 = \text{diag}(\Theta_0) + \Theta_{0,E_0\cap E} \tag{24}$$

for E as in (23), which is identical to Θ_0 on all diagonal entries and entries indexed by $E_0 \cap E$, with the rest being set to zero. As shown in the proof of Corollary 17, by thresholding, we have identified a *sparse subset* of edges E of size at most $4S_{0,n}$, such that the corresponding bias $\left\|\Theta_{0,\mathcal{D}}\right\|_F := \|\widetilde{\Theta}_0 - \Theta_0\|_F$ is relatively small, that is, as bounded in Proposition 4.

3.2.4 REFITTING

In view of Proposition 4, we aim to recover $\widetilde{\Theta}_0$ given a sparse subset E; toward this goal, we use (13) to obtain the final estimator $\widehat{\Theta}_n$ and $\widehat{\Sigma}_n = (\widehat{\Theta}_n)^{-1}$. We give a more detailed account of this procedure in Section D, with a focus on elaborating the bias and variance tradeoff. We show the rate of convergence in Frobenius norm for the estimated $\widehat{\Theta}_n$ and $\widehat{\Sigma}_n$ in Theorem 6, 19 and 20, and the bound for Kullback Leibler divergence in Theorem 21 respectively.

3.3 Discussion on Covariance Estimation Based on Maximum Likelihood

The maximum likelihood estimate minimizes over all $\Theta \succ 0$,

$$\widehat{R}_n(\Theta) = \mathrm{tr}(\Theta \widehat{S}_n) - \log|\Theta|, \tag{25}$$

where $\widehat{S}_n$ is the sample covariance matrix. Minimizing $\widehat{R}_n(\Theta)$ without constraints gives $\widehat{\Sigma}_n = \widehat{S}_n$. We now would like to minimize (25) under the constraints that some pre-defined subset $\mathcal{D}$ of edges are set to zero. Then the follow relationships hold regarding $\widehat{\Theta}_n(E)$ defined in (20) and its inverse $\widehat{\Sigma}_n$, and $\widehat{S}_n$: for E as defined in (23),

$$\begin{aligned} \widehat{\Theta}_{n,ij} &= 0,\ \forall (i,j) \in \mathcal{D},\ \text{and} \\ \widehat{\Sigma}_{n,ij} &= \widehat{S}_{n,ij},\ \forall (i,j) \in E \cup \{(i,i), i = 1,\ldots,p\}. \end{aligned}$$

Hence the entries in the covariance matrix $\widehat{\Sigma}_n$ for the chosen set of edges in E and the diagonal entries are set to their corresponding values in $\widehat{S}_n$. Indeed, we can derive the above relationships via the Lagrange form, where we add Lagrange constants γ_{jk} for edges in $\mathcal{D}$,

$$\ell_C(\Theta) = \log|\Theta| - \mathrm{tr}(\widehat{S}_n \Theta) - \sum_{(j,k)\in\mathcal{D}} \gamma_{jk}\theta_{jk}. \tag{26}$$

Now the gradient equation of (26) is:

$$\Theta^{-1} - \widehat{S}_n - \Gamma = 0,$$

where Γ is a matrix of Lagrange parameters such that $\gamma_{jk} \neq 0$ for all $(j,k) \in \mathcal{D}$ and $\gamma_{jk} = 0$ otherwise.

Similarly, the follow relationships hold regarding $\widehat{\Theta}_n(E)$ defined in (13) in case $\Sigma_{0,ii} = 1$ for all i, where $\widehat{S}_n$ is replaced with $\widehat{\Gamma}_n$, and its inverse $\widehat{\Sigma}_n$, and $\widehat{\Gamma}_n$: for E as defined in (23),

$$\begin{aligned} \widehat{\Theta}_{n,ij} &= 0,\ \forall (i,j) \in \mathcal{D},\ \text{and} \\ \widehat{\Sigma}_{n,ij} &= \widehat{\Gamma}_{n,ij} = \widehat{S}_{n,ij}/\widehat{\sigma}_i\widehat{\sigma}_j,\ \forall (i,j) \in E,\ \text{and} \\ \widehat{\Sigma}_{n,ii} &= 1,\ \forall i = 1,\ldots,p. \end{aligned}$$

Finally, we state Theorem 6, which yields a general bound on estimating the inverse of the correlation matrix, when $\Sigma_{0,11},\ldots,\Sigma_{0,pp}$ take arbitrary unknown values in $\mathbb{R}^+ = (0,\infty)$. The corresponding

estimator is based on estimating the inverse of the correlation matrix, which we denote by Ω_0. We use the following notations. Let $\Psi_0 = (\rho_{0,ij})$ be the true correlation matrix and let $\Omega_0 = \Psi_0^{-1}$. Let $W = \mathrm{diag}(\Sigma_0)^{1/2}$. Let us denote the diagonal entries of W with $\sigma_1, \ldots, \sigma_p$ where $\sigma_i := \Sigma_{0,ii}^{1/2}$ for all i. Then the following holds:

$$\Sigma_0 \;=\; W\Psi_0 W \;\text{ and } \Theta_0 \;=\; W^{-1}\Omega_0 W^{-1}.$$

Given sample covariance matrix $\widehat{S}_n$, we construct sample correlation matrix $\widehat{\Gamma}_n$ as follows. Let $\widehat{W} = \mathrm{diag}(\widehat{S}_n)^{1/2}$ and

$$\widehat{\Gamma}_n = \widehat{W}^{-1}(\widehat{S}_n)\widehat{W}^{-1}, \;\text{ where }\; \widehat{\Gamma}_{n,ij} = \frac{\widehat{S}_{n,ij}}{\widehat{\sigma}_i\widehat{\sigma}_j} = \frac{\langle X_i, X_j\rangle}{\|X_i\|_2 \left\|X_j\right\|_2}$$

where $\widehat{\sigma}_i^2 := \widehat{S}_{n,ii}$. Thus $\widehat{\Gamma}_n$ is a matrix with diagonal entries being all 1s and non-diagonal entries being the sample correlation coefficients, which we denote by $\widehat{\rho}_{ij}$.

The maximum likelihood estimate for $\Omega_0 = \Psi_0^{-1}$ minimizes over all $\Omega \succ 0$,

$$\widehat{R}_n(\Omega) = \mathrm{tr}(\Omega\widehat{\Gamma}_n) - \log|\Omega|. \tag{27}$$

To facilitate technical discussions, we need to introduce some more notation. Let $\mathcal{S}_{++}^p$ denote the set of $p \times p$ symmetric positive definite matrices:

$$\mathcal{S}_{++}^p = \{\Theta \in \mathbb{R}^{p\times p} | \Theta \succ 0\}.$$

Let us define a subspace $\mathcal{S}_E^p$ corresponding to an edge set $E \subset \{(i,j) : i,j = 1,\ldots,p, i \neq j\}$:

$$\mathcal{S}_E^p := \{\Theta \in \mathbb{R}^{p\times p} \mid \theta_{ij} = 0 \;\forall\, i \neq j \;\text{ s.t. } (i,j) \notin E\} \;\text{ and denote }\; \mathcal{S}_n \;=\; \mathcal{S}_{++}^p \cap \mathcal{S}_E^p. \tag{28}$$

Minimizing $\widehat{R}_n(\Theta)$ without constraints gives $\widehat{\Psi}_n = \widehat{\Gamma}_n$. Subject to the constraints that $\Omega \in \mathcal{S}_n$ as defined in (28), we write the maximum likelihood estimate for Ω_0:

$$\widehat{\Omega}_n(E) := \arg\min_{\Omega\in\mathcal{S}_n} \widehat{R}_n(\Omega) = \arg\min_{\Omega\in\mathcal{S}_{++}^p\cap\mathcal{S}_E^p} \left\{\mathrm{tr}(\Omega\widehat{\Gamma}_n) - \log|\Omega|\right\}, \tag{29}$$

which yields the following relationships regarding $\widehat{\Omega}_n(E)$, its inverse $\widehat{\Psi}_n = (\widehat{\Omega}_n(E))^{-1}$, and $\widehat{\Gamma}_n$. For E as defined in (23),

$$\begin{aligned} \widehat{\Omega}_{n,ij} &= 0,\ \forall (i,j) \in \mathcal{D}, \\ \widehat{\Psi}_{n,ij} &= \widehat{\Gamma}_{n,ij} := \widehat{\rho}_{ij} \quad \forall (i,j) \in E, \\ \text{and}\quad \widehat{\Psi}_{n,ii} &= 1 \quad \forall i = 1,\ldots,p. \end{aligned}$$

Given $\widehat{\Omega}_n(E)$ and its inverse $\widehat{\Psi}_n = (\widehat{\Omega}_n(E))^{-1}$, we obtain

$$\widehat{\Sigma}_n = \widehat{W}\widehat{\Psi}_n\widehat{W} \;\text{ and }\; \widehat{\Theta}_n = \widehat{W}^{-1}\widehat{\Omega}_n\widehat{W}^{-1}$$

and therefore the following holds: for E as defined in (23),

$$\begin{aligned} \widehat{\Theta}_{n,ij} &= 0, \ \forall (i,j) \in \mathcal{D}, \\ \widehat{\Sigma}_{n,ij} &= \widehat{\sigma}_i \widehat{\sigma}_j \widehat{\Psi}_{n,ij} = \widehat{\sigma}_i \widehat{\sigma}_j \widehat{\Gamma}_{n,ij} = \widehat{S}_{n,ij} \quad \forall (i,j) \in E, \\ \text{and} \quad \widehat{\Psi}_{n,ii} &= \widehat{\sigma}_i^2 = \widehat{S}_{n,ii} \quad \forall i = 1, \ldots, p. \end{aligned}$$

The proof of Theorem 6 appears in Section E.

Theorem 6 *Consider data generating random variables as in expression (15) and assume that* $(A1)$ *and* $(A2)$ *hold. Let* $\sigma^2_{\max} := \max_i \Sigma_{0,ii} < \infty$ *and* $\sigma^2_{\min} := \min_i \Sigma_{0,ii} > 0$. *Let* $\mathcal{E}$ *be some event such that* $\mathbb{P}(\mathcal{E}) \geq 1 - d/p^2$ *for a small constant* d. *Let* $S_{0,n}$ *be as defined in* (8). *Suppose on event* $\mathcal{E}$*:*

1. *We obtain an edge set* E *such that its size* $|E| = lin(S_{0,n})$ *is a linear function in* $S_{0,n}$.
2. *And for* $\widetilde{\Theta}_0$ *as in* (24) *and for some constant* C_{bias} *to be specified in* (59)*, we have*

$$\left\|\Theta_{0,\mathcal{D}}\right\|_F := \left\|\widetilde{\Theta}_0 - \Theta_0\right\|_F \leq C_{\text{bias}} \sqrt{2 S_{0,n} \log(p)/n}. \tag{30}$$

Let $\widehat{\Omega}_n(E)$ *be as defined in* (29) *Suppose the sample size satisfies for* $C_3 \geq 4\sqrt{5/3}$,

$$n > \frac{144 \sigma^4_{\max}}{M^2_{\text{low}}} \left(4C_3 + \frac{13 M_{\text{upp}}}{12 \sigma^2_{\min}} \right)^2 \max \left\{ 2|E| \log \max(n,p),\ C^2_{\text{bias}} 2 S_{0,n} \log p \right\}. \tag{31}$$

Then with probability $\geq 1 - (d+1)/p^2$, *we have for* $M = (9\sigma^4_{\max}/(2\underline{k}^2)) \cdot \left(4C_3 + 13 M_{\text{upp}}/(12\sigma^2_{\min})\right)$

$$\left\|\widehat{\Omega}_n(E) - \Omega_0\right\|_F \leq (M+1) \max \left\{ \sqrt{2|E| \log \max(n,p)/n},\ C_{\text{bias}} \sqrt{2 S_{0,n} \log(p)/n} \right\}. \tag{32}$$

Remark 7 *We note that the constants in Theorem 6 are by no means the best possible. From* (32)*, we can derive bounds on* $\|\widehat{\Theta}_n(E) - \Theta_0\|_2$ *and* $\|\widehat{\Sigma}_n(E) - \Sigma_0\|_2$ *to be in the same order as in* (32) *following the analysis in Rothman et al. [2008, Theorem 2]. The corresponding bounds on the Frobenius norms on covariance estimation would be in the order of* $O_P\left(\sqrt{\frac{p+S_0}{n}}\right)$ *as stated in Remark 3.*

4. Numerical Results

We consider the empirical performance for simulated and real data. We compare our estimation method with the GLasso, the Space method and a simplified Gelato estimator without thresholding for inferring the conditional independence graph. The comparison with the latter should yield some evidence about the role of thresholding in Gelato. The GLasso is defined as:

$$\widehat{\Theta}_{\text{GLasso}} = \underset{\Theta \succ 0}{\text{argmin}} (\text{tr}(\widehat{\Gamma}_n \Theta) - \log|\Theta| + \rho \sum_{i<j} |\theta_{ij}|),$$

where $\widehat{\Gamma}_n$ is the empirical correlation matrix and the minimization is over positive definite matrices. Sparse partial correlation estimation (Space) is an approach for selecting non-zero partial correlations in the high-dimensional framework. The method assumes an overall sparsity of the partial correlation matrix and employs sparse regression techniques for model fitting. For details see Peng et al. [2009]. We use Space with weights all equal to one, which refers to the method type `space` in Peng et al. [2009]. For the Space method, estimation of Θ_0 is done via maximum likelihood as in (13) based on the edge set $\widehat{E}_n^{(Space)}$ from the estimated sparse partial correlation matrix. For computation of the three different methods, we used the R-packages `glmnet` [Friedman et al., 2010], `glasso` [Friedman et al., 2007] and `space` [Peng et al., 2009].

4.1 Simulation Study

In our simulation study, we look at three different models.

- An AR(1)-Block model. In this model the covariance matrix is block-diagonal with equal-sized AR(1)-blocks of the form $\Sigma_{Block} = \{0.9^{|i-j|}\}_{i,j}$.
- The random concentration matrix model considered in Rothman et al. [2008]. In this model, the concentration matrix is $\Theta = B + \delta I$ where each off-diagonal entry in B is generated independently and equal to 0 or 0.5 with probability $1-\pi$ or π, respectively. All diagonal entries of B are zero, and δ is chosen such that the condition number of Θ is p.
- The exponential decay model considered in Fan et al. [2009]. In this model we consider a case where no element of the concentration matrix is exactly zero. The elements of Θ_0 are given by $\theta_{0,ij} = \exp(-2|i-j|)$ equals essentially zero when the difference $|i-j|$ is large.

We compare the three estimators for each model with $p = 300$ and $n = 40, 80, 320$. For each model we sample data $X^{(1)}, \ldots, X^{(n)}$ i.i.d. $\sim \mathcal{N}(0, \Sigma_0)$. We use two different performance measures. The Frobenius norm of the estimation error $\|\widehat{\Sigma}_n - \Sigma_0\|_F$ and $\|\widehat{\Theta}_n - \Theta_0\|_F$, and the Kullback Leibler divergence between $\mathcal{N}(0, \Sigma_0)$ and $\mathcal{N}(0, \widehat{\Sigma}_n)$ as defined in (19).

For the three estimation methods we have various tuning parameters, namely λ, τ (for Gelato), ρ (for GLasso) and η (for Space). We denote the regularization parameter of the Space technique by η in contrary to Peng et al. [2009], in order to distinguish it from the other parameters. Due to the computational complexity we specify the two parameters of our Gelato method sequentially. That is, we derive the optimal value of the penalty parameter λ by 10-fold cross-validation with respect to the test set squared error for all the nodewise regressions. After fixing $\lambda = \lambda_{CV}$ we obtain the optimal threshold τ again by 10-fold cross-validation but with respect to the negative Gaussian log-likelihood ($\mathrm{tr}(\widehat{\Theta}\widehat{S}^{out}) - \log|\widehat{\Theta}|$, where $\widehat{S}^{out}$ is the empirical covariance of the hold-out data). We could use individual tuning parameters for each of the regressions. However, this turned out to be sub-optimal in some simulation scenarios (and never really better than using a single tuning parameter λ, at least in the scenarios we considered). For the penalty parameter ρ of the GLasso estimator and the parameter η of the Space method we also use a 10-fold cross-validation with

respect to the negative Gaussian log-likelihood. The grids of candidate values are given as follows:

$$\lambda_k = A_k\sqrt{\frac{\log p}{n}} \quad k = 1,\ldots,10 \quad \text{with} \quad \tau_k = 0.75\cdot B_k\sqrt{\frac{\log p}{n}},$$
$$\rho_k = C_k\sqrt{\frac{\log p}{n}} \quad k = 1,\ldots,10,$$
$$\eta_r = 1.56\sqrt{n}\Phi^{-1}\left(1-\frac{D_r}{2p^2}\right) \quad r = 1,\ldots,7,$$

where $A_k, B_k, C_k \in \{0.01, 0.05, 0.1, 0.3, 0.5, 1, 2, 4, 8, 16\}$ and $D_r \in \{0.01, 0.05, 0.075, 0.1, 0.2, 0.5, 1\}$. The two different performance measures are evaluated for the estimators based on the sample $X^{(1)},\ldots,X^{(n)}$ with optimal CV-estimated tuning parameters λ, τ, ρ and η for each model from above. All results are based on 50 independent simulation runs.

4.1.1 THE AR(1)-BLOCK MODEL

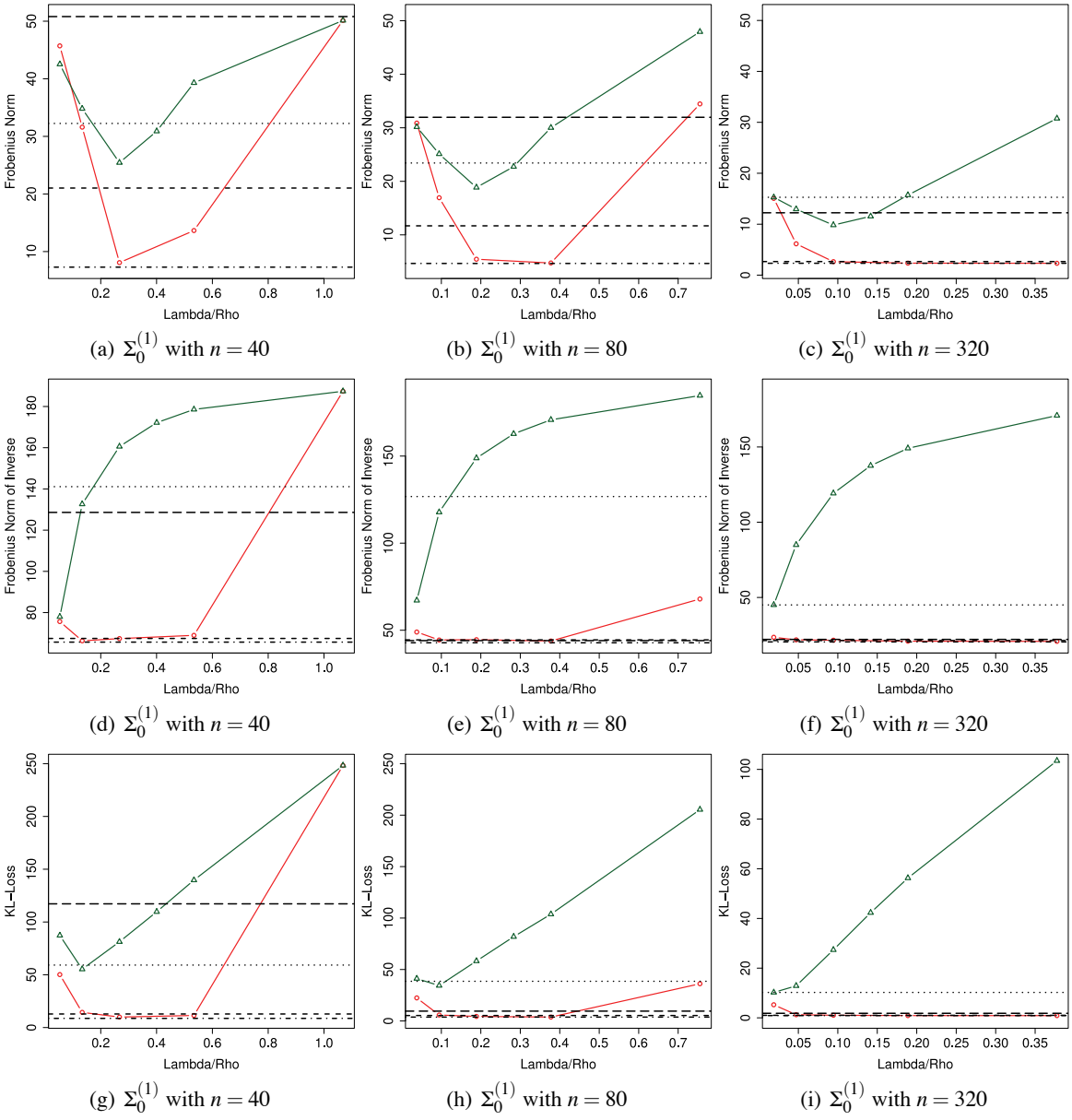
We consider two different covariance matrices. The first one is a simple auto-regressive process of order one with trivial block size equal to $p = 300$, denoted by $\Sigma_0^{(1)}$. This is also known as a Toeplitz matrix. That is, we have $\Sigma^{(1)}_{0;i,j} = 0.9^{|i-j|}\ \forall\ i,j \in \{1,...,p\}$. The second matrix $\Sigma_0^{(2)}$ is a block-diagonal matrix with AR(1) blocks of equal block size 30×30, and hence the block-diagonal of $\Sigma_0^{(2)}$ equals $\Sigma_{Block;i,j} = 0.9^{|i-j|}$, $i,j \in \{1,\ldots,30\}$. The simulation results for the AR(1)-block models are shown in Figure 1 and 2.

The figures show a substantial performance gain of our method compared to the GLasso in both considered covariance models. This result speaks for our method, especially because AR(1)-block models are very simple. The Space method performs about as well as Gelato, except for the Frobenius norm of $\widehat{\Sigma}_n - \Sigma_0$. There we see an performance advantage of the Space method compared to Gelato. We also exploit the clear advantage of thresholding in Gelato for a small sample size.

4.1.2 THE RANDOM PRECISION MATRIX MODEL

For this model we also consider two different matrices, which differ in sparsity. For the sparser matrix $\Theta_0^{(3)}$ we set the probability π to 0.1. That is, we have an off diagonal entry in $\Theta^{(3)}$ of 0.5 with probability $\pi = 0.1$ and an entry of 0 with probability 0.9. In the case of the second matrix $\Theta_0^{(4)}$ we set π to 0.5 which provides us with a denser concentration matrix. The simulation results for the two performance measures are given in Figure 3 and 4.

From Figures 3 and 4 we see that GLasso performs better than Gelato with respect to $\|\widehat{\Theta}_n - \Theta_0\|_F$ and the Kullback Leibler divergence in both the sparse and the dense simulation setting. If we consider $\|\widehat{\Sigma}_n - \Sigma_0\|_F$, Gelato seems to keep up with GLasso to some degree. For the Space method we have a similar situation to the one with GLasso. The Space method outperforms Gelato for $\|\widehat{\Theta}_n - \Theta_0\|_F$ and $D_{\mathrm{KL}}(\Sigma_0\|\widehat{\Sigma}_n)$ but for $\|\widehat{\Sigma}_n - \Sigma_0\|_F$, Gelato somewhat keeps up with Space.

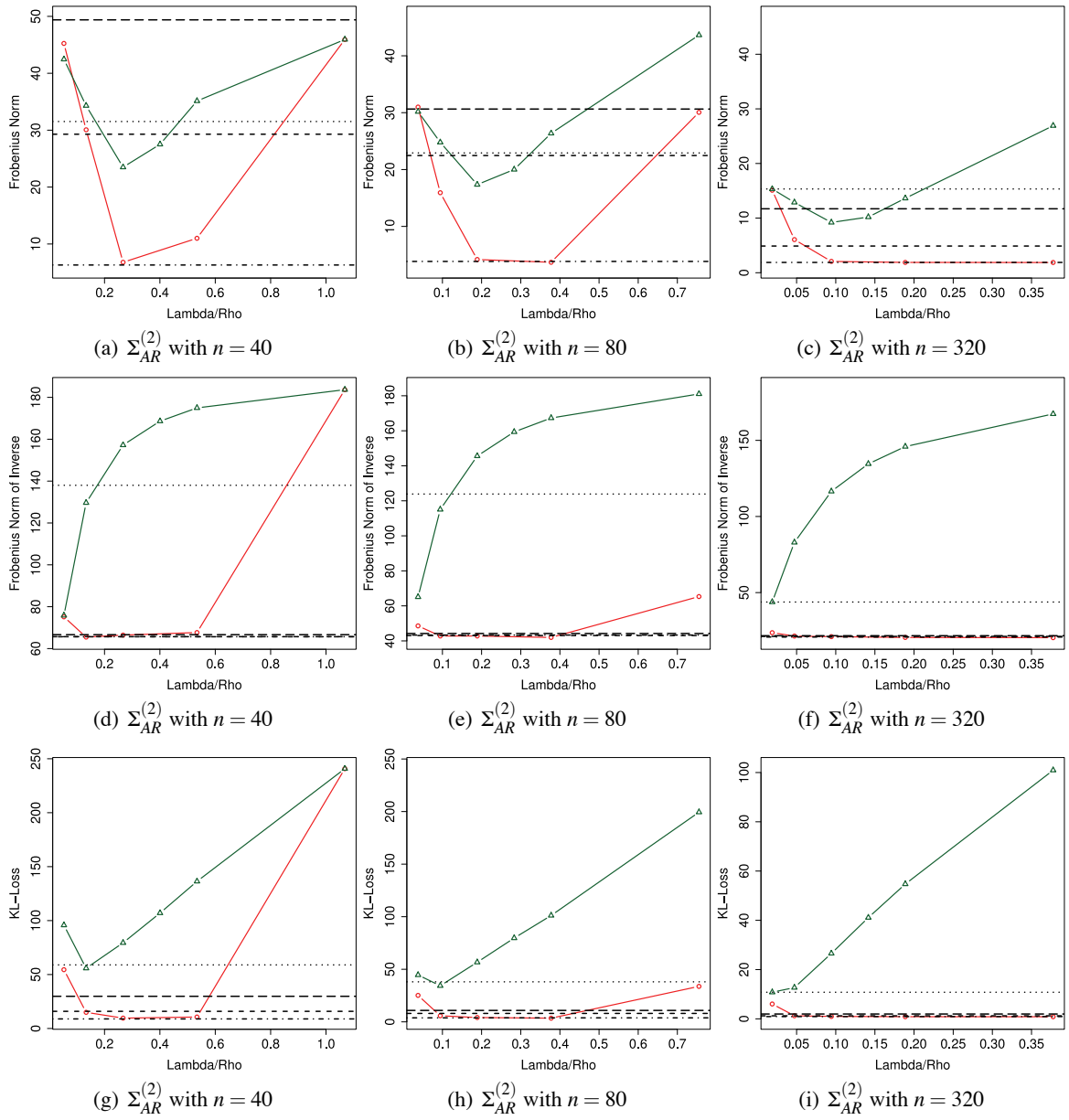


(a) $\Sigma_0^{(1)}$ with $n = 40$ (b) $\Sigma_0^{(1)}$ with $n = 80$ (c) $\Sigma_0^{(1)}$ with $n = 320$

(d) $\Sigma_0^{(1)}$ with $n = 40$ (e) $\Sigma_0^{(1)}$ with $n = 80$ (f) $\Sigma_0^{(1)}$ with $n = 320$

(g) $\Sigma_0^{(1)}$ with $n = 40$ (h) $\Sigma_0^{(1)}$ with $n = 80$ (i) $\Sigma_0^{(1)}$ with $n = 320$

Figure 1: Plots for model $\Sigma_0^{(1)}$. The triangles (green) stand for the GLasso and the circles (red) for our Gelato method with a reasonable value of τ. The horizontal lines show the performances of the three techniques for cross-validated tuning parameters λ, τ, ρ and η. The dashed line stands for our Gelato method, the dotted one for the GLasso and the dash-dotted line for the Space technique. The additional dashed line with the longer dashes stands for the Gelato without thresholding. Lambda/Rho stands for λ or ρ, respectively.

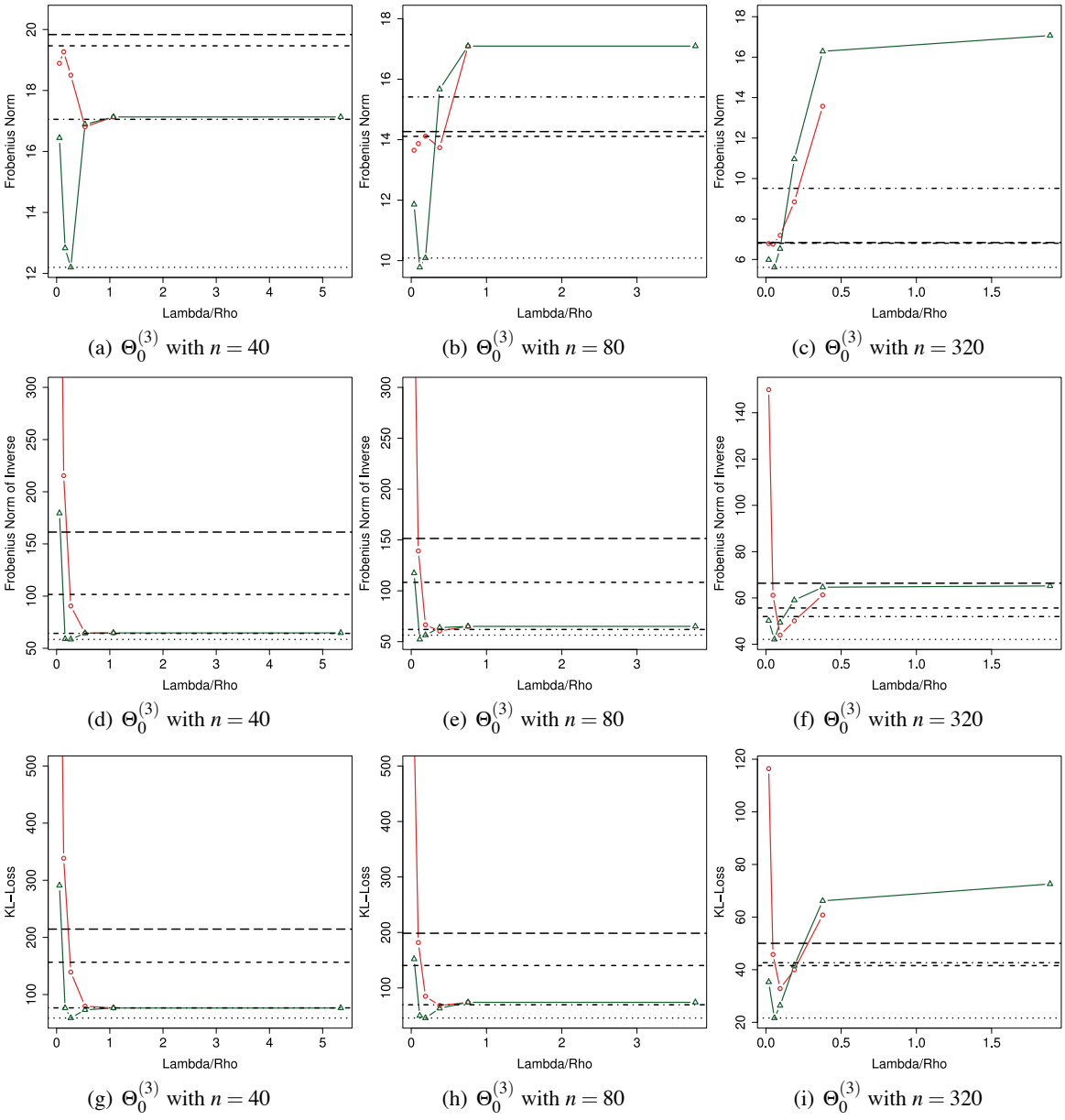


(a) $\Sigma_{AR}^{(2)}$ with $n = 40$ (b) $\Sigma_{AR}^{(2)}$ with $n = 80$ (c) $\Sigma_{AR}^{(2)}$ with $n = 320$

(d) $\Sigma_{AR}^{(2)}$ with $n = 40$ (e) $\Sigma_{AR}^{(2)}$ with $n = 80$ (f) $\Sigma_{AR}^{(2)}$ with $n = 320$

(g) $\Sigma_{AR}^{(2)}$ with $n = 40$ (h) $\Sigma_{AR}^{(2)}$ with $n = 80$ (i) $\Sigma_{AR}^{(2)}$ with $n = 320$

Figure 2: Plots for model $\Sigma_0^{(2)}$. The triangles (green) stand for the GLasso and the circles (red) for our Gelato method with a reasonable value of τ. The horizontal lines show the performances of the three techniques for cross-validated tuning parameters λ, τ, ρ and η. The dashed line stands for our Gelato method, the dotted one for the GLasso and the dash-dotted line for the Space technique. The additional dashed line with the longer dashes stands for the Gelato without thresholding. Lambda/Rho stands for λ or ρ, respectively.

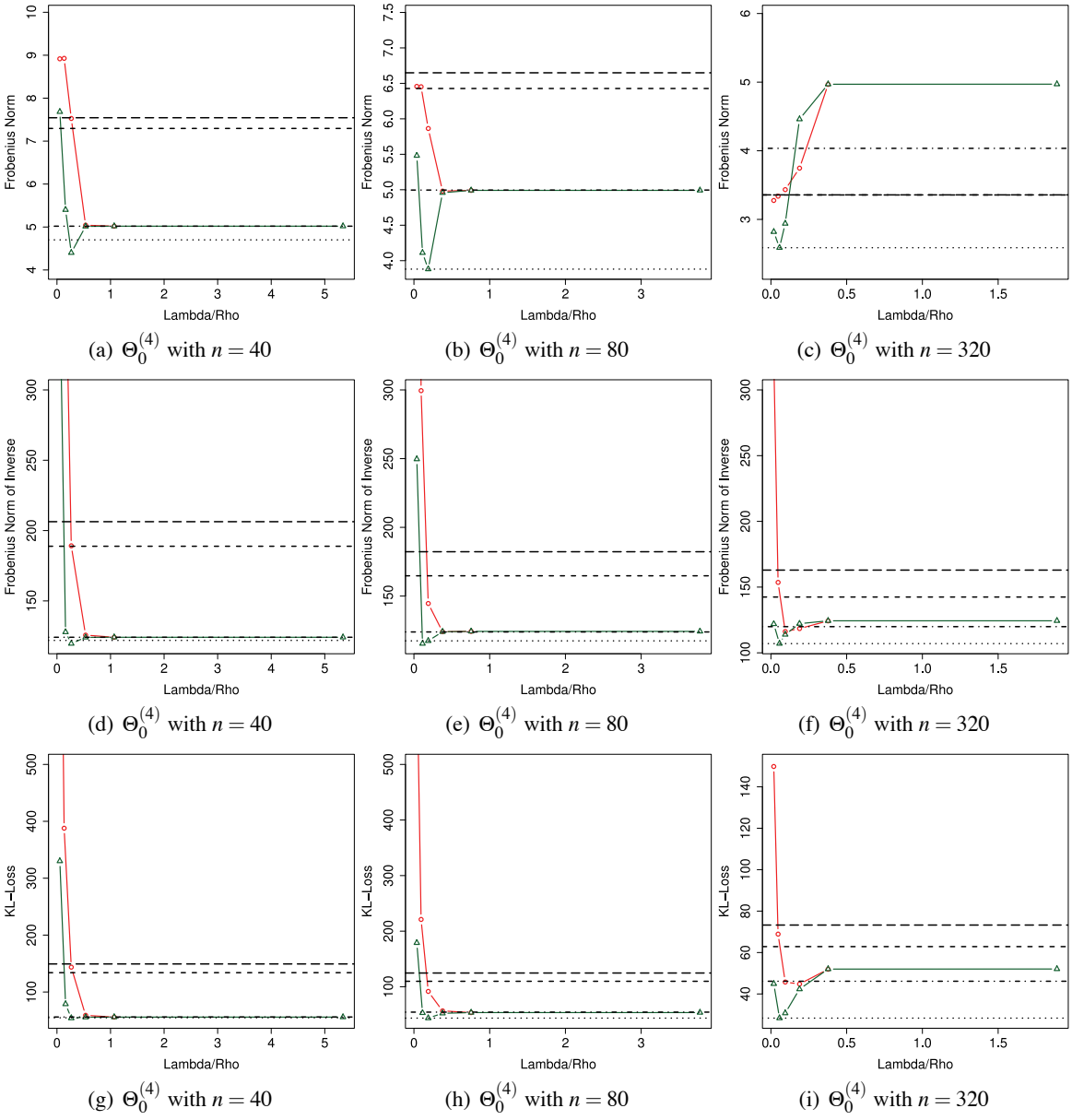


(a) $\Theta_0^{(3)}$ with $n = 40$ (b) $\Theta_0^{(3)}$ with $n = 80$ (c) $\Theta_0^{(3)}$ with $n = 320$

(d) $\Theta_0^{(3)}$ with $n = 40$ (e) $\Theta_0^{(3)}$ with $n = 80$ (f) $\Theta_0^{(3)}$ with $n = 320$

(g) $\Theta_0^{(3)}$ with $n = 40$ (h) $\Theta_0^{(3)}$ with $n = 80$ (i) $\Theta_0^{(3)}$ with $n = 320$

Figure 3: Plots for model $\Theta_0^{(3)}$. The triangles (green) stand for the GLasso and the circles (red) for our Gelato method with a reasonable value of τ. The horizontal lines show the performances of the three techniques for cross-validated tuning parameters λ, τ, ρ and η. The dashed line stands for our Gelato method, the dotted one for the GLasso and the dash-dotted line for the Space technique. The additional dashed line with the longer dashes stands for the Gelato without thresholding. Lambda/Rho stands for λ or ρ, respectively.



(a) $\Theta_0^{(4)}$ with $n = 40$ (b) $\Theta_0^{(4)}$ with $n = 80$ (c) $\Theta_0^{(4)}$ with $n = 320$

(d) $\Theta_0^{(4)}$ with $n = 40$ (e) $\Theta_0^{(4)}$ with $n = 80$ (f) $\Theta_0^{(4)}$ with $n = 320$

(g) $\Theta_0^{(4)}$ with $n = 40$ (h) $\Theta_0^{(4)}$ with $n = 80$ (i) $\Theta_0^{(4)}$ with $n = 320$

Figure 4: Plots for model $\Theta_0^{(4)}$. The triangles (green) stand for the GLasso and the circles (red) for our Gelato method with a reasonable value of τ. The horizontal lines show the performances of the three techniques for cross-validated tuning parameters λ, τ, ρ and η. The dashed line stands for our Gelato method, the dotted one for the GLasso and the dash-dotted line for the Space technique. The additional dashed line with the longer dashes stands for the Gelato without thresholding. Lambda/Rho stands for λ or ρ, respectively.

4.1.3 THE EXPONENTIAL DECAY MODEL

In this simulation setting we only have one version of the concentration matrix $\Theta_0^{(5)}$. The entries of $\Theta_0^{(5)}$ are generated by $\theta_{0,ij}^{(5)} = \exp(-2|i-j|)$. Thus, Σ_0 is a banded and sparse matrix.

Figure 5 shows the results of the simulation. We find that all three methods show equal performances in both the Frobenius norm and the Kullback Leibler divergence. This is interesting because even with a sparse approximation of Θ_0 (with GLasso or Gelato), we obtain competitive performance for (inverse) covariance estimation.

4.1.4 SUMMARY

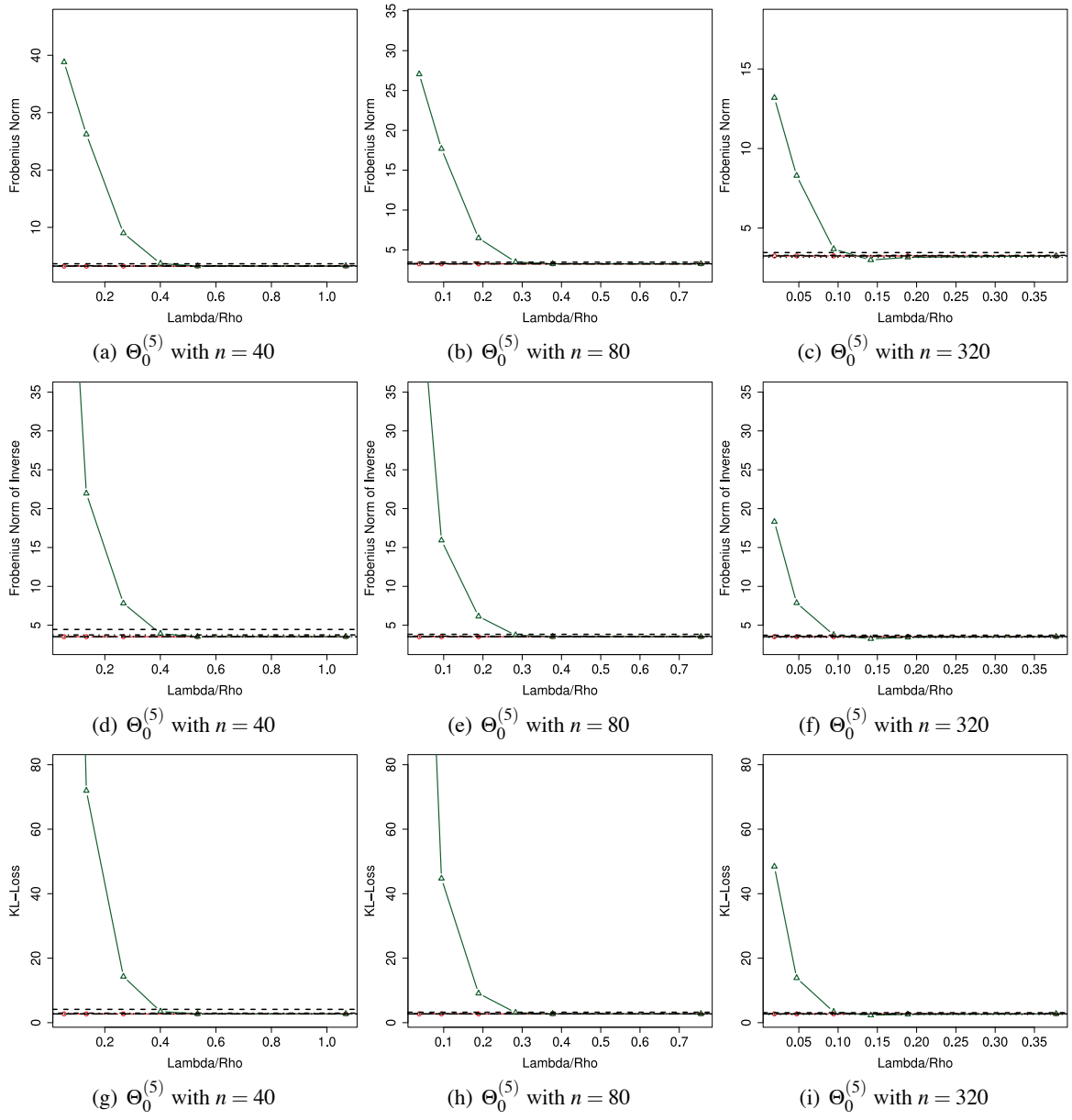
Overall we can say that the performance of the methods depend on the model. For the models $\Sigma_0^{(1)}$ and $\Sigma_0^{(2)}$ the Gelato method performs best. In case of the models $\Theta_0^{(3)}$ and $\Theta_0^{(4)}$, Gelato gets outperformed by GLasso and the Space method and for the model $\Theta_0^{(5)}$ none of the three methods has a clear advantage. In Figures 1 to 4, we see the advantage of Gelato with thresholding over the one without thresholding, in particular, for the simulation settings $\Sigma_0^{(1)}$, $\Sigma_0^{(2)}$ and $\Theta_0^{(3)}$. Thus thresholding is a useful feature of Gelato.

4.2 Application to Real Data

We show two examples in this subsection.

4.2.1 ISOPRENOID GENE PATHWAY IN ARABIDOBSIS THALIANA

In this example we compare the two estimators on the isoprenoid biosynthesis pathway data given in Wille et al. [2004]. Isoprenoids play various roles in plant and animal physiological processes and as intermediates in the biological synthesis of other important molecules. In plants they serve numerous biochemical functions in processes such as photosynthesis, regulation of growth and development. The data set consists of $p = 39$ isoprenoid genes for which we have $n = 118$ gene expression patterns under various experimental conditions. In order to compare the two techniques we compute the negative log-likelihood via 10-fold cross-validation for different values of λ, τ and ρ. In Figure 6 we plot the cross-validated negative log-likelihood against the logarithm of the average number of non-zero entries (logarithm of the ℓ_0-norm) of the estimated concentration matrix $\widehat{\Theta}_n$. The logarithm of the ℓ_0-norm reflects the sparsity of the matrix $\widehat{\Theta}_n$ and therefore the figures show the performance of the estimators for different levels of sparsity. The plots do not allow for a clear conclusion. The GLasso performs slightly better when allowing for a rather dense fit. On the other hand, when requiring a sparse fit, the Gelato performs better.

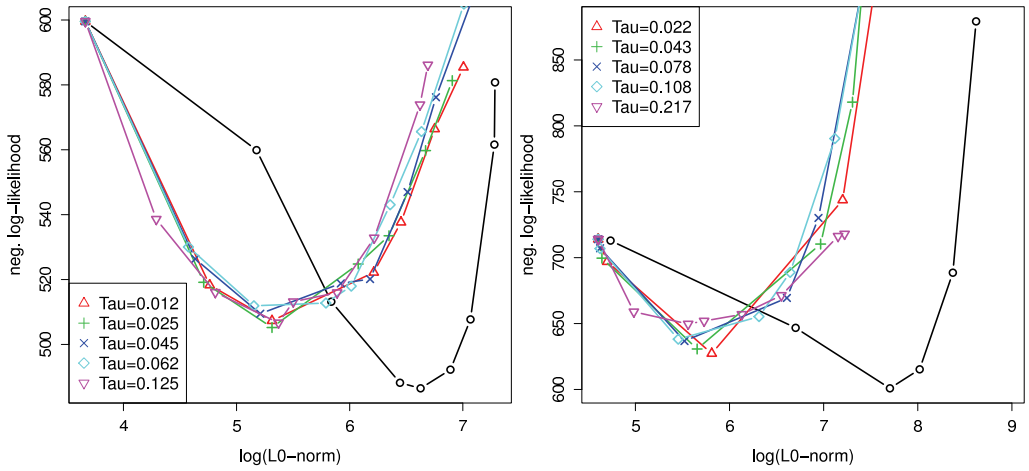


(a) $\Theta_0^{(5)}$ with $n = 40$ (b) $\Theta_0^{(5)}$ with $n = 80$ (c) $\Theta_0^{(5)}$ with $n = 320$

(d) $\Theta_0^{(5)}$ with $n = 40$ (e) $\Theta_0^{(5)}$ with $n = 80$ (f) $\Theta_0^{(5)}$ with $n = 320$

(g) $\Theta_0^{(5)}$ with $n = 40$ (h) $\Theta_0^{(5)}$ with $n = 80$ (i) $\Theta_0^{(5)}$ with $n = 320$

Figure 5: Plots for model $\Theta_0^{(5)}$. The triangles (green) stand for the GLasso and the circles (red) for our Gelato method with a reasonable value of τ. The horizontal lines show the performances of the three techniques for cross-validated tuning parameters λ, τ, ρ and η. The dashed line stands for our Gelato method, the dotted one for the GLasso and the dash-dotted line for the Space technique. The additional dashed line with the longer dashes stands for the Gelato without thresholding. Lambda/Rho stands for λ or ρ, respectively.



(a) isoprenoid data (b) breast cancer data

Figure 6: Plots for the isoprenoid data from arabidopsis thaliana (a) and the human breast cancer data (b). 10-fold cross-validation of negative log-likelihood against the logarithm of the average number of non-zero entries of the estimated concentration matrix $\widehat{\Theta}_n$. The circles stand for the GLasso and the Gelato is displayed for various values of τ.

4.2.2 CLINICAL STATUS OF HUMAN BREAST CANCER

As a second example, we compare the two methods on the breast cancer data set from West et al. [2001]. The tumor samples were selected from the Duke Breast Cancer SPORE tissue bank. The data consists of $p = 7129$ genes with $n = 49$ breast tumor samples. For the analysis we use the 100 variables with the largest sample variance. As before, we compute the negative log-likelihood via 10-fold cross-validation. Figure 6 shows the results. In this real data example the interpretation of the plots is similar as for the arabidopsis data set. For dense fits, GLasso is better while Gelato has an advantage when requiring a sparse fit.

5. Conclusions

We propose and analyze the Gelato estimator. Its advantage is that it automatically yields a positive definite covariance matrix $\widehat{\Sigma}_n$, it enjoys fast convergence rate with respect to the operator and Frobenius norm of $\widehat{\Sigma}_n - \Sigma_0$ and $\widehat{\Theta}_n - \Theta_0$. For estimation of Θ_0, Gelato has in some settings a better rate of convergence than the GLasso or SCAD type estimators. From a theoretical point of view, our method is clearly aimed for bounding the operator and Frobenius norm of the inverse covariance matrix. We also derive bounds on the convergence rate for the estimated covariance matrix and on the Kullback Leibler divergence. From a non-asymptotic point of view, our method has a clear advantage when the sample size is small relative to the sparsity $S = |E_0|$: for a given sample size n, we bound the variance in our re-estimation stage by excluding edges of E_0 with small weights from

the selected edge set $\widehat{E}_n$ while ensuring that we do not introduce too much bias. Our Gelato method also addresses the bias problem inherent in the GLasso estimator since we no longer shrink the entries in the covariance matrix corresponding to the selected edge set $\widehat{E}_n$ in the maximum likelihood estimate, as shown in Section 3.3.

Our experimental results show that Gelato performs better than GLasso or the Space method for AR-models while the situation is reversed for some random precision matrix models; in case of an exponential decay model for the precision matrix, all methods exhibit the same performance. For Gelato, we demonstrate that thresholding is a valuable feature. We also show experimentally how one can use cross-validation for choosing the tuning parameters in regression and thresholding. Deriving theoretical results on cross-validation is not within the scope of this paper.

Acknowledgments

We thank Larry Wasserman, Liza Levina, the anonymous reviewers and the editor for helpful comments on this work. Shuheng Zhou thanks Bin Yu warmly for hosting her visit at UC Berkeley while she was conducting this research in Spring 2010. SZ's research was supported in part by the Swiss National Science Foundation (SNF) Grant 20PA21-120050/1. Min Xu's research was supported by NSF grant CCF-0625879 and AFOSR contract FA9550-09-1-0373.

Appendix A. Theoretical Analysis and Proofs

In this section, we specify some preliminary definitions. First, note that when we discuss estimating the parameters Σ_0 and $\Theta_0 = \Sigma_0^{-1}$, we always assume that

$$\varphi_{\max}(\Sigma_0) := 1/\varphi_{\min}(\Theta_0) \leq 1/\underline{c} < \infty \text{ and } 1/\varphi_{\max}(\Theta_0) = \varphi_{\min}(\Sigma_0) \geq \underline{k} > 0, \tag{33}$$

$$\text{where we assume } \underline{k}, \underline{c} \leq 1 \quad \text{so that } \underline{c} \leq 1 \leq 1/\underline{k}. \tag{34}$$

It is clear that these conditions are exactly that of (A2) in Section 3 with

$$M_{\text{upp}} := 1/\underline{c} \quad \text{and} \quad M_{\text{low}} := \underline{k},$$

where it is clear that for $\Sigma_{0,ii} = 1, i = 1, \ldots, p$, we have the sum of p eigenvalues of Σ_0, $\sum_{i=1}^{p} \varphi_i(\Sigma_0) = \text{tr}(\Sigma_0) = p$. Hence it will make sense to assume that (34) holds, since otherwise, (33) implies that $\varphi_{\min}(\Sigma_0) = \varphi_{\max}(\Sigma_0) = 1$ which is unnecessarily restrictive.

We now define parameters relating to the key notion of *essential sparsity* s_0 as explored in Candès and Tao [2007] and Zhou [2009, 2010a] for regression. Denote the number of non-zero non-diagonal entries in each row of Θ_0 by s^i. Let $s = \max_{i=1,\ldots,p} s^i$ denote the highest node degree in $G = (V, E_0)$. Consider nodewise regressions as in (2), where we are given vectors of parameters $\{\beta^i_j, j = 1, \ldots, p, j \neq i\}$ for $i = 1, \ldots, p$. With respect to the degree of node i for each i, we define

$s_0^i \le s^i \le s$ as the smallest integer such that

$$\sum_{j=1,j\neq i}^{p} \min((\beta_j^i)^2, \lambda^2 \mathrm{Var}(V_i)) \le s_0^i \lambda^2 \mathrm{Var}(V_i), \text{ where } \lambda = \sqrt{2\log p/n}, \tag{35}$$

where s_0^i denotes $s_{0,n}^i$ as defined in (7).

Definition 8 (Bounded degree parameters.) *The size of the node degree s^i for each node i is upper bounded by an integer $s < p$. For s_0^i as in* (35)*, define*

$$s_0 \;:=\; \max_{i=1,\ldots,p} s_0^i \le s \text{ and } \; S_{0,n} \;:=\; \sum_{i=1,\ldots,p} s_0^i, \tag{36}$$

where $S_{0,n}$ is exactly the same as in (8)*, although we now drop subscript n from $s_{0,n}^i$ in order to simplify our notation.*

We now define the following parameters related to Σ_0. For an integer $m \le p$, we define the smallest and largest **m-sparse eigenvalues** of Σ_0 as follows:

$$\sqrt{\rho_{\min}(m)} \;:=\; \min_{t\neq 0; m-\text{sparse}} \frac{\left\|\Sigma_0^{1/2} t\right\|_2}{\|t\|_2}, \;\; \sqrt{\rho_{\max}(m)} := \max_{t\neq 0; m-\text{sparse}} \frac{\left\|\Sigma_0^{1/2} t\right\|_2}{\|t\|_2}.$$

Definition 9 (Restricted eigenvalue condition $RE(s_0,k_0,\Sigma_0)$). *For some integer $1 \le s_0 < p$ and a positive number k_0, the following condition holds for all $\upsilon \neq 0$,*

$$\frac{1}{K(s_0,k_0,\Sigma_0)} := \min_{\substack{J\subseteq\{1,\ldots,p\},\, \|\upsilon_{J^c}\|_1 \le k_0\|\upsilon_J\|_1 \\ |J|\le s_0}} \frac{\left\|\Sigma_0^{1/2}\upsilon\right\|_2}{\|\upsilon_J\|_2} > 0, \tag{37}$$

where υ_J represents the subvector of $\upsilon \in \mathbb{R}^p$ confined to a subset J of $\{1,\ldots,p\}$.

When s_0 and k_0 become smaller, this condition is easier to satisfy. When we only aim to estimate the graphical structure E_0 itself, the global conditions (33) need not hold in general. Hence up till Section D, we only need to assume that Σ_0 satisfies (37) for s_0 as in (35), and the sparse eigenvalue $\rho_{\min}(s) > 0$. In order of estimate the covariance matrix Σ_0, we do assume that (33) holds, which guarantees that the *RE* condition always holds on Σ_0, and $\rho_{\max}(m), \rho_{\min}(m)$ are upper and lower bounded by some constants for all $m \le p$. We continue to adopt parameters such as K, $\rho_{\max}(s)$, and $\rho_{\max}(3s_0)$ for the purpose of defining constants that are reasonable tight under condition (33). In general, one can think of

$$\rho_{\max}(\max(3s_0, s)) \ll 1/\underline{c} < \infty \text{ and } K^2(s_0,k_0,\Sigma_0) \ll 1/\underline{k} < \infty,$$

for $\underline{c}, \underline{k}$ as in (33) when s_0 is small.

Roughly speaking, for two variables X_i, X_j as in (1) such that their corresponding entry in $\Theta_0 = (\theta_{0,ij})$ satisfies: $\theta_{0,ij} < \lambda\sqrt{\theta_{0,ii}}$, where $\lambda = \sqrt{2\log(p)/n}$, we can not guarantee that $(i,j) \in \widehat{E}_n$ when

we aim to keep $\asymp s_0^i$ edges for node $i, i = 1, \ldots, p$. For a given Θ_0, as the sample size n increases, we are able to select edges with smaller coefficient $\theta_{0,ij}$. In fact it holds that

$$|\theta_{0,ij}| < \lambda\sqrt{\theta_{0,ii}} \text{ which is equivalent to } |\beta_j^i| < \lambda\sigma_{V_i}, \text{ for all } j \geq s_0^i + 1 + \mathbb{I}_{i \leq s_0^i + 1}, \tag{38}$$

where $\mathbb{I}_{\{\cdot\}}$ is the indicator function, if we order the regression coefficients as follows:

$$|\beta_1^i| \geq |\beta_2^i| \ldots \geq |\beta_{i-1}^i| \geq |\beta_{i+1}^i| \ldots \geq |\beta_p^i|,$$

in view of (2), which is the same as if we order for row i of Θ_0,

$$|\theta_{0,i1}| \geq |\theta_{0,i,2}| \ldots \geq |\theta_{0,i,i-1}| \geq |\theta_{0,i,i+1}| \ldots \geq |\theta_{0,i,p}|.$$

This has been shown by Candès and Tao [2007]; See also Zhou [2010a].

A.1 Concentration Bounds for the Random Design

For the random design X generated by (15), let $\Sigma_{0,ii} = 1$ for all i. In preparation for showing the oracle results of Lasso in Theorem 33, we first state some concentration bounds on X. Define for some $0 < \theta < 1$

$$\mathcal{F}(\theta) := \left\{X : \forall j = 1, \ldots, p,\ 1 - \theta \leq \left\|X_j\right\|_2 / \sqrt{n} \leq 1 + \theta\right\}, \tag{39}$$

where $X_1, \ldots, X_p$ are the column vectors of the $n \times p$ design matrix X. When all columns of X have an Euclidean norm close to $\sqrt{n}$ as in (39) , it makes sense to discuss the RE condition in the form of (40) as formulated by Bickel et al. [2009]. For the integer $1 \leq s_0 < p$ as defined in (35) and a positive number k_0, $RE(s_0, k_0, X)$ requires that the following holds for all $\upsilon \neq 0$,

$$\frac{1}{K(s_0, k_0, X)} \stackrel{\triangle}{=} \min_{\substack{J \subset \{1, \ldots, p\},\ \|\upsilon_{J^c}\|_1 \leq k_0 \|\upsilon_J\|_1 \\ |J| \leq s_0}} \frac{\|X\upsilon\|_2}{\sqrt{n}\,\|\upsilon_J\|_2} > 0. \tag{40}$$

The parameter $k_0 > 0$ is understood to be the same quantity throughout our discussion. The following event $\mathcal{R}$ provides an upper bound on $K(s_0, k_0, X)$ for a given $k_0 > 0$ when Σ_0 satisfies $RE(s_0, k_0, \Sigma_0)$ condition:

$$\mathcal{R}(\theta) := \left\{X : RE(s_0, k_0, X) \text{ holds with } 0 < K(s_0, k_0, X) \leq \frac{K(s_0, k_0, \Sigma_0)}{1 - \theta}\right\}.$$

For some integer $m \leq p$, we define the smallest and largest m-sparse eigenvalues of X to be

$$\begin{aligned} \Lambda_{\min}(m) &:= \min_{\upsilon \neq 0; m-\text{sparse}} \|X\upsilon\|_2^2 / (n\|\upsilon\|_2^2) \text{ and} \\ \Lambda_{\max}(m) &:= \max_{\upsilon \neq 0; m-\text{sparse}} \|X\upsilon\|_2^2 / (n\|\upsilon\|_2^2), \end{aligned}$$

upon which we define the following event:

$$\begin{aligned} \mathcal{M}(\theta) := \{X :\ &(41) \text{ holds } \forall m \leq \max(s, (k_0 + 1)s_0)\}, \text{ for which} \\ 0 < (1 - \theta)\sqrt{\rho_{\min}(m)} \leq \sqrt{\Lambda_{\min}(m)} \leq \sqrt{\Lambda_{\max}(m)} &\leq (1 + \theta)\sqrt{\rho_{\max}(m)}. \end{aligned} \tag{41}$$

Formally, we consider the set of random designs that satisfy all events as defined, for some $0 < \theta < 1$. Theorem 10 shows concentration results that we need for the present work, which follows from Theorem 1.6 in Zhou [2010b] and Theorem 3.2 in Rudelson and Zhou [2011].

Theorem 10 *Let $0 < \theta < 1$. Let $\rho_{\min}(s) > 0$, where $s < p$ is the maximum node-degree in G. Suppose $RE(s_0, 4, \Sigma_0)$ holds for s_0 as in* (36)*, where $\Sigma_{0,ii} = 1$ for $i = 1, \ldots, p$.*

Let $f(s_0) = \min(4s_0\rho_{\max}(s_0)\log(5ep/s_0), s_0 \log p)$. Let $c, \alpha, c' > 0$ be some absolute constants. Then, for a random design X as generated by (15), we have

$$\mathbb{P}(\mathcal{X}) := \mathbb{P}(\mathcal{R}(\theta) \cap \mathcal{F}(\theta) \cap \mathcal{M}(\theta)) \geq 1 - 3\exp(-c\theta^2 n/\alpha^4)$$

as long as the sample size satisfies

$$n > \max\left\{ \frac{9c'\alpha^4}{\theta^2} \max\left(36K^2(s_0, 4, \Sigma_0) f(s_0), \log p\right), \frac{80s\alpha^4}{\theta^2} \log\left(\frac{5ep}{s\theta}\right) \right\}. \tag{42}$$

Remark 11 *We note that the constraint $s < p/2$, which has appeared in Zhou [2010b, Theorem 1.6] is unnecessary. Under a stronger RE condition on Σ_0, a tighter bound on the sample size n, which is independent of $\rho_{\max}(s_0)$, is given in Rudelson and Zhou [2011] in order to guarantee $\mathcal{R}(\theta)$. We do not pursue this optimization here as we assume that $\rho_{\max}(s_0)$ is a bounded constant throughout this paper. We emphasize that we only need the first term in* (42) *in order to obtain $\mathcal{F}(\theta)$ and $\mathcal{R}(\theta)$; The second term is used to bound sparse eigenvalues of order s.*

A.2 Definitions Of Other Various Events

Under (A1) as in Section 3, excluding event $\mathcal{X}^c$ as bounded in Theorem 10 and events $\mathcal{C}_a, \mathcal{X}_0$ to be defined in this subsection, we can then proceed to treat $X \in \mathcal{X} \cap \mathcal{C}_a$ as a deterministic design in regression and thresholding, for which $\mathcal{R}(\theta) \cap \mathcal{M}(\theta) \cap \mathcal{F}(\theta)$ holds with $\mathcal{C}_a$, We then make use of event $\mathcal{X}_0$ in the MLE refitting stage for bounding the Frobenius norm. We now define two types of correlations events $\mathcal{C}_a$ and $\mathcal{X}_0$.

A.2.1 CORRELATION BOUNDS ON X_j AND V_i

In this section, we first bound the maximum correlation between pairs of random vectors (V_i, X_j), for all i, j where $i \neq j$, each of which corresponds to a pair of variables (V_i, X_j) as defined in (2) and (3). Here we use X_j and V_i, for all i, j, to denote both random vectors and their corresponding variables.

Let us define $\sigma_{V_j} := \sqrt{\mathrm{Var}(V_j)} \geq v > 0$ as a shorthand. Let $V_j' := V_j/\sigma_{V_j}$, $j = 1, \ldots, p$ be a standard normal random variable. Let us now define for all $j, k \neq j$,

$$Z_{jk} = \frac{1}{n}\langle V_j', X_k \rangle = \frac{1}{n}\sum_{i=1}^{n} v_{j,i}' x_{k,i},$$

where for all $i = 1,\ldots,n$ $v'_{j,i}, x_{k,i}, \forall j, k \neq j$ are independent standard normal random variables. For some $a \geq 6$, let event

$$\mathcal{C}_a := \left\{ \max_{j,k} |Z_{jk}| < \sqrt{1+a}\sqrt{(2\log p)/n} \text{ where } a \geq 6 \right\}.$$

A.2.2 BOUNDS ON PAIRWISE CORRELATIONS IN COLUMNS OF X

Let $\Sigma_0 := (\sigma_{0,ij})$, where we denote $\sigma_{0,ii} := \sigma_i^2$. Denote by $\Delta = X^T X/n - \Sigma_0$. Consider for some constant $C_3 > 4\sqrt{5/3}$,

$$\mathcal{X}_0 := \left\{ \max_{j,k} |\Delta_{jk}| < C_3 \sigma_i \sigma_j \sqrt{\log \max\{p,n\}/n} < 1/2 \right\}. \tag{43}$$

We first state Lemma 12, which is used for bounding a type of correlation events across all regressions; see proof of Theorem 15. It is also clear that event $\mathcal{C}_a$ is equivalent to the event to be defined in (44). Lemma 12 also justifies the choice of λ_n in nodewise regressions (cf. Theorem 15). We then bound event $\mathcal{X}_0$ in Lemma 13. Both proofs appear in Section A.3.

Lemma 12 *Suppose that $p < e^{n/4C_2^2}$. Then with probability at least $1 - 1/p^2$, we have*

$$\forall j \neq k, \ \left| \frac{1}{n} \langle V_j, X_k \rangle \right| \leq \sigma_{V_j} \sqrt{1+a} \sqrt{(2\log p)/n}, \tag{44}$$

where $\sigma_{V_j} = \sqrt{\mathrm{Var}(V_j)}$ and $a \geq 6$. Hence $\mathbb{P}(\mathcal{C}_a) \geq 1 - 1/p^2$.

Lemma 13 *For a random design X as in (1) with $\Sigma_{0,jj} = 1, \forall j \in \{1,\ldots,p\}$, and for $p < e^{n/4C_3^2}$, where $C_3 > 4\sqrt{5/3}$, we have*

$$\mathbb{P}(\mathcal{X}_0) \geq 1 - 1/\max\{n,p\}^2.$$

We note that the upper bounds on p in Lemma 12 and 13 clearly hold given (A1). For the rest of the paper, we prove Theorem 15 in Section B for nodewise regressions. We proceed to derive bounds on selecting an edge set E in Section C. We then derive various bounds on the maximum likelihood estimator given E in Theorem 19- 21 in Section D, where we also prove Theorem 1. Next, we prove Lemma 12 and 13 in Section A.3.

A.3 Proof of Lemma 12 and 13

We first state the following large inequality bound on products of correlated normal random variables.

Lemma 14 Zhou et al., 2008, Lemma 38 *Given a set of identical independent random variables $Y_1,\ldots,Y_n \sim Y$, where $Y = x_1 x_2$, with $x_1, x_2 \sim N(0,1)$ and $\sigma_{12} = \rho_{12}$ with $\rho_{12} \leq 1$ being their correlation coefficient. Let us now define $Q = \frac{1}{n}\sum_{i=1}^n Y_i =: \frac{1}{n}\langle X_1, X_2 \rangle = \frac{1}{n}\sum_{i=1}^n x_{1,i} x_{2,i}$. Let $\Psi_{12} =$*

$(1+\sigma_{12}^2)/2$. *For* $0 \leq \tau \leq \Psi_{12}$,

$$\mathbb{P}(|Q - \mathbb{E}Q| > \tau) \leq \exp\left\{-\frac{3n\tau^2}{10(1+\sigma_{12}^2)}\right\}. \tag{45}$$

Proof of Lemma 12. It is clear that event (44) is the same as event $\mathcal{C}_a$. Clearly we have at most $p(p-1)$ unique entries $Z_{jk}, \forall j \neq k$. By the union bound and by taking $\tau = C_2\sqrt{\frac{\log p}{n}}$ in (45) with $\sigma_{jk} = 0, \forall j,k$, where $\sqrt{2(1+a)} \geq C_2 > 2\sqrt{10/3}$ for $a \geq 6$.

$$\begin{aligned}
1 - \mathbb{P}(\mathcal{C}_a) &= \mathbb{P}\left(\max_{jk}|Z_{jk}| \geq \sqrt{2(1+a)}\sqrt{\frac{\log p}{n}}\right) \\
&\leq \mathbb{P}\left(\max_{jk}|Z_{jk}| \geq C_2\sqrt{\frac{\log p}{n}}\right) \leq (p^2-p)\exp\left(-\frac{3C_2^2\log p}{10}\right) \\
&\leq p^2\exp\left(-\frac{3C_2^2\log p}{10}\right) = p^{-\frac{3C_2^2}{10}+2} < \frac{1}{p^2},
\end{aligned}$$

where we apply Lemma 14 with $\rho_{jk} = 0, \forall j,k = 1,\ldots,p, j \neq k$ and use the fact that $\mathbb{E}Z_{jk} = 0$. Note that $p < e^{n/4C_2^2}$ guarantees that $C_2\sqrt{\frac{\log p}{n}} < 1/2$. ■

In order to bound the probability of event $\mathcal{X}_0$, we first state the following bound for the non-diagonal entries of Σ_0, which follows immediately from Lemma 14 by plugging in $\sigma_i^2 = \sigma_{0,ii} = 1, \forall i = 1,\ldots,p$ and using the fact that $|\sigma_{0,jk}| = |\rho_{jk}\sigma_j\sigma_k| \leq 1, \forall j \neq k$, where ρ_{jk} is the correlation coefficient between variables X_j and X_k. Let $\Psi_{jk} = (1+\sigma_{0,jk}^2)/2$. Then

$$\mathbb{P}\left(|\Delta_{jk}| > \tau\right) \leq \exp\left\{-\frac{3n\tau^2}{10(1+\sigma_{0,jk}^2)}\right\} \leq \exp\left\{-\frac{3n\tau^2}{20}\right\} \text{ for } 0 \leq \tau \leq \Psi_{jk}. \tag{46}$$

We now also state a large deviation bound for the χ_n^2 distribution [Johnstone, 2001]:

$$\mathbb{P}\left(\frac{\chi_n^2}{n} - 1 > \tau\right) \leq \exp\left(\frac{-3n\tau^2}{16}\right), \text{ for } 0 \leq \tau \leq \frac{1}{2}. \tag{47}$$

Lemma 13 follows from (46) and (47) immediately.

Proof of Lemma 13. Now it is clear that we have $p(p-1)/2$ unique non-diagonal entries $\sigma_{0,jk}, \forall j \neq k$ and p diagonal entries. By the union bound and by taking $\tau = C_3\sqrt{\frac{\log\max\{p,n\}}{n}}$ in (47) and (46) with $\sigma_{0,jk} \leq 1$, we have

$$\begin{aligned}
\mathbb{P}((\mathcal{X}_0)^c) &= \mathbb{P}\left(\max_{jk}|\Delta_{jk}| \geq C_3\sqrt{\frac{\log\max\{p,n\}}{n}}\right) \\
&\leq p\exp\left(-\frac{3C_3^2\log\max\{p,n\}}{16}\right) + \frac{p^2-p}{2}\exp\left(-\frac{3C_3^2\log\max\{p,n\}}{20}\right) \\
&\leq p^2\exp\left(-\frac{3C_3^2\log\max\{p,n\}}{20}\right) = (\max\{p,n\})^{-\frac{3C_3^2}{20}+2} < \frac{1}{(\max\{p,n\})^2}
\end{aligned}$$

for $C_3 > 4\sqrt{5/3}$, where for the diagonal entries we use (47), and for the non-diagonal entries, we use (46). Finally, $p < e^{n/4C_3^2}$ guarantees that $C_3\sqrt{\frac{\log\max\{p,n\}}{n}} < 1/2$. ■

Appendix B. Bounds for Nodewise Regressions

In Theorem 15 and Lemma 16 we let s_0^i be as in (35) and T_0^i denote locations of the s_0^i largest coefficients of β^i in absolute values. For the vector h^i to be defined in Theorem 15, we let T_1^i denote the s_0^i largest positions of h^i in absolute values outside of T_0^i; Let $T_{01}^i := T_0^i \cup T_1^i$. We suppress the superscript in T_0^i, T_1^i, and T_{01}^i throughout this section for clarity.

Theorem 15 (Oracle inequalities of the nodewise regressions) *Let $0 < \theta < 1$. Let $\rho_{\min}(s) > 0$, where $s < p$ is the maximum node-degree in G. Suppose $RE(s_0, 4, \Sigma_0)$ holds for $s_0 \le s$ as in* (36)*, where $\Sigma_{0,ii} = 1$ forall i. Suppose $\rho_{\max}(\max(s, 3s_0)) < \infty$. The data is generated by $X^{(1)}, \ldots, X^{(n)}$ i.i.d. $\sim \mathcal{N}_p(0, \Sigma_0)$, where the sample size n satisfies* (42)*.*

Consider the nodewise regressions in (10)*, where for each i, we regress X_i onto the other variables $\{X_k;\ k \neq i\}$ following* (2)*, where $V_i \sim N(0, \mathrm{Var}(V_i))$ is independent of $X_j, \forall j \neq i$ as in* (3)*.*

Let β_{init}^i be an optimal solution to (10) *for each i. Let $\lambda_n = d_0\lambda = d_0^i\lambda\sigma_{V_i}$ where d_0 is chosen such that $d_0 \ge 2(1+\theta)\sqrt{1+a}$ holds for some $a \ge 6$. Let $h^i = \beta_{\text{init}}^i - \beta_{T_0}^i$. Then simultaneously for all i, on $\mathcal{C}_a \cap \mathcal{X}$, where $\mathcal{X} := \mathcal{R}(\theta) \cap \mathcal{F}(\theta) \cap \mathcal{M}(\theta)$, we have*

$$\begin{aligned} \left\|\beta_{\text{init}}^i - \beta^i\right\|_2 &\le \lambda\sqrt{s_0^i} d_0 \sqrt{2D_0^2 + 2D_1^2 + 2}, \text{ where} \\ \|h_{T_{01}}\|_2 &\le D_0 d_0 \lambda \sqrt{s_0^i} \quad \text{and} \quad \left\|h_{T_0^c}^i\right\|_1 = \left\|\beta_{\text{init},T_0^c}^i\right\|_1 \le D_1 d_0 \lambda s_0^i, \end{aligned} \tag{48}$$

where D_0, D_1 are defined in (82) *and* (83) *respectively.*

Suppose we choose for some constant $c_0 \ge 4\sqrt{2}$ and $a_0 = 7$,

$$d_0 = c_0(1+\theta)^2\sqrt{\rho_{\max}(s)\rho_{\max}(3s_0)},$$

where we assume that $\rho_{\max}(\max(s, 3s_0)) < \infty$ is reasonably bounded. Then

$$D_0 \le \frac{5K^2(s_0, 4, \Sigma_0)}{(1-\theta)^2} \text{ and } D_1 \le \frac{49K^2(s_0, 4, \Sigma_0)}{16(1-\theta)^2}.$$

The choice of d_0 will be justified in Section F, where we also the upper bound on D_0, D_1 as above.

Proof Consider each regression function in (10) with $X_{\cdot\setminus i}$ being the design matrix and X_i the response vector, where $X_{\cdot\setminus i}$ denotes columns of X excluding X_i. It is clear that for $\lambda_n = d_0\lambda$, we have for $i = 1, \ldots, p$ and $a \ge 6$,

$$\lambda_n = (d_0/\sigma_{V_i})\sigma_{V_i}\lambda := d_0^i\sigma_{V_i}\lambda \ge d_0\lambda\sigma_{V_i} \ge 2(1+\theta)\lambda\sqrt{1+a}\sigma_{V_i} = 2(1+\theta)\lambda_{\sigma,a,p}$$

such that (81) holds given that $\sigma_{V_i} \le 1, \forall i$, where it is understood that $\sigma := \sigma_{V_i}$.

It is also clear that on $\mathcal{C}_a \cap \mathcal{X}$, event $\mathcal{T}_a \cap \mathcal{X}$ holds for each regression when we invoke Theorem 33, with $Y := X_i$ and $X := X_{\setminus i}$, for $i = 1, \ldots, p$. By definition $d_0^i \sigma_{V_i} = d_0$. We can then invoke bounds for each individual regression as in Theorem 33 to conclude. ■

Appendix C. Bounds on Thresholding

In this section, we first show Lemma 16, following conditions in Theorem 15. We then show Corollary 17, which proves Proposition 4 and the first statement of Theorem 1. D_0, D_1 are the same constants as in Theorem 15.

Lemma 16 *Suppose $RE(s_0, 4, \Sigma_0)$ holds for s_0 be as in* (36) *and $\rho_{\min}(s) > 0$, where $s < p$ is the maximum node-degree in G. Suppose $\rho_{\max}(\max(s, 3s_0)) < \infty$. Let $S^i = \{j : j \neq i,\ \beta_j^i \neq 0\}$. Let $c_0 \geq 4\sqrt{2}$ be some absolute constant. Suppose n satisfies* (42). *Let β^i_{init} be an optimal solution to* (10) *with*

$$\lambda_n = d_0 \lambda \ \text{ where } \ d_0 = c_0 (1+\theta)^2 \sqrt{\rho_{\max}(s) \rho_{\max}(3s_0)};$$

Suppose for each regression, we apply the same thresholding rule to obtain a subset I^i as follows,

$$I^i = \{j : j \neq i,\ \left|\beta^i_{j,\text{init}}\right| \geq t_0 = f_0 \lambda\}, \ \text{ and } \ \mathcal{D}^i := \{1, \ldots, i-1, i+1, \ldots, p\} \setminus I^i,$$

where $f_0 := D_4 d_0$ for some constant D_4 to be specified. Then we have on event $\mathcal{C}_a \cap \mathcal{X}$,

$$\begin{aligned} |I^i| &\leq s_0^i (1 + D_1/D_4) \text{ and } |I^i \cup S^i| \leq s^i + (D_1/D_4) s_0^i, \text{ and} \\ \left\|\beta^i_{\mathcal{D}}\right\|_2 &\leq d_0 \lambda \sqrt{s_0^i} \sqrt{1 + (D_0 + D_4)^2}, \end{aligned} \tag{49}$$

where $\mathcal{D}$ is understood to be $\mathcal{D}^i$.

Recall $\Theta_0 = \Sigma_0^{-1}$. Let $\Theta_{0,\mathcal{D}}$ denote the submatrix of Θ_0 indexed by $\mathcal{D}$ as in (22) with all other positions set to be 0. Let E_0 be the true edge set.

Corollary 17 *Suppose all conditions in Lemma 16 hold. Then on event $\mathcal{C}_a \cap \mathcal{X}$, for $\widetilde{\Theta}_0$ as in* (24) *and E as in* (23), *we have for $S_{0,n}$ as in* (36) *and $\Theta_0 = (\theta_{0,ij})$*

$$|E| \leq (1 + D_1/D_4) S_{0,n} \quad \text{where} \quad |E \setminus E_0| \leq (D_1/D_4) S_{0,n}, \tag{50}$$

and

$$\begin{aligned} \left\|\Theta_{0,\mathcal{D}}\right\|_F &:= \left\|\widetilde{\Theta}_0 - \Theta_0\right\|_F \\ &\leq \sqrt{\min\{S_{0,n}(\max_{i=1,\ldots p} \theta^2_{0,ii}), s_0 \left\|\text{diag}(\Theta_0)\right\|_F^2\}} \sqrt{(1 + (D_0 + D_4)^2)} d_0 \lambda \\ &:= \sqrt{S_{0,n}\,(1 + (D_0 + D_4)^2)} C_{\text{diag}} d_0 \lambda, \end{aligned} \tag{51}$$

where $C^2_{\text{diag}} := \min\{\max_{i=1,\ldots p} \theta^2_{0,ii}, (s_0/S_{0,n}) \left\|\text{diag}(\Theta_0)\right\|_F^2\}$. For $D_4 \geq D_1$, we have (18).

Proof By the OR rule in (9), we could select at most $\sum_{i=1}^{p}|I_i|$ edges. We have by (49)

$$|E| \leq \sum_{i=1,\ldots p}(1+D_1/D_4)s_0^i = (1+D_1/D_4)\,S_{0,n},$$

where $(D_1/D_4)S_{0,n}$ is an upper bound on $|E\setminus E_0|$ by (52). Thus

$$\begin{aligned}\left\|\Theta_{0,\mathcal{D}}\right\|_F^2 &\leq \sum_{i=1}^{p}\theta_{0,ii}^2\left\|\beta_{\mathcal{D}}^i\right\|_2^2 \leq (1+(D_0+D_4)^2)d_0^2\lambda^2\sum_{i=1}^{p}\theta_{0,ii}^2 s_0^i \\ &\leq \min\{S_{0,n}(\max_{i=1,\ldots p}\theta_{0,ii}^2), s_0\left\|\mathrm{diag}(\Theta_0)\right\|_F^2\}(1+(D_0+D_4)^2)d_0^2\lambda^2.\end{aligned}$$

■

Remark 18 *Note that if s_0 is small, then the second term in C_{diag} will provide a tighter bound.*

Proof of Lemma 16. Let $T_0 := T_0^i$ denote the s_0^i largest coefficients of β^i in absolute values. We have by (48),

$$|I^i\cap T_0^c| \leq \left\|\beta_{\mathrm{init},T_0^c}^i\right\|_1\frac{1}{f_0\lambda} \leq D_1 d_0 s_0^i/(D_4 d_0) \leq D_1 s_0^i/D_4, \tag{52}$$

where D_1 is understood to be the same constant that appears in (48). Thus we have

$$\left|I^i\right| = |I^i\cap T_0^c| + |I^i\cap T_0| \leq s_0^i(1+D_1/D_4).$$

Now the second inequality in (49) clearly holds given (52) and the following:

$$|I^i\cup S^i| \leq |S^i| + |I^i\cap(S^i)^c| \leq s^i + |I^i\cap(T_0^i)^c|.$$

We now bound $\left\|\beta_{\mathcal{D}}^i\right\|_2^2$ following essentially the arguments as in Zhou [2009]. We have

$$\left\|\beta_{\mathcal{D}}^i\right\|_2^2 = \left\|\beta_{T_0\cap\mathcal{D}}^i\right\|_2^2 + \left\|\beta_{T_0^c\cap\mathcal{D}}^i\right\|_2^2,$$

where for the second term, we have $\left\|\beta_{T_0^c\cap\mathcal{D}}^i\right\|_2^2 \leq \left\|\beta_{T_0^c}^i\right\|_2^2 \leq s_0^i\lambda^2\sigma_{V_i}^2$ by definition of s_0^i as in (35) and (38); For the first term, we have by the triangle inequality and (48),

$$\begin{aligned}\left\|\beta_{T_0\cap\mathcal{D}}^i\right\|_2 &\leq \left\|(\beta^i-\beta_{\mathrm{init}}^i)_{T_0\cap\mathcal{D}}\right\|_2 + \left\|(\beta_{\mathrm{init}}^i)_{T_0\cap\mathcal{D}}\right\|_2 \\ &\leq \left\|(\beta^i-\beta_{\mathrm{init}}^i)_{T_0}\right\|_2 + t_0\sqrt{|T_0\cap\mathcal{D}|} \leq \left\|h_{T_0}\right\|_2 + t_0\sqrt{s_0^i} \\ &\leq D_0 d_0\lambda\sqrt{s_0^i} + D_4 d_0\lambda\sqrt{s_0^i} \leq (D_0+D_4)d_0\lambda\sqrt{s_0^i}.\end{aligned}$$

■

Appendix D. Bounds on MLE Refitting

Recall the maximum likelihood estimate $\widehat{\Theta}_n$ minimizes over all $\Theta \in \mathcal{S}_n$ the empirical risk:

$$\widehat{\Theta}_n(E) = \arg\min_{\Theta \in \mathcal{S}_n} \widehat{R}_n(\Theta) := \arg \min_{\Theta \in \mathcal{S}_{++}^p \cap \mathcal{S}_E^p} \left\{ \mathrm{tr}(\Theta \widehat{\Gamma}_n) - \log|\Theta| \right\}, \tag{53}$$

which gives the "best" refitted sparse estimator given a sparse subset of edges E that we obtain from the nodewise regressions and thresholding. We note that the estimator (53) remains to be a convex optimization problem, as the constraint set is the intersection the positive definite cone $\mathcal{S}_{++}^p$ and the linear subspace $\mathcal{S}_E^p$. Implicitly, by using $\widehat{\Gamma}_n$ rather than $\widehat{S}_n$ in (53), we force the diagonal entries in $(\widehat{\Theta}_n(E))^{-1}$ to be identically 1. It is not hard to see that the estimator (53) is equivalent to (13), after we replace $\widehat{S}_n$ with $\widehat{\Gamma}_n$.

Theorem 19 *Consider data generating random variables as in expression (15) and assume that* (A1), (33), *and* (34) *hold. Suppose* $\Sigma_{0,ii} = 1$ *for all* i. *Let* $\mathcal{E}$ *be some event such that* $\mathbb{P}(\mathcal{E}) \geq 1 - d/p^2$ *for a small constant* d. *Let* $S_{0,n}$ *be as defined in* (36)*; Suppose on event* $\mathcal{E}$*:*

1. *We obtain an edge set* E *such that its size* $|E| = lin(S_{0,n})$ *is a linear function in* $S_{0,n}$.
2. *And for* $\widetilde{\Theta}_0$ *as in* (24) *and for some constant* C_{bias} *to be specified, we have*

$$\left\|\Theta_{0,\mathcal{D}}\right\|_F := \left\|\widetilde{\Theta}_0 - \Theta_0\right\|_F \leq C_{\mathrm{bias}} \sqrt{2S_{0,n}\log(p)/n} < \underline{c}/32. \tag{54}$$

Let $\widehat{\Theta}_n(E)$ *be as defined in* (53). *Suppose the sample size satisfies for* $C_3 \geq 4\sqrt{5/3}$,

$$n > \frac{106}{\underline{k}^2}\left(4C_3 + \frac{32}{31\underline{c}^2}\right)^2 \max\left\{2|E|\log\max(n,p),\, C_{\mathrm{bias}}^2 2S_{0,n}\log p\right\}. \tag{55}$$

Then on event $\mathcal{E} \cap \mathcal{X}_0$, *we have for* $M = (9/(2\underline{k}^2)) \cdot \left(4C_3 + 32/(31\underline{c}^2)\right)$

$$\left\|\widehat{\Theta}_n(E) - \Theta_0\right\|_F \leq (M+1)\max\left\{\sqrt{2|E|\log\max(n,p)/n},\, C_{\mathrm{bias}}\sqrt{2S_{0,n}\log(p)/n}\right\}. \tag{56}$$

We note that although Theorem 19 is meant for proving Theorem 1, we state it as an independent result; For example, one can indeed take E from Corollary 17, where we have $|E| \leq cS_{0,n}$ for some constant c for $D_4 \asymp D_1$. In view of (51), we aim to recover $\widetilde{\Theta}_0$ by $\widehat{\Theta}_n(E)$ as defined in (53). In Section D.2, we will focus in Theorem 19 on bounding for W suitably chosen,

$$\left\|\widehat{\Theta}_n(E) - \widetilde{\Theta}_0\right\|_F = O_P\left(W\sqrt{S_{0,n}\log\max(n,p)/n}\right).$$

By the triangle inequality, we conclude that

$$\left\|\widehat{\Theta}_n(E) - \Theta_0\right\|_F \leq \left\|\widehat{\Theta}_n(E) - \widetilde{\Theta}_0\right\|_F + \left\|\widetilde{\Theta}_0 - \Theta_0\right\|_F = O_P\left(W\sqrt{S_{0,n}\log(n)/n}\right).$$

We now state bounds for the convergence rate on Frobenius norm of the covariance matrix and for KL divergence. We note that constants have not been optimized. Proofs of Theorem 20 and 21 appear in Section D.3 and D.4 respectively.

Theorem 20 *Suppose all conditions, events, and bounds on $|E|$ and $\left\|\Theta_{0,\mathcal{D}}\right\|_F$ in Theorem 19 hold. Let $\widehat{\Theta}_n(E)$ be as defined in* (53)*. Suppose the sample size satisfies for $C_3 \geq 4\sqrt{5/3}$ and C_{bias}, M as defined in Theorem 19*

$$n > \frac{106}{\underline{c}^2\underline{k}^4}\left(4C_3 + \frac{32}{31\underline{c}^2}\right)^2 \max\left\{2|E|\log\max(p,n),\, C_{\text{bias}}^2 2S_{0,n}\log p\right\}. \tag{57}$$

Then on event $\mathcal{E}\cap\mathcal{X}_0$, we have $\varphi_{\min}(\widehat{\Theta}_n(E)) > \underline{c}/2 > 0$ and for $\widehat{\Sigma}_n(E) = (\widehat{\Theta}_n(E))^{-1}$,

$$\left\|\widehat{\Sigma}_n(E) - \Sigma_0\right\|_F \leq \frac{2(M+1)}{\underline{c}^2}\max\left\{\sqrt{\frac{2|E|\log\max(n,p)}{n}},\, C_{\text{bias}}\sqrt{\frac{2S_{0,n}\log(p)}{n}}\right\}. \tag{58}$$

Theorem 21 *Suppose all conditions, events, and bounds on $|E|$ and $\left\|\Theta_{0,\mathcal{D}}\right\|_F := \left\|\widetilde{\Theta}_0 - \Theta_0\right\|_F$ in Theorem 19 hold. Let $\widehat{\Theta}_n(E)$ be as defined in* (53)*. Suppose the sample size satisfies* (55) *for $C_3 \geq 4\sqrt{5/3}$ and C_{bias}, M as defined in Theorem 19. Then on event $\mathcal{E}\cap\mathcal{X}_0$, we have for $R(\widehat{\Theta}_n(E)) - R(\Theta_0) \geq 0$,*

$$R(\widehat{\Theta}_n(E)) - R(\Theta_0) \leq M(C_3 + 1/8)\max\left\{2|E|\log\max(n,p)/n,\, C_{\text{bias}}^2 2S_{0,n}\log(p)/n\right\}.$$

D.1 Proof of Theorem 1

Clearly the sample requirement as in (42) is satisfied for some $\theta > 0$ that is appropriately chosen, given (55). In view of Corollary 17, we have on $\mathcal{E} := \mathcal{X}\cap\mathcal{C}_a$: for C_{diag} as in (17)

$$\begin{aligned} |E| &\leq (1 + \frac{D_1}{D_4})S_{0,n} \leq 2S_{0,n} \text{ for } D_4 \geq D_1 \text{ and} \\ \left\|\Theta_{0,\mathcal{D}}\right\|_F &:= \left\|\widetilde{\Theta}_0 - \Theta_0\right\|_F \leq C_{\text{bias}}\sqrt{2S_{0,n}\log(p)/n} \leq \underline{c}/32, \end{aligned}$$

where

$$\begin{aligned} C_{\text{bias}}^2 &:= \min\left\{\max_{i=1,\ldots p}\theta_{0,ii}^2, \frac{s_0}{S_{0,n}}\left\|\text{diag}(\Theta_0)\right\|_F^2\right\} d_0^2(1 + (D_0+D_4)^2) \\ &= C_{\text{diag}}^2 d_0^2(1 + (D_0+D_4)^2). \end{aligned} \tag{59}$$

Clearly the last inequality in (54) hold so long as $n > 32^2 C_{\text{bias}}^2 2S_{0,n}\log(p)/\underline{c}^2$, which holds given (55). Plugging in $|E|$ in (56), we have on $\mathcal{E}\cap\mathcal{X}_0$,

$$\left\|\widehat{\Theta}_n(E) - \Theta_0\right\|_F \leq (M+1)\max\left\{\sqrt{\frac{2(1+D_1/D_4)S_{0,n}\log\max(n,p)}{n}},\, C_{\text{bias}}\sqrt{\frac{2S_{0,n}\log p}{n}}\right\}.$$

Now if we take $D_4 \geq D_1$, then we have (18) on event $\mathcal{E}$; and moreover on $\mathcal{E}\cap\mathcal{X}_0$,

$$\begin{aligned} \left\|\widehat{\Theta}_n(E) - \Theta_0\right\|_F &\leq (M+1)\max\left\{\sqrt{4S_{0,n}\log\max(n,p)/n},\, C_{\text{bias}}\sqrt{2S_{0,n}\log(p)/n}\right\} \\ &\leq W\sqrt{S_{0,n}\log\max(n,p)/n}, \end{aligned}$$

where $W \leq \sqrt{2}(M+1)\max\{C_{\text{diag}} d_0 \sqrt{1+(D_0+D_4)^2}, 2\}$. Similarly, we get the bound on $\left\|\widehat{\Sigma}_n - \Sigma_0\right\|_F$ with Theorem 20, and the bound on risk following Theorem 21. Thus all statements in Theorem 1 hold. ■

Remark 22 *Suppose event $\mathcal{E} \cap \mathcal{X}_0$ holds. Now suppose that we take $D_4 = 1$, that is, if we take the threshold to be exactly the penalty parameter λ_n:*

$$t_0 = d_0 \lambda := \lambda_n.$$

Then we have on event $\mathcal{E}$, $|E| \leq (1+D_1)S_{0,n}$ and $|E \setminus E_0| \leq D_1 S_{0,n}$ by (50)*; And on event $\mathcal{E} \cap \mathcal{X}_0$, for $C'_{\text{bias}} := C_{\text{diag}} d_0 \sqrt{1+(D_0+1)^2}$,*

$$\left\|\widehat{\Theta}_n(E) - \Theta_0\right\|_F \leq M \max\left\{\sqrt{\frac{2(1+D_1)S_{0,n}\log\max(n,p)}{n}},\, C'_{\text{bias}}\sqrt{\frac{2S_{0,n}\log p}{n}}\right\}.$$

It is not hard to see that we achieve essential the same rate as stated in Theorem 1, with perhaps slightly more edges included in E.

D.2 Proof of Theorem 19

Suppose event $\mathcal{E}$ holds throughout this proof. We first obtain the bound on spectrum of $\widetilde{\Theta}_0$: It is clear that by (33) and (54), we have on $\mathcal{E}$,

$$\varphi_{\min}(\widetilde{\Theta}_0) \;\geq\; \varphi_{\min}(\Theta_0) - \left\|\widetilde{\Theta}_0 - \Theta_0\right\|_2 \geq \varphi_{\min}(\Theta_0) - \left\|\Theta_{0,\mathcal{D}}\right\|_F > 31\underline{c}/32, \tag{60}$$

$$\varphi_{\max}(\widetilde{\Theta}_0) \;<\; \varphi_{\max}(\Theta_0) + \left\|\widetilde{\Theta}_0 - \Theta_0\right\|_2 \leq \varphi_{\max}(\Theta_0) + \left\|\Theta_{0,\mathcal{D}}\right\|_F < \frac{\underline{c}}{32} + \frac{1}{\underline{k}}. \tag{61}$$

Throughout this proof, we let $\Sigma_0 = (\sigma_{0,ij}) := \Theta_0^{-1}$. In view of (60), define $\widetilde{\Sigma}_0 := (\widetilde{\Theta}_0)^{-1}$. We use $\widehat{\Theta}_n := \widehat{\Theta}_n(E)$ as a shorthand.

Given $\widetilde{\Theta}_0 \in \mathcal{S}^p_{++} \cap \mathcal{S}^p_E$ as guaranteed in (60), let us define a new convex set:

$$U_n(\widetilde{\Theta}_0) := (\mathcal{S}^p_{++} \cap \mathcal{S}^p_E) - \widetilde{\Theta}_0 = \{B - \widetilde{\Theta}_0 | B \in \mathcal{S}^p_{++} \cap \mathcal{S}^p_E\} \subset \mathcal{S}^p_E,$$

which is a translation of the original convex set $\mathcal{S}^p_{++} \cap \mathcal{S}^p_E$. Let $\underline{0}$ be a matrix with all entries being zero. Thus it is clear that $U_n(\widetilde{\Theta}_0) \ni \underline{0}$ given that $\widetilde{\Theta}_0 \in \mathcal{S}^p_{++} \cap \mathcal{S}^p_E$. Define for $\widehat{R}_n$ as in expression (53)

$$\begin{aligned}\widetilde{Q}(\Theta) \;&:=\; \widehat{R}_n(\Theta) - \widehat{R}_n(\widetilde{\Theta}_0) = \operatorname{tr}(\Theta\widehat{\Gamma}_n) - \log|\Theta| - \operatorname{tr}(\widetilde{\Theta}_0\widehat{\Gamma}_n) + \log|\widetilde{\Theta}_0| \\ &=\; \operatorname{tr}\left((\Theta - \widetilde{\Theta}_0)(\widehat{\Gamma}_n - \widetilde{\Sigma}_0)\right) - (\log|\Theta| - \log|\widetilde{\Theta}_0|) + \operatorname{tr}\left((\Theta - \widetilde{\Theta}_0)\widetilde{\Sigma}_0\right).\end{aligned}$$

For an appropriately chosen r_n and a large enough $M > 0$, let

$$\begin{aligned}\mathbb{T}_n \;&=\; \{\Delta \in U_n(\widetilde{\Theta}_0), \|\Delta\|_F = Mr_n\}, \text{ and} \\ \Pi_n \;&=\; \{\Delta \in U_n(\widetilde{\Theta}_0), \|\Delta\|_F < Mr_n\}.\end{aligned}$$

It is clear that both Π_n and $\mathbb{T}_n \cup \Pi_n$ are convex. It is also clear that $\underline{0} \in \Pi_n$. Throughout this section, we let

$$r_n = \max\left\{\sqrt{\frac{2|E|\log\max(n,p)}{n}}, C_{\text{bias}}\sqrt{\frac{2S_{0,n}\log p}{n}}\right\}. \tag{62}$$

Define for $\Delta \in U_n(\widetilde{\Theta}_0)$,

$$\widetilde{G}(\Delta) := \widetilde{Q}(\widetilde{\Theta}_0 + \Delta) = \text{tr}(\Delta(\widehat{\Gamma}_n - \widetilde{\Sigma}_0)) - (\log|\widetilde{\Theta}_0 + \Delta| - \log|\widetilde{\Theta}_0|) + \text{tr}(\Delta\widetilde{\Sigma}_0).$$

It is clear that $\widetilde{G}(\Delta)$ is a convex function on $U_n(\widetilde{\Theta}_0)$ and $\widetilde{G}(\underline{0}) = \widetilde{Q}(\widetilde{\Theta}_0) = 0$.

Now, $\widehat{\Theta}_n$ minimizes $\widetilde{Q}(\Theta)$, or equivalently $\widehat{\Delta} = \widehat{\Theta}_n - \widetilde{\Theta}_0$ minimizes $\widetilde{G}(\Delta)$. Hence by definition,

$$\widetilde{G}(\widehat{\Delta}) \le \widetilde{G}(\underline{0}) = 0.$$

Note that $\mathbb{T}_n$ is non-empty, while clearly $\underline{0} \in \Pi_n$. Indeed, consider $B_\varepsilon := (1+\varepsilon)\widetilde{\Theta}_0$, where $\varepsilon > 0$; it is clear that $B_\varepsilon - \widetilde{\Theta}_0 \in \mathcal{S}^p_{++} \cap \mathcal{S}^p_E$ and $\left\|B_\varepsilon - \widetilde{\Theta}_0\right\|_F = |\varepsilon| \left\|\widetilde{\Theta}_0\right\|_F = Mr_n$ for $|\varepsilon| = Mr_n / \left\|\widetilde{\Theta}_0\right\|_F$. Note also if $\Delta \in \mathbb{T}_n$, then $\Delta_{ij} = 0 \forall (i,j : i \ne j) \notin E$; Thus we have $\Delta \in \mathcal{S}^p_E$ and

$$\|\Delta\|_0 = \|\text{diag}(\Delta)\|_0 + \|\text{offd}(\Delta)\|_0 \le p + 2|E| \text{ where } |E| = \text{lin}(S_{0,n}).$$

We now show the following two propositions. Proposition 23 follows from standard results.

Proposition 23 *Let B be a $p \times p$ matrix. If $B \succ 0$ and $B + D \succ 0$, then $B + vD \succ 0$ for all $v \in [0,1]$.*

Proposition 24 *Under* (33)*, we have for all $\Delta \in \mathbb{T}_n$ such that $\|\Delta\|_F = Mr_n$ for r_n as in* (62)*, $\widetilde{\Theta}_0 + v\Delta \succ 0, \forall v \in$ an open interval $I \supset [0,1]$ on event $\mathcal{E}$.*

Proof In view of Proposition 23, it is sufficient to show that $\widetilde{\Theta}_0 + (1+\varepsilon)\Delta, \widetilde{\Theta}_0 - \varepsilon\Delta \succ 0$ for some $\varepsilon > 0$. Indeed, by definition of $\Delta \in \mathbb{T}_n$, we have $\varphi_{\min}(\widetilde{\Theta}_0 + \Delta) \succ 0$ on event $\mathcal{E}$; thus

$$\begin{aligned} \varphi_{\min}(\widetilde{\Theta}_0 + (1+\varepsilon)\Delta) &\ge \varphi_{\min}(\widetilde{\Theta}_0 + \Delta) - \varepsilon\|\Delta\|_2 > 0, \\ \text{and } \varphi_{\min}(\widetilde{\Theta}_0 - \varepsilon\Delta) &\ge \varphi_{\min}(\widetilde{\Theta}_0) - \varepsilon\|\Delta\|_2 > 31\underline{c}/32 - \varepsilon\|\Delta\|_2 > 0 \end{aligned}$$

for $\varepsilon > 0$ that is sufficiently small. ■

Thus we have that $\log|\widetilde{\Theta}_0 + v\Delta|$ is infinitely differentiable on the open interval $I \supset [0,1]$ of v. This allows us to use the Taylor's formula with integral remainder to obtain the following:

Lemma 25 *On event $\mathcal{E} \cap \mathcal{X}_0$, $\widetilde{G}(\Delta) > 0$ for all $\Delta \in \mathbb{T}_n$.*

Proof Let us use $\widetilde{A}$ as a shorthand for

$$\text{vec}\Delta^T \left(\int_0^1 (1-v)(\widetilde{\Theta}_0 + v\Delta)^{-1} \otimes (\widetilde{\Theta}_0 + v\Delta)^{-1} dv\right) \text{vec}\Delta,$$

where $\otimes$ is the Kronecker product (if $W = (w_{ij})_{m\times n}, P = (b_{k\ell})_{p\times q}$, then $W \otimes P = (w_{ij}P)_{mp\times nq}$), and $\text{vec}\Delta \in \mathbb{R}^{p^2}$ is $\Delta_{p\times p}$ vectorized. Now, the Taylor expansion gives for all $\Delta \in \mathbb{T}_n$,

$$\begin{aligned} \log|\widetilde{\Theta}_0 + \Delta| - \log|\widetilde{\Theta}_0| &= \frac{d}{dv}\log|\widetilde{\Theta}_0 + v\Delta||_{v=0}\Delta + \int_0^1 (1-v)\frac{d^2}{dv^2}\log|\widetilde{\Theta}_0 + v\Delta| dv \\ &= \text{tr}(\widetilde{\Sigma}_0\Delta) - \widetilde{A}. \end{aligned}$$

Hence for all $\Delta \in \mathbb{T}_n$,

$$\widetilde{G}(\Delta) = \widetilde{A} + \mathrm{tr}\left(\Delta(\widehat{\Gamma}_n - \widetilde{\Sigma}_0)\right) = \widetilde{A} + \mathrm{tr}\left(\Delta(\widehat{\Gamma}_n - \Sigma_0)\right) - \mathrm{tr}\left(\Delta(\widetilde{\Sigma}_0 - \Sigma_0)\right), \tag{63}$$

where we first bound $\mathrm{tr}(\Delta(\widetilde{\Sigma}_0 - \Sigma_0))$ as follows: by (54) and (60), we have on event $\mathcal{E}$

$$\begin{aligned}
\left|\mathrm{tr}(\Delta(\widetilde{\Sigma}_0 - \Sigma_0))\right| &= \left|\langle \Delta, (\widetilde{\Sigma}_0 - \Sigma_0)\rangle\right| \le \|\Delta\|_F \left\|\widetilde{\Sigma}_0 - \Sigma_0\right\|_F \\
&\le \|\Delta\|_F \frac{\left\|\Theta_{0,\mathcal{D}}\right\|_F}{\varphi_{\min}(\widetilde{\Theta}_0)\varphi_{\min}(\Theta_0)} \\
&< \|\Delta\|_F \frac{32 C_{\mathrm{bias}}\sqrt{2S_{0,n}\log p/n}}{31\underline{c}^2} \le \|\Delta\|_F \frac{32 r_n}{31\underline{c}^2}.
\end{aligned} \tag{64}$$

Conditioned on event $\mathcal{X}_0$, by (70) and (55)

$$\max_{j,k} |\widehat{\Gamma}_{n,jk} - \sigma_{0,jk}| \le 4C_3\sqrt{\log\max(n,p)/n} =: \delta_n.$$

Thus on event $\mathcal{E} \cap X_0$, we have $\left|\mathrm{tr}\big(\Delta(\widehat{\Gamma}_n - \Sigma_0)\big)\right| \le \delta_n |\mathrm{offd}(\Delta)|_1$, where

$$|\mathrm{offd}(\Delta)|_1 \le \sqrt{\|\mathrm{offd}(\Delta)\|_0}\,\|\mathrm{offd}(\Delta)\|_F \le \sqrt{2|E|}\,\|\Delta\|_F,$$

and

$$\mathrm{tr}\left(\Delta(\widehat{\Gamma}_n - \Sigma_0)\right) \ge -4C_3\sqrt{\log\max(n,p)/n}\sqrt{2|E|}\,\|\Delta\|_F \ge -4C_3 r_n \|\Delta\|_F. \tag{65}$$

Finally, we bound $\widetilde{A}$. First we note that for $\Delta \in \mathbb{T}_n$, we have on event $\mathcal{E}$,

$$\|\Delta\|_2 \le \|\Delta\|_F = M r_n < \frac{7}{16\underline{k}}, \tag{66}$$

given (55): $n > (\frac{16}{7}\cdot\frac{9}{2\underline{k}})^2\left(4C_3 + \frac{32}{31\underline{c}^2}\right)^2 \max\left\{(2|E|)\log(n),\, C_{\mathrm{bias}}^2 2S_{0,n}\log p\right\}$. Now we have by (61) and (34) following Rothman et al. [2008] (see Page 502, proof of Theorem 1 therein): on event $\mathcal{E}$,

$$\begin{aligned}
\widetilde{A} &\ge \|\Delta\|_F^2 \Big/ \left(2\left(\varphi_{\max}(\widetilde{\Theta}_0) + \|\Delta\|_2\right)^2\right) \\
&\ge \|\Delta\|_F^2 \Big/ \left(2(\frac{1}{\underline{k}} + \frac{\underline{c}}{32} + \frac{7}{16\underline{k}})^2\right) > \|\Delta\|_F^2 \frac{2\underline{k}^2}{9}.
\end{aligned} \tag{67}$$

Now on event $\mathcal{E} \cap \mathcal{X}_0$, for all $\Delta \in \mathbb{T}_n$, we have by (63),(67), (65), and (64),

$$\begin{aligned}
\widetilde{G}(\Delta) &> \|\Delta\|_F^2 \frac{2\underline{k}^2}{9} - 4C_3 r_n \|\Delta\|_F - \|\Delta\|_F \frac{32 r_n}{31\underline{c}^2} \\
&= \|\Delta\|_F^2 \left(\frac{2\underline{k}^2}{9} - \frac{1}{\|\Delta\|_F}\left(4C_3 r_n + \frac{32 r_n}{31\underline{c}^2}\right)\right) \\
&= \|\Delta\|_F^2 \left(\frac{2\underline{k}^2}{9} - \frac{1}{M}\left(4C_3 + \frac{32}{31\underline{c}^2}\right)\right),
\end{aligned}$$

hence we have $\widetilde{G}(\Delta) > 0$ for M large enough, in particular $M = (9/(2\underline{k}^2))\left(4C_3 + 32/(31\underline{c}^2)\right)$ suffices. ■

We next state Proposition 26, which follows exactly that of Claim 12 of Zhou et al. [2008].

Proposition 26 *Suppose event $\mathcal{E}$ holds. If $\widetilde{G}(\Delta) > 0, \forall \Delta \in \mathbb{T}_n$, then $\widetilde{G}(\Delta) > 0$ for all Δ in*

$$\mathbb{W}_n = \{\Delta : \Delta \in U_n(\widetilde{\Theta}_0), \|\Delta\|_F > Mr_n\}$$

for r_n as in (62)*; Hence if $\widetilde{G}(\Delta) > 0$ for all $\Delta \in \mathbb{T}_n$, then $\widetilde{G}(\Delta) > 0$ for all $\Delta \in \mathbb{T}_n \cup \mathbb{W}_n$.*

Note that for $\widehat{\Theta}_n \in \mathcal{S}^p_{++} \cap \mathcal{S}^p_E$, we have $\widehat{\Delta} = \widehat{\Theta}_n - \widetilde{\Theta}_0 \in U_n(\widetilde{\Theta}_0)$. By Proposition 26 and the fact that $\widetilde{G}(\widehat{\Delta}) \leq \widetilde{G}(\underline{0}) = 0$ on event $\mathcal{E}$, we have the following: on event $\mathcal{E}$, if $\widetilde{G}(\Delta) > 0, \forall \Delta \in \mathbb{T}_n$ then $\|\widehat{\Delta}\|_F < Mr_n$, given that $\widehat{\Delta} \in U_n(\widetilde{\Theta}_0) \setminus (\mathbb{T}_n \cup \mathbb{W}_n)$. Therefore

$$\begin{aligned}
\mathbb{P}\left(\|\widehat{\Delta}\|_F \geq Mr_n\right) &\leq \mathbb{P}(\mathcal{E}^c) + \mathbb{P}(\mathcal{E}) \cdot \mathbb{P}\left(\|\widehat{\Delta}\|_F \geq Mr_n | \mathcal{E}\right) \\
&= \mathbb{P}(\mathcal{E}^c) + \mathbb{P}(\mathcal{E}) \cdot (1 - \mathbb{P}\left(\|\widehat{\Delta}\|_F < Mr_n | \mathcal{E}\right)) \\
&\leq \mathbb{P}(\mathcal{E}^c) + \mathbb{P}(\mathcal{E}) \cdot (1 - \mathbb{P}\left(\widetilde{G}(\Delta) > 0, \forall \Delta \in \mathbb{T}_n | \mathcal{E}\right)) \\
&\leq \mathbb{P}(\mathcal{E}^c) + \mathbb{P}(\mathcal{E}) \cdot (1 - \mathbb{P}(\mathcal{X}_0 | \mathcal{E})) \\
&= \mathbb{P}(\mathcal{E}^c) + \mathbb{P}(\mathcal{X}_0^c \cap \mathcal{E}) \leq \mathbb{P}(\mathcal{E}^c) + \mathbb{P}(\mathcal{X}_0^c) \\
&\leq \frac{c}{p^2} + \frac{1}{\max(n,p)^2} \leq \frac{c+1}{p^2}.
\end{aligned}$$

We thus establish that the theorem holds. ■

D.3 Frobenius Norm for the Covariance Matrix

We use the bound on $\left\|\widehat{\Theta}_n(E) - \Theta_0\right\|_F$ as developed in Theorem 19; in addition, we strengthen the bound on Mr_n in (66) in (68). Before we proceed, we note the following bound on bias of $(\widetilde{\Theta}_0)^{-1}$.

Remark 27 *Clearly we have on event $\mathcal{E}$, by* (64)

$$\left\|(\widetilde{\Theta}_0)^{-1} - \Sigma_0\right\|_F \leq \frac{\left\|\Theta_{0,\mathcal{D}}\right\|_F}{\varphi_{\min}(\widetilde{\Theta}_0)\varphi_{\min}(\Theta_0)} \leq \frac{32C_{\text{bias}}\sqrt{2S_{0,n}\log p/n}}{31\underline{c}^2}.$$

Proof of Theorem 20. Suppose event $\mathcal{E} \cap \mathcal{X}_0$ holds. Now suppose

$$n > (\frac{16}{7\underline{c}} \cdot \frac{9}{2\underline{k}^2})^2 \left(C_3 + \frac{32}{31\underline{c}^2}\right)^2 \max\left\{2|E|\log\max(n,p),\, C^2_{\text{bias}} 2S_{0,n}\log p\right\},$$

which clearly holds given (57). Then in addition to the bound in (66), on event $\mathcal{E} \cap \mathcal{X}_0$, we have

$$Mr_n < 7\underline{c}/16, \tag{68}$$

for r_n as in (62). Then, by Theorem 19, for the same M as therein, on event $\mathcal{E} \cap \mathcal{X}_0$, we have

$$\left\|\widehat{\Theta}_n(E) - \Theta_0\right\|_F \leq (M+1)\max\left\{\sqrt{2|E|\log\max(n,p)/n},\, C_{\text{bias}}\sqrt{2S_{0,n}\log(p)/n}\right\},$$

given that sample bound in (55) is clearly satisfied. We now proceed to bound $\left\|\widehat{\Sigma}_n - \Sigma_0\right\|_F$ given (56). First note that by (68), we have on event $\mathcal{E} \cap \mathcal{X}_0$ for $M > 7$

$$\begin{aligned}\varphi_{\min}(\widehat{\Theta}_n(E)) &\geq \varphi_{\min}(\Theta_0) - \left\|\widehat{\Theta}_n - \Theta_0\right\|_2 \geq \varphi_{\min}(\Theta_0) - \left\|\widehat{\Theta}_n - \Theta_0\right\|_F \\ &\geq \underline{c} - (M+1)r_n > \underline{c}/2.\end{aligned}$$

Now clearly on event $\mathcal{E} \cap \mathcal{X}_0$, (58) holds by (56) and

$$\left\|\widehat{\Sigma}_n(E) - \Sigma_0\right\|_F \leq \frac{\left\|\widehat{\Theta}_n(E) - \Theta_0\right\|_F}{\varphi_{\min}(\widehat{\Theta}_n(E))\varphi_{\min}(\Theta_0)} < \frac{2}{\underline{c}^2}\left\|\widehat{\Theta}_n(E) - \Theta_0\right\|_F.$$

■

D.4 Risk Consistency

We now derive the bound on risk consistency. Before proving Theorem 21, we first state two lemmas given the following decomposition of our loss in terms of the risk as defined in (16):

$$0 \leq R(\widehat{\Theta}_n(E)) - R(\Theta_0) = (R(\widehat{\Theta}_n(E)) - R(\widetilde{\Theta}_0)) + (R(\widetilde{\Theta}_0) - R(\Theta_0)), \tag{69}$$

where clearly $R(\widehat{\Theta}_n(E)) \geq R(\Theta_0)$ by definition. It is clear that $\widetilde{\Theta}_0 \in \mathcal{S}_n$ for $\mathcal{S}_n$ as defined in (28), and thus $\widehat{R}_n(\widetilde{\Theta}_0) \geq \widehat{R}_n(\widehat{\Theta}_n(E))$ by definition of $\widehat{\Theta}_n(E) = \arg\min_{\Theta \in \mathcal{S}_n} \widehat{R}_n(\Theta)$.

We now bound the two terms on the RHS of (69), where clearly $R(\widetilde{\Theta}_0) \geq R(\Theta_0)$.

Lemma 28 *On event $\mathcal{E}$, we have for $C_{\text{bias}}, \Theta_0, \widetilde{\Theta}_0$ as in Theorem 19,*

$$0 \leq R(\widetilde{\Theta}_0) - R(\Theta_0) \leq (32/(31\underline{c}))^2 C_{\text{bias}}^2 \frac{2S_{0,n}\log p}{2n} \leq (32/(31\underline{c}))^2 \cdot r_n^2/2 \leq Mr_n^2/8,$$

for r_n as in (62)*, where the last inequality holds given that $M \geq 9/2(4C_3 + 32/(31\underline{c}^2))$.*

Lemma 29 *Under $\mathcal{E} \cap \mathcal{X}_0$, we have for r_n as in* (62) *and M, C_3 as in Theorem 19*

$$R(\widehat{\Theta}_n(E)) - R(\widetilde{\Theta}_0) \leq MC_3 r_n^2.$$

Proof of Theorem 21. We have on $\mathcal{E} \cap \mathcal{X}_0$, for r_n is as in (62)

$$R(\widehat{\Theta}_n(E)) - R(\Theta_0) = (R(\widehat{\Theta}_n(E)) - R(\widetilde{\Theta}_0)) + (R(\widetilde{\Theta}_0) - R(\Theta_0)) \leq Mr_n^2(C_3 + 1/8)$$

as desired, using Lemma 28 and 29. ■

Proof of Lemma 28. For simplicity, we use Δ_0 as a shorthand for the rest of our proof:

$$\Delta_0 := \Theta_{0,\mathcal{D}} = \widetilde{\Theta}_0 - \Theta_0.$$

We use $\widetilde{B}$ as a shorthand for

$$\mathrm{vec}\Delta_0{}^T \left(\int_0^1 (1-v)(\Theta_0 + v\Delta_0)^{-1} \otimes (\Theta_0 + v\Delta_0)^{-1} dv \right) \mathrm{vec}\Delta_0,$$

where $\otimes$ is the Kronecker product. First, we have for $\widetilde{\Theta}_0, \Theta_0 \succ 0$

$$\begin{aligned} R(\widetilde{\Theta}_0) - R(\Theta_0) &= \mathrm{tr}(\widetilde{\Theta}_0\Sigma_0) - \log|\widetilde{\Theta}_0| - \mathrm{tr}(\Theta_0\Sigma_0) + \log|\Theta_0| \\ &= \mathrm{tr}((\widetilde{\Theta}_0 - \Theta_0)\Sigma_0) - \left(\log|\widetilde{\Theta}_0| - \log|\Theta_0|\right) := \widetilde{B} \geq 0, \end{aligned}$$

where $\widetilde{B} = 0$ holds when $\|\Delta_0\|_F = 0$, and in the last equation, we bound the difference between two $\log|\cdot|$ terms using the Taylor's formula with integral remainder following that in proof of Theorem 19. Indeed, it is clear that on $\mathcal{E}$, we have

$$\Theta_0 + v\Delta_0 \succ 0 \ \text{ for } \ v \in (-1,2) \supset [0,1],$$

given that $\varphi_{\min}(\Theta_0) \geq \underline{c}$ and $\|\Delta_0\|_2 \leq \|\Delta_0\|_F \leq \underline{c}/32$ by (54). Thus $\log|\Theta_0 + v\Delta_0|$ is infinitely differentiable on the open interval $I \supset [0,1]$ of v. Now, the Taylor expansion gives

$$\begin{aligned} \log|\Theta_0 + \Delta_0| - \log|\Theta_0| &= \frac{d}{dv}\log|\Theta_0 + v\Delta_0||_{v=0}\Delta_0 + \int_0^1 (1-v)\frac{d^2}{dv^2}\log|\Theta_0 + v\Delta_0| dv \\ &= \mathrm{tr}(\Sigma_0\Delta_0) - \widetilde{B}. \end{aligned}$$

We now obtain an upper bound on $\widetilde{B} \geq 0$. Clearly, we have on event $\mathcal{E}$, Lemma 28 holds given that

$$\widetilde{B} \leq \|\Delta_0\|_F^2 \cdot \varphi_{\max}\left(\int_0^1 (1-v)(\Theta_0 + v\Delta_0)^{-1} \otimes (\Theta_0 + v\Delta_0)^{-1} dv \right),$$

where $\|\Delta_0\|_F^2 \leq C_{\mathrm{bias}}^2 2S_{0,n}\log(p)/n$ and

$$\begin{aligned} &\varphi_{\max}\left(\int_0^1 (1-v)(\Theta_0 + v\Delta_0)^{-1} \otimes (\Theta_0 + v\Delta_0)^{-1} dv \right) \\ \leq\ & \int_0^1 (1-v)\varphi_{\max}^2(\Theta_0 + v\Delta_0)^{-1} dv \leq \sup_{v\in[0,1]} \varphi_{\max}^2(\Theta_0 + v\Delta_0)^{-1} \int_0^1 (1-v) dv \\ =\ & \frac{1}{2}\sup_{v\in[0,1]} \frac{1}{\varphi_{\min}^2(\Theta_0 + v\Delta_0)} = \frac{1}{2\inf_{v\in[0,1]}\varphi_{\min}^2(\Theta_0 + v\Delta_0)} \\ \leq\ & \frac{1}{2\left(\varphi_{\min}(\Theta_0) - \|\Delta_0\|_2\right)^2} \leq \frac{1}{2\left(31\underline{c}/32\right)^2}, \end{aligned}$$

where clearly for all $v \in [0,1]$, we have $\varphi_{\min}^2(\Theta_0 + v\Delta_0) \geq \left(\varphi_{\min}(\Theta_0) - \|\Delta_0\|_2\right)^2 \geq \left(31\underline{c}/32\right)^2$, given $\varphi_{\min}(\Theta_0) \geq \underline{c}$ and $\|\Delta_0\|_2 \leq \left\|\Theta_{0,\mathcal{D}}\right\|_F \leq \underline{c}/32$ by (54). ■

Proof of Lemma 29. Suppose $R(\widehat{\Theta}_n(E)) - R(\widetilde{\Theta}_0) < 0$, then we are done.

Otherwise, assume $R(\widehat{\Theta}_n(E)) - R(\widetilde{\Theta}_0) \geq 0$ throughout the rest of the proof. Define

$$\widehat{\Delta} := \widehat{\Theta}_n(E) - \widetilde{\Theta}_0,$$

which by Theorem 19, we have on event $\mathcal{E} \cap \mathcal{X}_0$, and for M as defined therein,

$$\left\|\widehat{\Delta}\right\|_F := \left\|\widehat{\Theta}_n(E) - \widetilde{\Theta}_0\right\|_F \leq Mr_n.$$

We have by definition $\widehat{R}_n(\widehat{\Theta}_n(E)) \leq \widehat{R}_n(\widetilde{\Theta}_0)$, and hence

$$\begin{aligned} 0 \leq R(\widehat{\Theta}_n(E)) - R(\widetilde{\Theta}_0) &= R(\widehat{\Theta}_n(E)) - \widehat{R}_n(\widehat{\Theta}_n(E)) + \widehat{R}_n(\widehat{\Theta}_n(E)) - R(\widetilde{\Theta}_0) \\ &\leq R(\widehat{\Theta}_n(E)) - \widehat{R}_n(\widehat{\Theta}_n(E)) + \widehat{R}_n(\widetilde{\Theta}_0) - R(\widetilde{\Theta}_0) \\ &= \mathrm{tr}(\widehat{\Theta}_n(E)(\Sigma_0 - \widehat{\Gamma}_n)) - \mathrm{tr}(\widetilde{\Theta}_0(\Sigma_0 - \widehat{\Gamma}_n)) \\ &= \mathrm{tr}((\widehat{\Theta}_n(E) - \widetilde{\Theta}_0)(\Sigma_0 - \widehat{\Gamma}_n)) = \mathrm{tr}(\widehat{\Delta}(\Sigma_0 - \widehat{\Gamma}_n)). \end{aligned}$$

Now, conditioned on event $\mathcal{E} \cap \mathcal{X}_0$, following the same arguments around (65), we have

$$\begin{aligned} \left|\mathrm{tr}\left(\widehat{\Delta}(\widehat{S}_n - \Sigma_0)\right)\right| &\leq \delta_n \left|\mathrm{offd}(\widehat{\Delta})\right|_1 \leq \delta_n \sqrt{2|E|} \left\|\mathrm{offd}(\widehat{\Delta})\right\|_F \\ &\leq Mr_n C_3 \sqrt{2|E| \log \max(n,p)/n} \leq MC_3 r_n^2, \end{aligned}$$

where $\left\|\mathrm{offd}(\widehat{\Delta})\right\|_0 \leq 2|E|$ by definition, and r_n is as defined in (62). ■

Appendix E. Proof of Theorem 6

We first bound $\mathbb{P}(\mathcal{X}_0)$ in Lemma 30, which follows exactly that of Lemma 13 as the covariance matrix Ψ_0 for variables $X_1/\sigma_1, \ldots, X_p/\sigma_p$ satisfy the condition that $\Psi_{0,ii} = 1, \forall i \in \{1, \ldots, p\}$.

Lemma 30 *For $p < e^{n/4C_3^2}$, where $C_3 > 4\sqrt{5/3}$, we have for X_0 as defined in* (43)

$$\mathbb{P}(\mathcal{X}_0) \geq 1 - 1/\max\{n,p\}^2.$$

On event $\mathcal{X}_0$, the following holds for $\tau = C_3\sqrt{\frac{\log\max\{p,n\}}{n}} < 1/2$, where we assume $p < e^{n/4C_3^2}$,

$$\begin{aligned} \forall i, \quad \left|\frac{\|X_i\|_2^2}{\sigma_i^2 n} - 1\right| &\leq \tau, \\ \forall i \neq j, \quad \left|\frac{1}{n}\langle X_i/\sigma_i, X_j/\sigma_j\rangle - \rho_{0,ij}\right| &\leq \tau. \end{aligned}$$

Let us first derive the large deviation bound for $\left|\widehat{\Gamma}_{n,ij}-\rho_{0,ij}\right|$. First note that on event $\mathcal{X}_0$ $\sqrt{1-\tau}\le \|X_i\|_2/(\sigma_i\sqrt{n})\le\sqrt{1+\tau}$ and for all $i\neq j$

$$\begin{aligned}
\left|\widehat{\Gamma}_{n,ij}-\rho_{0,ij}\right| &= \left|\frac{\widehat{S}_{n,ij}}{\widehat{\sigma}_i\widehat{\sigma}_j}-\rho_{0,ij}\right| := \left|\widehat{\rho}_{ij}-\rho_{0,ij}\right| \\
&= \left|\frac{\frac{1}{n}\langle X_i/\sigma_i, X_j/\sigma_j\rangle-\rho_{0,ij}}{(\|X_i\|_2/(\sigma_i\sqrt{n}))\cdot(\|X_j\|_2/(\sigma_j\sqrt{n}))}+\frac{\rho_{0,ij}}{(\|X_i\|_2/(\sigma_i\sqrt{n}))\cdot(\|X_j\|_2/(\sigma_j\sqrt{n}))}-\rho_{0,ij}\right| \\
&\le \left|\frac{\frac{1}{n}\langle X_i/\sigma_i, X_j/\sigma_j\rangle-\rho_{0,ij}}{(\|X_i\|_2/(\sigma_i\sqrt{n}))\cdot(\|X_j\|_2/(\sigma_j\sqrt{n}))}\right|+\left|\frac{\rho_{0,ij}}{(\|X_i\|_2/(\sigma_i\sqrt{n}))\cdot(\|X_j\|_2/(\sigma_j\sqrt{n}))}-\rho_{0,ij}\right| \\
&\le \frac{\tau}{1-\tau}+|\rho_{0,ij}|\left|\frac{1}{1-\tau}-1\right|\le\frac{2\tau}{1-\tau}<4\tau. \qquad (70)
\end{aligned}$$

Proof of Theorem 6. For $\widetilde{\Theta}_0$ as in (24), we define

$$\begin{aligned}
\widetilde{\Omega}_0 &= W\widetilde{\Theta}_0W = W(\mathrm{diag}(\Theta_0))W+W\Theta_{0,E_0\cap E}W \\
&= \mathrm{diag}(W\Theta_0W)+W\Theta_{0,E_0\cap E}W=\mathrm{diag}(\Omega_0)+\Omega_{0,E_0\cap E},
\end{aligned}$$

where $W=\mathrm{diag}(\Sigma_0)^{1/2}$. Then clearly $\widetilde{\Omega}_0\in\mathcal{S}_n$ as $\widetilde{\Theta}_0\in\mathcal{S}_n$. We first bound $\left\|\Theta_{0,\mathcal{D}}\right\|_F$ as follows.

$$\begin{aligned}
\left\|\Theta_{0,\mathcal{D}}\right\|_F &\le C_{\mathrm{bias}}\sqrt{2S_{0,n}\log(p)/n}<\frac{\underline{k}}{\sqrt{144}\sigma_{\max}^2\left(4C_3+\frac{13}{12\underline{c}^2\sigma_{\min}^2}\right)} \\
&\le \frac{\underline{k}\underline{c}^2\sigma_{\min}^2}{(48\underline{c}^2\sigma_{\min}^2C_3+13)\sigma_{\max}^2}\le\min\left\{\frac{\underline{k}}{48C_3\sigma_{\max}^2},\frac{\underline{c}\sigma_{\min}^2}{13\sigma_{\max}^2}\right\}\le\frac{\underline{c}}{13\sigma_{\max}^2}.
\end{aligned}$$

Suppose event $\mathcal{E}$ holds throughout this proof. We first obtain the bound on spectrum of $\widetilde{\Theta}_0$: It is clear that by (33) and (30), we have on $\mathcal{E}$,

$$\varphi_{\min}(\widetilde{\Theta}_0) \ge \varphi_{\min}(\Theta_0)-\left\|\widetilde{\Theta}_0-\Theta_0\right\|_2\ge\varphi_{\min}(\Theta_0)-\left\|\Theta_{0,\mathcal{D}}\right\|_F>\frac{12\underline{c}}{13}, \qquad (71)$$

$$\varphi_{\max}(\widetilde{\Theta}_0) < \varphi_{\max}(\Theta_0)+\left\|\widetilde{\Theta}_0-\Theta_0\right\|_2\le\varphi_{\max}(\Theta_0)+\left\|\Theta_{0,\mathcal{D}}\right\|_F<\frac{\underline{c}}{13\sigma_{\max}^2}+\frac{1}{\underline{k}}. \qquad (72)$$

Throughout this proof, we let $\Sigma_0=(\sigma_{0,ij}):=\Theta_0^{-1}$. In view of (71), define $\widetilde{\Sigma}_0:=(\widetilde{\Theta}_0)^{-1}$. Then

$$\widetilde{\Omega}_0^{-1}=W^{-1}(\widetilde{\Theta}_0)^{-1}W^{-1}=W^{-1}\widetilde{\Sigma}_0W^{-1}:=\widetilde{\Psi}_0.$$

We use $\widehat{\Omega}_n:=\widehat{\Omega}_n(E)$ as a shorthand. Thus we have for $\widetilde{\Omega}_0=W\widetilde{\Theta}_0W$,

$$\begin{aligned}
\varphi_{\max}(\widetilde{\Omega}_0) &\le \varphi_{\max}(W)\varphi_{\max}(\widetilde{\Theta}_0)\varphi_{\max}(W)\le\frac{\sigma_{\max}^2}{\underline{k}}+\frac{\underline{c}}{13} \\
\varphi_{\min}(\widetilde{\Omega}_0), &= \frac{1}{\varphi_{\max}(\widetilde{\Psi}_0)}=\frac{1}{\varphi_{\max}(W^{-1}\widetilde{\Sigma}_0W^{-1})}=\frac{1}{\varphi_{\max}(W^{-1})^2\varphi_{\max}(\widetilde{\Sigma}_0)} \\
&= \frac{\varphi_{\min}(W)^2}{\varphi_{\max}(\widetilde{\Sigma}_0)}=\varphi_{\min}(W)^2\varphi_{\min}(\widetilde{\Theta}_0)\ge\sigma_{\min}^2\frac{12\underline{c}}{13}. \qquad (73)
\end{aligned}$$

Given $\widetilde{\Omega}_0 \in \mathcal{S}_{++}^p \cap \mathcal{S}_E^p$ as guaranteed in (73), let us define a new convex set:

$$U_n(\widetilde{\Omega}_0) := (\mathcal{S}_{++}^p \cap \mathcal{S}_E^p) - \widetilde{\Omega}_0 = \{B - \widetilde{\Omega}_0 | B \in \mathcal{S}_{++}^p \cap \mathcal{S}_E^p\} \subset \mathcal{S}_E^p,$$

which is a translation of the original convex set $\mathcal{S}_{++}^p \cap \mathcal{S}_E^p$. Let $\underline{0}$ be a matrix with all entries being zero. Thus it is clear that $U_n(\widetilde{\Omega}_0) \ni \underline{0}$ given that $\widetilde{\Omega}_0 \in \mathcal{S}_{++}^p \cap \mathcal{S}_E^p$. Define for $\widehat{R}_n$ as in expression (27),

$$\begin{aligned} \widetilde{Q}(\Omega) &:= \widehat{R}_n(\Omega) - \widehat{R}_n(\widetilde{\Omega}_0) = \mathrm{tr}(\Omega\widehat{\Gamma}_n) - \log|\Omega| - \mathrm{tr}(\widetilde{\Omega}_0\widehat{\Gamma}_n) + \log|\widetilde{\Omega}_0| \\ &= \mathrm{tr}\Big((\Omega - \widetilde{\Omega}_0)(\widehat{\Gamma}_n - \widetilde{\Psi}_0)\Big) - (\log|\Omega| - \log|\widetilde{\Omega}_0|) + \mathrm{tr}\Big((\Omega - \widetilde{\Omega}_0)\widetilde{\Psi}_0\Big). \end{aligned}$$

For an appropriately chosen r_n and a large enough $M > 0$, let

$$\begin{aligned} \mathbb{T}_n &= \{\Delta \in U_n(\widetilde{\Omega}_0), \|\Delta\|_F = Mr_n\}, \text{ and} \\ \Pi_n &= \{\Delta \in U_n(\widetilde{\Omega}_0), \|\Delta\|_F < Mr_n\}. \end{aligned}$$

Both Π_n and $\mathbb{T}_n \cup \Pi_n$ are convex. It is clear that $\underline{0} \in \Pi_n$. Define for $\Delta \in U_n(\widetilde{\Omega}_0)$,

$$\widetilde{G}(\Delta) := \widetilde{Q}(\widetilde{\Omega}_0 + \Delta) = \mathrm{tr}(\Delta(\widehat{\Gamma}_n - \widetilde{\Psi}_0)) - (\log|\widetilde{\Omega}_0 + \Delta| - \log|\widetilde{\Omega}_0|) + \mathrm{tr}(\Delta\widetilde{\Psi}_0).$$

Thus $\widetilde{G}(\Delta)$ is a convex function on $U_n(\widetilde{\Omega}_0)$ and $\widetilde{G}(\underline{0}) = \widetilde{Q}(\widetilde{\Omega}_0) = 0$.

Now, $\widehat{\Omega}_n$ minimizes $\widetilde{Q}(\Omega)$, or equivalently $\widehat{\Delta} = \widehat{\Omega}_n - \widetilde{\Omega}_0$ minimizes $\widetilde{G}(\Delta)$. Hence by definition,

$$\widetilde{G}(\widehat{\Delta}) \leq \widetilde{G}(\underline{0}) = 0.$$

Note that $\mathbb{T}_n$ is non-empty, while clearly $\underline{0} \in \Pi_n$. Indeed, consider $B_\varepsilon := (1+\varepsilon)\widetilde{\Omega}_0$, where $\varepsilon > 0$; it is clear that $B_\varepsilon - \widetilde{\Omega}_0 \in \mathcal{S}_{++}^p \cap \mathcal{S}_E^p$ and $\left\|B_\varepsilon - \widetilde{\Omega}_0\right\|_F = |\varepsilon| \left\|\widetilde{\Omega}_0\right\|_F = Mr_n$ for $|\varepsilon| = Mr_n / \left\|\widetilde{\Omega}_0\right\|_F$. Note also if $\Delta \in \mathbb{T}_n$, then $\Delta_{ij} = 0 \forall (i,j : i \neq j) \notin E$; Thus we have $\Delta \in \mathcal{S}_E^p$ and

$$\|\Delta\|_0 = \|\mathrm{diag}(\Delta)\|_0 + \|\mathrm{offd}(\Delta)\|_0 \leq p + 2|E| \text{ where } |E| = \mathrm{lin}(S_{0,n}).$$

We now show the following proposition.

Proposition 31 *Under* (33)*, we have for all* $\Delta \in \mathbb{T}_n$ *such that* $\|\Delta\|_F = Mr_n$ *for* r_n *as in* (62)*,* $\widetilde{\Omega}_0 + v\Delta \succ 0, \forall v \in$ *an open interval* $I \supset [0,1]$ *on event* $\mathcal{E}$*.*

Proof In view of Proposition 23, it is sufficient to show that $\widetilde{\Omega}_0 + (1+\varepsilon)\Delta, \widetilde{\Omega}_0 - \varepsilon\Delta \succ 0$ for some $\varepsilon > 0$. Indeed, by definition of $\Delta \in \mathbb{T}_n$, we have $\varphi_{\min}(\widetilde{\Omega}_0 + \Delta) \succ 0$ on event $\mathcal{E}$; thus

$$\begin{aligned} \varphi_{\min}(\widetilde{\Omega}_0 + (1+\varepsilon)\Delta) &\geq \varphi_{\min}(\widetilde{\Omega}_0 + \Delta) - \varepsilon\|\Delta\|_2 > 0, \\ \text{and } \varphi_{\min}(\widetilde{\Omega}_0 - \varepsilon\Delta) &\geq \varphi_{\min}(\widetilde{\Omega}_0) - \varepsilon\|\Delta\|_2 > 12\sigma_{\min}^2 \underline{c}/13 - \varepsilon\|\Delta\|_2 > 0 \end{aligned}$$

for $\varepsilon > 0$ that is sufficiently small. ■

Thus we have that $\log|\widetilde{\Omega}_0 + v\Delta|$ is infinitely differentiable on the open interval $I \supset [0,1]$ of v. This allows us to use the Taylor's formula with integral remainder to obtain the following:

Lemma 32 *On event $\mathcal{E}\cap\mathcal{X}_0$, $\widetilde{G}(\Delta)>0$ for all $\Delta\in\mathbb{T}_n$.*

Proof Let us use $\widetilde{A}$ as a shorthand for

$$\mathrm{vec}\Delta^T\left(\int_0^1(1-v)(\widetilde{\Omega}_0+v\Delta)^{-1}\otimes(\widetilde{\Omega}_0+v\Delta)^{-1}dv\right)\mathrm{vec}\Delta,$$

where $\otimes$ is the Kronecker product (if $W=(w_{ij})_{m\times n}$, $P=(b_{k\ell})_{p\times q}$, then $W\otimes P=(w_{ij}P)_{mp\times nq}$), and $\mathrm{vec}\Delta\in\mathbb{R}^{p^2}$ is $\Delta_{p\times p}$ vectorized. Now, the Taylor expansion gives for all $\Delta\in\mathbb{T}_n$,

$$\begin{aligned}\log|\widetilde{\Omega}_0+\Delta|-\log|\widetilde{\Omega}_0| &= \frac{d}{dv}\log|\widetilde{\Omega}_0+v\Delta||_{v=0}\Delta+\int_0^1(1-v)\frac{d^2}{dv^2}\log|\widetilde{\Omega}_0+v\Delta|dv\\ &= \mathrm{tr}(\widetilde{\Psi}_0\Delta)-\widetilde{A}.\end{aligned}$$

Hence for all $\Delta\in\mathbb{T}_n$,

$$\widetilde{G}(\Delta)=\widetilde{A}+\mathrm{tr}\left(\Delta(\widehat{\Gamma}_n-\widetilde{\Psi}_0)\right)=\widetilde{A}+\mathrm{tr}\left(\Delta(\widehat{\Gamma}_n-\Psi_0)\right)-\mathrm{tr}\left(\Delta(\widetilde{\Psi}_0-\Psi_0)\right),\tag{74}$$

where we first bound $\mathrm{tr}(\Delta(\widetilde{\Psi}_0-\Psi_0))$ as follows: by (30) and (60), we have on event $\mathcal{E}$

$$\begin{aligned}\left|\mathrm{tr}(\Delta(\widetilde{\Psi}_0-\Psi_0))\right| &= \left|\langle\Delta,(\widetilde{\Psi}_0-\Psi_0)\rangle\right|\le\|\Delta\|_F\left\|\widetilde{\Psi}_0-\Psi_0\right\|_F\\ &\le \|\Delta\|_F\frac{13r_n}{12\sigma^2_{\min}\underline{c}^2},\end{aligned}\tag{75}$$

where we bound $\left\|\widetilde{\Psi}_0-\Psi_0\right\|_F$ as follows:

$$\begin{aligned}\left\|\widetilde{\Psi}_0-\Psi_0\right\|_F &= \left\|W^{-1}(\widetilde{\Sigma}_0-\Sigma_0)W^{-1}\right\|_F\le\max_i W_i^{-2}\left\|\widetilde{\Sigma}_0-\Sigma_0\right\|_F\\ &\le \frac{1}{\sigma^2_{\min}}\frac{\left\|\Theta_{0,\mathcal{D}}\right\|_F}{\varphi_{\min}(\widetilde{\Theta}_0)\varphi_{\min}(\Theta_0)}\\ &\le \frac{C_{\mathrm{bias}}\sqrt{2S_{0,n}\log p/n}}{12\sigma^2_{\min}\underline{c}^2/13}\le\frac{13r_n}{12\sigma^2_{\min}\underline{c}^2}.\end{aligned}$$

Now, conditioned on event $\mathcal{X}_0$, by (70)

$$\max_{j,k}|\widehat{\Gamma}_{n,jk}-\rho_{0,jk}|\le 4C_3\sqrt{\log\max(n,p)/n}=:\delta_n$$

and thus on event $\mathcal{E}\cap X_0$, we have $\left|\mathrm{tr}\big(\Delta(\widehat{\Gamma}_n-\Psi_0)\big)\right|\le\delta_n|\mathrm{offd}(\Delta)|_1$, where $|\mathrm{offd}(\Delta)|_1\le\sqrt{\|\mathrm{offd}(\Delta)\|_0}\,\|\mathrm{offd}(\Delta)\|_F\le\sqrt{2|E|}\,\|\Delta\|_F$, and

$$\mathrm{tr}\left(\Delta(\widehat{\Gamma}_n-\Psi_0)\right)\ge-4C_3\sqrt{\log\max(n,p)/n}\sqrt{2|E|}\,\|\Delta\|_F\ge-4C_3r_n\|\Delta\|_F.\tag{76}$$

Finally, we bound $\widetilde{A}$. First we note that for $\Delta\in\mathbb{T}_n$, we have on event $\mathcal{E}$,

$$\|\Delta\|_2\le\|\Delta\|_F=Mr_n<\frac{3\sigma^2_{\max}}{8\underline{k}},$$

given (31): $n > (\frac{8}{3}\cdot\frac{9}{2\underline{k}})^2\sigma_{\max}^4\left(4C_3+\frac{13}{12\sigma_{\min}^2\underline{c}^2}\right)^2\max\left\{2|E|)\log\max(n,p), C_{\text{bias}}^2 2S_{0,n}\log p\right\}$. We have by (72) and (34) following Rothman et al. [2008] (see Page 502, proof of Theorem 1 therein): on event $\mathcal{E}$,

$$\begin{aligned}\widetilde{A} &\geq \|\Delta\|_F^2 / \left(2\left(\varphi_{\max}(\widetilde{\Omega}_0)+\|\Delta\|_2\right)^2\right)\\ &> \|\Delta\|_F^2 / \left(2\sigma_{\max}^4\left(\frac{1}{\underline{k}}+\frac{\underline{c}}{13}+\frac{3}{8\underline{k}}\right)^2\right) > \|\Delta\|_F^2\,\frac{2\underline{k}^2}{9\sigma_{\max}^4}. \end{aligned} \tag{77}$$

Now on event $\mathcal{E}\cap\mathcal{X}_0$, for all $\Delta\in\mathbb{T}_n$, we have by (74),(77), (76), and (75),

$$\begin{aligned}\widetilde{G}(\Delta) &> \|\Delta\|_F^2\,\frac{2\underline{k}^2}{9\sigma_{\max}^4} - 4C_3 r_n\|\Delta\|_F - \|\Delta\|_F\,\frac{13 r_n}{12\sigma_{\min}^2\underline{c}^2}\\ &= \|\Delta\|_F^2\left(\frac{2\underline{k}^2}{9\sigma_{\max}^4}-\frac{1}{\|\Delta\|_F}\left(4C_3 r_n+\frac{13 r_n}{12\sigma_{\min}^2\underline{c}^2}\right)\right)\\ &= \|\Delta\|_F^2\left(\frac{2\underline{k}^2}{9\sigma_{\max}^4}-\frac{1}{M}\left(4C_3+\frac{13}{12\sigma_{\min}^2\underline{c}^2}\right)\right).\end{aligned}$$

Hence we have $\widetilde{G}(\Delta) > 0$ for M large enough, in particular $M = (9\sigma_{\max}^4/(2\underline{k}^2))\left(4C_3+13/(12\sigma_{\min}^2\underline{c}^2)\right)$ suffices. ■

The rest of the proof follows that of Theorem 19, see Proposition 26 and the bounds which follow. We thus establish that the theorem holds. ■

Appendix F. Oracle Inequalities for the Lasso

In this section, we consider recovering $\beta\in\mathbb{R}^p$ in the following linear model:

$$Y = X\beta+\varepsilon,$$

where X follows (15) and $\varepsilon\sim N(0,\sigma^2 I_n)$. Recall given λ_n, the Lasso estimator for $\beta\in\mathbb{R}^p$ is defined as:

$$\widehat{\beta} = \arg\min_{\beta}\frac{1}{2n}\|Y-X\beta\|_2^2+\lambda_n\|\beta\|_1, \tag{78}$$

which corresponds to the regression function in (10) by letting $Y := X_i$ and $X := X_{\cdot\setminus i}$ where $X_{\cdot\setminus i}$ denotes columns of X without i. Define s_0 as the smallest integer such that

$$\sum_{i=1}^{p}\min(\beta_i^2,\lambda^2\sigma^2)\leq s_0\lambda^2\sigma^2,\ \text{where}\ \lambda=\sqrt{2\log p/n}. \tag{79}$$

For $X\in\mathcal{F}(\theta)$ as defined in (39), define

$$\mathcal{T}_a = \left\{\varepsilon:\left\|\frac{X^T\varepsilon}{n}\right\|_\infty\leq(1+\theta)\lambda_{\sigma,a,p},\ \text{where}\ X\in\mathcal{F}(\theta),\ \text{for}\ 0<\theta<1\right\}, \tag{80}$$

where $\lambda_{\sigma,a,p} = \sigma\sqrt{1+a}\sqrt{(2\log p)/n}$, where $a \geq 0$. We have (cf. Lemma 34)

$$\mathbb{P}(\mathcal{T}_a) \geq 1 - (\sqrt{\pi \log p} p^a)^{-1};$$

In fact, for such a bound to hold, we only need $\frac{\|X_j\|_2}{\sqrt{n}} \leq 1+\theta, \forall j$ to hold in $\mathcal{F}(\theta)$.

We now state Theorem 33, which may be of independent interests as the bounds on ℓ_2 and ℓ_1 loss for the Lasso estimator are stated with respect to the *actual* sparsity s_0 rather than $s = |\operatorname{supp}(\beta)|$ as in Bickel et al. [2009, Theorem 7.2]. The proof is omitted as on event $\mathcal{T}_a \cap \mathcal{X}$, it follows exactly that of Zhou [2010a, Theorem 5.1] for a deterministic design matrix X which satisfies the RE condition, with some suitable adjustments on the constants.

Theorem 33 ((Oracle inequalities of the Lasso) Zhou, 2010a) *Let $Y = X\beta + \varepsilon$, for ε being i.i.d. $N(0,\sigma^2)$ and let X follow* (15). *Let s_0 be as in* (79) *and T_0 denote locations of the s_0 largest coefficients of β in absolute values. Suppose that $RE(s_0,4,\Sigma_0)$ holds with $K(s_0,4,\Sigma_0)$ and $\rho_{\min}(s) > 0$. Fix some $1 > \theta > 0$. Let β_{init} be an optimal solution to* (78) *with*

$$\lambda_n = d_0 \lambda \sigma \geq 2(1+\theta)\lambda_{\sigma,a,p} \tag{81}$$

where $a \geq 1$ and $d_0 \geq 2(1+\theta)\sqrt{1+a}$. Let $h = \beta_{\text{init}} - \beta_{T_0}$. Define

$$\mathcal{X} := \mathcal{R}(\theta) \cap \mathcal{F}(\theta) \cap \mathcal{M}(\theta).$$

Suppose that n satisfies (42). *Then on $\mathcal{T}_a \cap \mathcal{X}$, we have*

$$\begin{aligned} \|\beta_{\text{init}} - \beta\|_2 &\leq \lambda_n \sqrt{s_0}\sqrt{2D_0^2 + 2D_1^2 + 2} := \lambda\sigma\sqrt{s_0} d_0 \sqrt{2D_0^2 + 2D_1^2 + 2}, \\ \left\|h_{T_0^c}\right\|_1 &\leq D_1 \lambda_n s_0 := D_1 d_0 \lambda \sigma s_0, \end{aligned}$$

where D_0 and D_1 are defined in (82) *and* (83) *respectively, and $\mathbb{P}(\mathcal{X} \cap \mathcal{T}_a) \geq 1 - 3\exp(-c\theta^2 n/\alpha^4) - (\sqrt{\pi \log p} p^a)^{-1}$.*

Let T_1 denote the s_0 largest positions of h in absolute values outside of T_0; Let $T_{01} := T_0 \cup T_1$. The proof of Theorem 33 yields the following bounds on $\mathcal{X} \cap \mathcal{T}_a$: $\|h_{T_{01}}\|_2 \leq D_0 d_0 \lambda \sigma \sqrt{s_0}$ where

$$\begin{aligned} D_0 &= \max\left\{ \frac{D}{d_0},\ 2\sqrt{2}(1+\theta)\frac{K(s_0,4,\Sigma_0)\sqrt{\rho_{\max}(s-s_0)}}{(1-\theta)d_0} + \frac{3\sqrt{2}K^2(s_0,4,\Sigma_0)}{(1-\theta)^2} \right\} \\ \text{where } D &= \frac{3(1+\theta)\sqrt{\rho_{\max}(s-s_0)}}{(1-\theta)\sqrt{\rho_{\min}(2s_0)}} + \frac{2(1+\theta)^4 \rho_{\max}(3s_0)\rho_{\max}(s-s_0)}{d_0(1-\theta)^2 \rho_{\min}(2s_0)}, \end{aligned} \tag{82}$$

and

$$D_1 = \max\left\{ \frac{4(1+\theta)^2\rho_{\max}(s-s_0)}{d_0^2}, \left(\frac{(1+\theta)\sqrt{\rho_{\max}(s-s_0)}}{d_0} + \frac{3K(s_0,4,\Sigma_0)}{2(1-\theta)} \right)^2 \right\}. \tag{83}$$

We note that implicit in these constants, we have used the concentration bounds for $\Lambda_{\max}(3s_0)$, $\Lambda_{\max}(s-s_0)$ and $\Lambda_{\min}(2s_0)$ as derived in Theorem 10, given that (41) holds for $m \leq \max(s, (k_0 +$

$1)s_0)$, where we take $k_0 > 3$. In general, these maximum sparse eigenvalues as defined above will increase with s_0 and s; Taking this issue into consideration, we fix for $c_0 \geq 4\sqrt{2}$, $\lambda_n = d_0\lambda\sigma$ where

$$d_0 = c_0(1+\theta)^2\sqrt{\rho_{\max}(s-s_0)\rho_{\max}(3s_0)} \geq 2(1+\theta)\sqrt{1+a},$$

where the second inequality holds for $a = 7$ as desired, given $\rho_{\max}(3s_0), \rho_{\max}(s-s_0) \geq 1$.

Thus we have for $\rho_{\max}(3s_0) \geq \rho_{\max}(2s_0) \geq \rho_{\min}(2s_0)$

$$\begin{aligned} D/d_0 &\leq \frac{3}{c_0(1+\theta)(1-\theta)\sqrt{\rho_{\max}(3s_0)}\sqrt{\rho_{\min}(2s_0)}} + \frac{2}{c_0^2(1-\theta)^2\rho_{\min}(2s_0)} \\ &\leq \frac{3\sqrt{\rho_{\min}(2s_0)}}{c_0(1-\theta)^2\sqrt{\rho_{\max}(3s_0)}\rho_{\min}(2s_0)} + \frac{2}{c_0^2(1-\theta)^2\rho_{\min}(2s_0)} \\ &\leq \frac{2(3c_0+2)K^2(s_0,4,\Sigma_0)}{c_0^2(1-\theta)^2} \leq \frac{7\sqrt{2}K^2(s_0,4,\Sigma_0)}{8(1-\theta)^2} \end{aligned}$$

which holds given that $\rho_{\max}(3s_0) \geq 1$, and $1 \leq \frac{1}{\sqrt{\rho_{\min}(2s_0)}} \leq \sqrt{2}K(s_0,k_0,\Sigma_0)$, and thus $\frac{1}{K^2(s_0,k_0,\Sigma_0)} \leq 2$ as shown in Lemma 35; Hence

$$\begin{aligned} D_0 &\leq \max\left\{D/d_0, \frac{(4+3\sqrt{2}c_0)\sqrt{\rho_{\max}(s-s_0)\rho_{\max}(3s_0)}(1+\theta)^2K^2(s_0,4,\Sigma_0)}{d_0(1-\theta)^2}\right\}, \\ &\leq \frac{7K^2(s_0,4,\Sigma_0)}{\sqrt{2}(1-\theta)^2} < \frac{5K^2(s_0,4,\Sigma_0)}{(1-\theta)^2} \text{ and} \\ D_1 &\leq \left(\frac{6}{4(1-\theta)} + \frac{1}{4}\right)^2 K^2(s_0,4,\Sigma_0) \leq \frac{49K^2(s_0,4,\Sigma_0)}{16(1-\theta)^2}, \end{aligned}$$

where for both D_1, we have used the fact that

$$\begin{aligned} \frac{2(1+\theta)^2\rho_{\max}(s-s_0)}{d_0^2} &= \frac{2}{c_0^2(1+\theta)^2\rho_{\max}(3s_0)} \leq \frac{2}{c_0^2(1+\theta)^2\rho_{\min}(2s_0)} \\ &\leq \frac{4K^2(s_0,4,\Sigma_0)}{c_0^2(1+\theta)^2} \leq \frac{K^2(s_0,4,\Sigma_0)}{8}. \end{aligned}$$

Appendix G. Misc Bounds

Lemma 34 *For fixed design X with $\max_j \|X_j\|_2 \leq (1+\theta)\sqrt{n}$, where $0 < \theta < 1$, we have for $\mathcal{T}_a$ as defined in* (80)*, where $a > 0$, $\mathbb{P}(\mathcal{T}_a^c) \leq (\sqrt{\pi \log p}\, p^a)^{-1}$.*

Proof Define random variables: $Y_j = \frac{1}{n}\sum_{i=1}^n \varepsilon_i X_{i,j}$. Note that $\max_{1\leq j\leq p}|Y_j| = \|X^T\varepsilon/n\|_\infty$. We have $\mathbb{E}(Y_j) = 0$ and $\mathsf{Var}((Y_j)) = \|X_j\|_2^2\sigma^2/n^2 \leq (1+\theta)\sigma^2/n$. Let $c_1 = 1+\theta$. Obviously, Y_j has its tail probability dominated by that of $Z \sim N(0, \frac{c_1^2\sigma^2}{n})$:

$$\mathbb{P}(|Y_j| \geq t) \leq \mathbb{P}(|Z| \geq t) \leq \frac{2c_1\sigma}{\sqrt{2\pi n t}} \exp\left(\frac{-nt^2}{2c_1^2\sigma_\varepsilon^2}\right).$$

We can now apply the union bound to obtain:

$$\begin{aligned}\mathbb{P}\left(\max_{1\leq j\leq p}|Y_j|\geq t\right) &\leq p\frac{c_1\sigma}{\sqrt{nt}}\exp\left(\frac{-nt^2}{2c_1^2\sigma^2}\right)\\ &= \exp\left(-\left(\frac{nt^2}{2c_1^2\sigma^2}+\log\frac{t\sqrt{\pi n}}{\sqrt{2}c_1\sigma}-\log p\right)\right).\end{aligned}$$

By choosing $t=c_1\sigma\sqrt{1+a}\sqrt{2\log p/n}$, the right-hand side is bounded by $(\sqrt{\pi\log p}p^a)^{-1}$ for $a\geq 0$. ■

Lemma 35 (Zhou, 2010b) *Suppose that $RE(s_0,k_0,\Sigma_0)$ holds for $k_0>0$, then for $m=(k_0+1)s_0$,*

$$\begin{aligned}\sqrt{\rho_{\min}(m)} &\geq \frac{1}{\sqrt{2+k_0^2}K(s_0,k_0,\Sigma_0)};\ \textit{and clearly}\\ \textit{if } \Sigma_{0,ii}=1,\forall i,\ \textit{then } 1\geq\sqrt{\rho_{\min}(2s_0)} &\geq \frac{1}{\sqrt{2}K(s_0,k_0,\Sigma_0)}\ \textit{ for }\ k_0\geq 1.\end{aligned}$$

References

O. Banerjee, L. El Ghaoui, and A. d'Aspremont. Model selection through sparse maximum likelihood estimation for multivariate Gaussian or binary data. *Journal of Machine Learning Research*, 9:485–516, 2008.

P. Bickel and E. Levina. Some theory for Fisher's linear discriminant function, "naive Bayes", and some alternatives when there are many morevariables than observations. *Bernoulli*, 10:989–1010, 2004.

P. Bickel and E. Levina. Regulatized estimation of large covariance matrices. *The Annals of Statistics*, 36:199–227, 2008.

P. Bickel, Y. Ritov, and A. Tsybakov. Simultaneous analysis of Lasso and Dantzig selector. *The Annals of Statistics*, 37:1705–1732, 2009.

P. Bühlmann and L. Meier. Discussion: One-step sparse estimates in nonconcave penalized likelihood models. *The Annals of Statistics*, 36:1534–1541, 2008.

T. Cai, W. Liu, and X. Luo. A constrained ℓ_1 minimization approach to sparse precision matrix estimation. *Journal of the American Statistical Association*, 106:594–607, 2011.

E. Candès and T. Tao. The Dantzig selector: statistical estimation when p is much larger than n. *Annals of Statistics*, 35(6):2313–2351, 2007.

S. Chaudhuri, M. Drton, and T. Richardson. Estimation of a covariance matrix with zeros. *Biometrika*, 94:1–18, 2007.

A. d'Aspremont, O. Banerjee, and L. El Ghaoui. First-order methods for sparse covariance selection. *SIAM Journal on Matrix Analysis and Applications*, 30:56–66, 2008.

J. Fan, Y. Feng, and Y. Wu. Network exploration via the adaptive lasso and scad penalties. *The Annals of Applied Statistics*, 3:521–541, 2009.

J. Friedman, T. Hastie, and R. Tibshirani. Sparse inverse covariance estimation with the graphical Lasso. *Biostatistics*, 9:432–441, 2007.

J. Friedman, T. Hastie, and R. Tibshirani. Regularization paths for generalized linear models via coordinate descent. *Journal of Statistical Software*, 33, 2010.

R. Furrer and T. Bengtsson. Estimation of high-dimensional prior and posterior covariance matrices in Kalman filter variants. *Journal of Multivariate Analysis*, 98:227–255, 2007.

J. Huang, N. Liu, M. Pourahmad, and L. Liu. Covariance matrix selection and estimation via penalised normal likelihood. *Biometrika*, 93:85–98, 2006.

J. Huang, S. Ma, and C. Zhang. Adaptive Lasso for sparse highdimensional regression. *Statistica Sinica*, 18:1603–1618, 2008.

I. Johnstone. Chi-square oracle inequalities. *In State of the Art in Probability and Statistics, Festchrift for Willem R. van Zwet, M. de Gunst and C. Klaassen and A. van der Waart editors, IMS Lecture Notes - Monographs*, 36:399–418, 2001.

C. Lam and J. Fan. Sparsistency and rates of convergence in large covariance matrices estimation. *The Annals of Statistics*, 37:4254–4278, 2009.

E. Levina, A. Rothman, and J. Zhu. Sparse estimation of large covariance matrices via a nested Lasso penalty. *The Annals of Applied Statistics*, 2:245–263, 2008.

N. Meinshausen. Relaxed Lasso. *Computational Statistics and Data Analysis*, 52:374–393, 2007.

N. Meinshausen. A note on the Lasso for gaussian graphical model selection. *Statistics and Probability Letters*, 78:880–884, 2008.

N. Meinshausen and P. Bühlmann. High dimensional graphs and variable selection with the Lasso. *The Annals of Statistics*, 34:1436–1462, 2006.

N. Meinshausen and B. Yu. Lasso-type recovery of sparse representations for high-dimensional data. *Annals of Statistics*, 37(1):246–270, 2009.

J. Peng, P. Wang, N. Zhou, and J. Zhu. Partial correlation estimation by joint sparse regression models. *Journal of the American Statistical Association*, 104:735–746, 2009.

P. Ravikumar, M. Wainwright, G. Raskutti, and B. Yu. High-dimensional covariance estimation by minimizing ℓ_1-penalized log-determinant divergence. *Electronic Journal of Statistics*, 4:935–980, 2011.

A. Rothman, P. Bickel, E. Levina, and J. Zhu. Sparse permutation invariant covariance estimation. *Electronic Journal of Statistics*, 2:494–515, 2008.

M. Rudelson and S. Zhou. Reconstruction from anisotropic random measurements, 2011. University of Michigan, Department of Statistics, Technical Report 522. Available at arXiv:1106.1151v1.

P. Rütimann and P. Bühlmann. High dimensional sparse covariance estimation via directed acyclic graphs. *Electronic Journal of Statistics*, 3:1133–1160, 2009.

C. Uhler. Geometry of maximum likelihood estimation in gaussian graphical models. Available at arXiv:1012.2643v1, 2011.

S. van de Geer, P. Bühlmann, and S. Zhou. The adaptive and the thresholded Lasso for potentially misspecified models (and a lower bound for the lasso). *Electronic Journal of Statistics*, 5:688–749, 2011.

N. Verzelen. Adaptive estimation of covariance matrices via cholesky decomposition. *Electronic Journal of Statistics*, 4:1113–1150, 2010.

M. West, C. Blanchette, H. Dressman, E. Huang, S. Ishida, R. Spang, H. Zuzan, J.A. Olson Jr., J.R. Marks, and J.R. Nevins. Predicting the clinical status of human breast cancer by using gene expression profiles. *PNAS*, 98:11462–11467, 2001.

A. Wille, P. Zimmermann, E. Vranova, A. Fürholz, O. Laule, S. Bleuler, L. Hennig, A. Prelic, P. von Rohr, L. Thiele, E. Zitzler, W. Gruissem, and P. Bühlmann. Sparse graphical Gaussian modeling of the isoprenoid gene network in arabidopsis thaliana. *Genome Biology*, 5:R92, 2004.

W. Wu and M. Pourahmadi. Nonparametric estimation of large covariance matrices of longitudinal data. *Biometrika*, 90:831–844, 2003.

M. Yuan. High dimensional inverse covariance matrix estimation via linear programming. *Journal of Machine Learning Research*, 11:2261–2286, 2010.

M. Yuan and Y. Lin. Model selection and estimation in the gaussian graphical model. *Biometrika*, 94:19–35, 2007.

P. Zhao and B. Yu. On model selection consistency of Lasso. *Journal of Machine Learning Research*, 7:2541–2563, 2006.

S. Zhou. Thresholding procedures for high dimensional variable selection and statistical estimation. In *Advances in Neural Information Processing Systems 22*. MIT Press, 2009.

S. Zhou. Thresholded Lasso for high dimensional variable selection and statistical estimation. University of Michigan, Department of Statistics Technical Report 511. Available at arXiv:1002.1583v2, 2010a.

S. Zhou. Restricted eigenvalue conditions on subgaussian random matrices, 2010b. Manuscript, earlier version available at arXiv:0904.4723v2.

S. Zhou, J. Lafferty, and L. Wasserman. Time varying undirected graphs. In *Proceedings of the 21st Annual Conference on Computational Learning Theory (COLT'08)*, July 2008.

S. Zhou, S. van de Geer, and P. Bühlmann. Adaptive Lasso for high dimensional regression and gaussian graphical modeling, 2009. Available at arXiv:0903.2515.

H. Zou. The adaptive Lasso and its oracle properties. *Journal of the American Statistical Association*, 101:1418–1429, 2006.

H. Zou and R. Li. One-step sparse estimates in nonconcave penalized likelihood models. *The Annals of Statistics*, 36:1509–1533, 2008.

Journal of Machine Learning Research 12 (2011) 3027-3063 Submitted 4/10; Revised 1/11; Published 10/11

Robust Approximate Bilinear Programming for Value Function Approximation

Marek Petrik MPETRIK@US.IBM.COM
IBM T.J. Watson Research Center
P.O. Box 218
Yorktown Heights, NY 10598, USA

Shlomo Zilberstein SHLOMO@CS.UMASS.EDU
Department of Computer Science
University of Massachusetts
Amherst, MA 01003, USA

Editor: Shie Mannor

Abstract

Value function approximation methods have been successfully used in many applications, but the prevailing techniques often lack useful *a priori* error bounds. We propose a new *approximate bilinear programming* formulation of value function approximation, which employs global optimization. The formulation provides strong a priori guarantees on both robust and expected policy loss by minimizing specific norms of the Bellman residual. Solving a bilinear program optimally is NP-hard, but this worst-case complexity is unavoidable because the Bellman-residual minimization itself is NP-hard. We describe and analyze the formulation as well as a simple approximate algorithm for solving bilinear programs. The analysis shows that this algorithm offers a *convergent* generalization of approximate policy iteration. We also briefly analyze the behavior of bilinear programming algorithms under incomplete samples. Finally, we demonstrate that the proposed approach can consistently minimize the Bellman residual on simple benchmark problems.

Keywords: value function approximation, approximate dynamic programming, Markov decision processes

1. Introduction

Solving large Markov Decision Processes (MDPs) is a very useful, but computationally challenging problem addressed widely in the AI literature, particularly in the area of reinforcement learning. It is widely accepted that large MDPs can be solved only approximately. The commonly used approximation methods can be divided into three broad categories: 1) *policy search*, which explores a restricted space of all policies, 2) *approximate dynamic programming*—or value function approximation—which searches a restricted space of value functions, and 3) *approximate linear programming*, which approximates the solution using a linear program. The goal of approximate methods is to compute a policy that minimizes the *policy loss*—the difference between the returns of the computed policy and an optimal one. While all of these approximate methods have achieved impressive results in various application domains, they have significant limitations.

Policy search methods rely on local search in a restricted policy space. The policy may be represented, for example, as a finite-state controller (Stanley and Miikkulainen, 2004) or as a greedy policy with respect to an approximate value function (Szita and Lorincz, 2006). Policy search

methods have achieved impressive results in such domains as Tetris (Szita and Lorincz, 2006) and helicopter control (Abbeel et al., 2006). However, they are notoriously hard to analyze. We are not aware of any established theoretical guarantees regarding the quality of the solution.

Approximate dynamic programming (ADP) methods iteratively approximate the value function (Bertsekas and Ioffe, 1997; Powell, 2007; Sutton and Barto, 1998). They have been extensively analyzed and are the most commonly used methods. However, approximate dynamic programming methods typically do not converge and they only provide weak guarantees of approximation quality. The approximation error bounds are usually expressed in terms of the worst-case approximation of the value function over all policies (Bertsekas and Ioffe, 1997). In addition, most available bounds are with respect to the L_∞ norm, while the algorithms often minimize the L_2 norm. While there exist some L_2-based bounds (Munos, 2003), they require values that are difficult to obtain.

Approximate linear programming (ALP) uses a linear program to compute the approximate value function in a particular vector space (de Farias, 2002). ALP has been previously used in a wide variety of settings (Adelman, 2004; de Farias and van Roy, 2004; Guestrin et al., 2003). Although ALP often does not perform as well as ADP, there have been some recent efforts to close the gap (Petrik and Zilberstein, 2009). ALP has better theoretical properties than ADP and policy search. It is guaranteed to converge and return the closest L_1-norm approximation $\tilde{v}$ of the optimal value function $v^\star$ up to a multiplicative factor. However, the L_1 norm must be properly weighted to guarantee a small policy loss, and there is no *reliable* method for selecting appropriate weights (de Farias, 2002).

To summarize, existing reinforcement learning techniques often provide good solutions, but typically require significant domain knowledge (Powell, 2007). The domain knowledge is needed partly because useful a priori error bounds are not available, as mentioned above. Our goal is to develop a more *reliable* method that is guaranteed to minimize bounds on the policy loss in various settings.

This paper presents new formulations for value function approximation that provably minimize bounds on policy loss using a global optimization framework; we consider both L_∞ and weighted L_1 error bounds. To minimize the policy loss, we derive new bounds based on approximate value functions. These bounds do not require coefficients that are hard to obtain or compute, unlike, for example, bounds for approximate linear programming.

An advantage of the approach we propose is that the actual solutions and their properties are independent of the methods used to compute them. The paper focuses on the development of models for value function approximation and their properties. Although we do present two methods for solving these models, it is likely that more efficient algorithms will be developed in the future.

We start with a description of the framework and notation in Section 2 and a description of value function approximation in Section 3. Then, in Section 4, we describe the proposed approximate bilinear programming (ABP) formulations. Bilinear programs are typically solved using global optimization methods, which we briefly discuss in Section 5. A drawback of the bilinear formulation is that solving bilinear programs may require exponential time. We show in Section 5, however, that this complexity is unavoidable because minimizing the approximation error bound is in fact NP-hard.

In practice, only sampled versions of ABPs are often solved. While a thorough treatment of sampling is beyond the scope of this paper, we examine the impact of sampling and establish some guarantees in Section 6. Unlike classical sampling bounds on approximate linear programming, we describe bounds that apply to the worst-case error. Section 7 shows that ABP is related to other

approximate dynamic programming methods, such as approximate linear programming and policy iteration. Section 8 demonstrates the applicability of ABP using common reinforcement learning benchmark problems.

The general setting considered in this paper is a restricted form of reinforcement learning. In reinforcement learning, methods can use samples without requiring a model of the environment. The methods we propose can also be based on samples, but they require additional structure. In particular, they require that all or most actions are sampled for every state. Such samples can be easily generated when a model of the environment is available.

2. Framework and Notation

This section defines the framework and the notation we use. We also define Markov decision processes and the associated approximation errors. Markov decision processes come in many forms, depending on the objective function that is optimized. This work focuses on infinite-horizon discounted MDPs, which are defined as follows; a more extensive treatment is available, for example, in Puterman (2005).

Definition 1 *A* Markov Decision Process *is a tuple* $(\mathcal{S}, \mathcal{A}, P, r, \alpha)$*, where* $\mathcal{S}$ *is a finite set of states,* $\mathcal{A}$ *is a finite set of actions,* $P : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \mapsto [0,1]$ *is the transition function (*$P(s,a,s')$ *is the probability of transiting to state* s' *from state* s *given action* a*), and* $r : \mathcal{S} \times \mathcal{A} \mapsto \mathbb{R}$ *is a reward function. The initial distribution is:* $\alpha : \mathcal{S} \mapsto [0,1]$*, such that* $\sum_{s \in \mathcal{S}} \alpha(s) = 1$.

The goal in solving an MDP is to find a sequence of actions that maximizes the expected γ-discounted cumulative sum of rewards, also called the *return*. A solution of a Markov decision process is a policy, defined as follows.

Definition 2 *A deterministic stationary* policy $\pi : \mathcal{S} \mapsto \mathcal{A}$ *assigns an action to each state of the Markov decision process. A stochastic stationary* policy $\pi : \mathcal{S} \times \mathcal{A} \mapsto [0,1]$ *satisfies* $\sum_{a \in \mathcal{A}} \pi(s,a) = 1$ *for each* $s \in \mathcal{S}$*. The set of all stochastic stationary policies is denoted as* Π.

Non-stationary policies may take different actions in the same state in different time-steps. We limit our treatment to stationary policies, since for infinite-horizon MDPs there exists an optimal *stationary* and *deterministic* policy. We also consider stochastic policies because they are more convenient to use in some settings. A policy $\pi \in \Pi$ together with the transition matrix induces a distribution over the state space $\mathcal{S}$ in every time step resulting in random variables S_t for $t = 0 \ldots \infty$. The return of a policy is then defined as:

$$\rho(\pi, \alpha) = \mathbf{E}_\alpha \left[\sum_{t=0}^{\infty} \sum_{a \in \mathcal{A}} \gamma^t \pi(S_t, a) r(S_t, a) \right] ,$$

where α is the distribution of S_0. Our objective is then $\max_{\pi \in \Pi} \rho(\pi, \alpha)$, for which the optimal solution is some deterministic policy $\pi^\star$.

The transition matrix and reward function for a *deterministic* policy π are defined as:

$$P_\pi : (s, s') \mapsto P(s, \pi(s), s') \quad \text{and} \quad r_\pi : s \mapsto r(s, \pi(s)) .$$

The transition matrix and reward function for a *stochastic* policy π are defined as:

$$P_\pi : (s, s') \mapsto \sum_{a \in \mathcal{A}} \pi(s,a) P(s,a,s') \quad \text{and} \quad r_\pi : s \mapsto \sum_{a \in \mathcal{A}} \pi(s,a) r(s,a) .$$

In addition, we use P_a and r_a to denote these values for a constant policy $\pi(s) = a$ for some $a \in \mathcal{A}$.

The value function $v : \mathcal{S} \to \mathbb{R}$ represents the expected return when starting in a particular state. The set of all value functions is denoted as $\mathcal{V} = \mathbb{R}^{|\mathcal{S}|}$. A value function v_π of a policy π is: $v_\pi = (\mathbf{I} - \gamma P_\pi)^{-1} r_\pi$.

The value function update for a policy π is denoted by L_π, and the Bellman operator is denoted by L and defined as:

$$L_\pi v = \gamma P_\pi v + r_\pi \ , \qquad\qquad Lv = \max_{\pi \in \Pi} L_\pi v \ .$$

The value function update for a stochastic policy π can be written as:

$$(L_\pi v)(s) = \sum_{a \in \mathcal{A}, s' \in \mathcal{S}} \pi(s,a) \left(\gamma P(s,a,s') v(s') + r(s,a) \right) \ .$$

A policy π is *greedy* with respect to a value function v when $L_\pi v = Lv$. The optimal value function $v^\star = v_{\pi^\star}$ satisfies $v^\star = Lv^\star$. The following proposition summarizes an important property of optimal value functions.

Proposition 3 (Section 6.9 in Puterman, 2005) *For any policy $\pi \in \Pi$ the optimal value function is an upper bound on the value function of any policy:*

$$v^\star \geq v_\pi \ .$$

We assume a vector representation of the policy $\pi \in \mathbb{R}^{|\mathcal{S}||\mathcal{A}|}$. The variables π are defined for all state-action pairs and represent policies. That is, $\pi(s,a)$ represents the probability of taking action $a \in \mathcal{A}$ in state $s \in \mathcal{S}$. The space of all stochastic policies can be represented using the following set of linear equations:

$$\sum_{a \in \mathcal{A}} \pi(s,a) = 1 \qquad\qquad \forall s \in \mathcal{S} \ ,$$
$$\pi(s,a) \geq 0 \qquad\qquad \forall s \in \mathcal{S}, \forall a \in \mathcal{A} \ .$$

These inequalities can be represented using matrix notation as follows:

$$B\pi = \mathbf{1} \qquad\qquad \pi \geq \mathbf{0} \ ,$$

where the matrix $B : |\mathcal{S}| \times (|\mathcal{S}| \cdot |\mathcal{A}|)$ is defined as follows:

$$B(s', (s,a)) = \begin{cases} 1 & s = s' \\ 0 & \text{otherwise} \end{cases} \ .$$

We use $\mathbf{0}$ and $\mathbf{1}$ to denote vectors of all zeros or ones of the appropriate size respectively. The symbol $\mathbf{I}$ denotes an identity matrix of the appropriate dimension.

In addition, a policy π induces a *state occupancy frequency* $u_\pi : \mathcal{S} \to \mathbb{R}$, defined as follows:

$$u_\pi = \left(\mathbf{I} - \gamma P_\pi^\mathsf{T} \right)^{-1} \alpha \ .$$

The set of all occupancy frequencies is denoted as $\mathcal{U} \subseteq \mathbb{R}^{|\mathcal{S}|}$. The return of a policy depends on the state-action occupancy frequencies and $\alpha^\mathsf{T} v_\pi = r_\pi^\mathsf{T} u_\pi$. The optimal state-action occupancy frequency

is $u_{\pi^\star}$ and is often denoted as $u^\star$. *State-action occupancy frequency* $u : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ is defined for all states and actions; notice the missing subscript. We use $u|_a : \mathcal{S} \to \mathbb{R}$ to denote the restriction of u to action $a \in \mathcal{A}$ and use $u|_\pi$ equivalently for a deterministic policy π as $u|_\pi : s \mapsto u(s, \pi(s,a))$. State-action occupancy frequencies u must satisfy (e.g., Section 6.9 in Puterman, 2005):

$$\sum_{a \in \mathcal{A}} (\mathbf{I} - \gamma P_a)^\mathsf{T} u|_a = \alpha \qquad \forall a \in \mathcal{A} \; .$$

To formulate approximate linear and bilinear programs, it is necessary to restrict the value functions so that their Bellman residuals are non-negative (or at least bounded from below). We call such value functions transitive-feasible and define them as follows.

Definition 4 *A value function is* transitive-feasible *when* $v \geq Lv$. *The set of transitive-feasible value functions is:*

$$\mathcal{K} = \{v \in \mathcal{V} \,|\, v \geq Lv\} \; .$$

Given some $\varepsilon \geq 0$, *the set of* ε-transitive-feasible *value functions is:*

$$\mathcal{K}(\varepsilon) = \{v \in \mathcal{V} \,|\, v \geq Lv - \varepsilon \mathbf{1}\} \; .$$

Notice that the optimal value function $v^\star$ is transitive-feasible.

Next, we summarize the key properties of value functions and policies that we use to derive the results. First, the following lemma summarizes the monotonicity of transition matrices; it follows from the geometric sequence representation of the matrix inverse.

Lemma 5 *[Monotonicity] Let* P *be a stochastic matrix. Then both linear operators* P *and* $(\mathbf{I} - \gamma P)^{-1}$ *are monotonous:*

$$x \geq y \Rightarrow Px \geq Py \; ,$$
$$x \geq y \Rightarrow (\mathbf{I} - \gamma P)^{-1} x \geq (\mathbf{I} - \gamma P)^{-1} y$$

for all x *and* y.

An important property, which we rely on, is that greedy policies are not affected by adding or subtracting a constant from a value function; we state this well-known property without proof.

Proposition 6 *Let* $v \in \mathcal{V}$ *be any value function and assume an arbitrary* $c \in \mathbb{R}$. *Then:*

$$L(v + c\mathbf{1}) = Lv + \gamma c \mathbf{1} \; .$$

In addition, if π *is a greedy policy with respect to* v *it is also greedy with respect to* $v + c\mathbf{1}$.

The models we define also rely on the following basic properties of the Bellman operator.

Lemma 7 *Let* u *be the state-action occupancy frequency of some policy* π. *Then:*

$$\mathbf{1}^\mathsf{T} u = 1/(1-\gamma) \; .$$

Proof The lemma follows because:

$$\sum_{a\in\mathcal{A}}(u|_a)^\mathsf{T}(\mathbf{I}-\gamma P_a) = \alpha^\mathsf{T} ,$$
$$\sum_{a\in\mathcal{A}}(u|_a)^\mathsf{T}(\mathbf{I}-\gamma P_a)\mathbf{1} = \alpha^\mathsf{T}\mathbf{1} ,$$
$$(1-\gamma)\sum_{a\in\mathcal{A}}(u|_a)^\mathsf{T}\mathbf{1} = 1 = (1-\gamma)u^\mathsf{T}\mathbf{1} .$$

■

Finally, an important property of transitive-feasible value functions is that they represent an upper bound on the optimal value function.

Lemma 8 *Transitive feasible value functions form an upper bound on the optimal value function. If $v \in \mathcal{K}(\varepsilon)$ is an ε-transitive-feasible value function, then:*

$$v \geq v^\star - \varepsilon/(1-\gamma)\mathbf{1} .$$

Proof Let $P^\star$ and $r^\star$ be the transition matrix and the reward vector of the optimal policy. Then, using Theorem 5, we get:

$$v \geq Lv - \varepsilon\mathbf{1} ,$$
$$v \geq \gamma P^\star v + r^\star - \varepsilon\mathbf{1} ,$$
$$(\mathbf{I}-\gamma P^\star)v \geq r^\star - \varepsilon\mathbf{1} ,$$
$$v \geq (\mathbf{I}-\gamma P^\star)^{-1} r^\star - \varepsilon/(1-\gamma) .$$

■

3. Value Function Approximation

This section describes basic methods for value function approximation used to solve large MDPs. Value function approximation, as its name indicates, only computes an approximate value function $\tilde{v}$ of the MDP. The actual solution of the MDP is then the greedy policy π with respect to this value function $\tilde{v}$. The quality of such a policy can be characterized using its value function v_π in one of the following two ways.

Definition 9 (Policy Loss) *Let π be a policy. The* expected policy loss σ_e *of π is defined as:*

$$\sigma_e(\pi) = \rho(\pi^\star,\alpha) - \rho(\pi,\alpha) = \|v^\star - v_\pi\|_{1,\alpha} = \alpha^\mathsf{T}v^\star - \alpha^\mathsf{T}v_\pi ,$$

where $\|x\|_{1,c}$ denotes the weighted L_1 norm: $\|x\|_{1,c} = \sum_i |c(i)x(i)|$.
The robust policy loss σ_r *of π is defined as:*

$$\sigma_r(\pi) = \max_{\{\alpha\geq\mathbf{0}\,|\,\mathbf{1}^\mathsf{T}\alpha=1\}} \rho(\pi^\star,\alpha) - \rho(\pi,\alpha) = \|v^\star - v_\pi\|_\infty = \max_{s\in\mathcal{S}} |v^\star(s) - v_\pi(s)| .$$

The expected policy loss captures the total loss of discounted reward when following the policy π instead of the optimal policy, given the initial state distribution. The robust policy loss ignores the initial distribution and, in some sense, measures the difference for the worst-case initial distribution.

A set of state features is a necessary component of value function approximation. These features must be supplied in advance and must capture the essential structure of the problem. The features are defined by mapping each state s to a vector $\phi(s)$ of features. We denote $\phi_i : \mathcal{S} \to \mathbb{R}$ to be a function that maps states to the value of feature i:

$$\phi_i(s) = (\phi(s))_i \ .$$

The desirable properties of the features depend strongly on the algorithm, samples, and attributes of the problem; the tradeoffs are not yet fully understood. The function ϕ_i can be treated as a vector, similarly to the value function v.

Value function approximation methods compute value functions that can be represented using the state features. We call such value functions *representable* and define them below.

Definition 10 *Given a* convex *polyhedral set $\tilde{\mathcal{V}} \subseteq \mathcal{V}$, a value function v is* representable *(in $\tilde{\mathcal{V}}$) if $v \in \tilde{\mathcal{V}}$.*

Many methods that compute a value function based on a given set of features have been developed, such as genetic algorithms and neural networks (Bertsekas and Tsitsiklis, 1996). Most of these methods are extremely hard to analyze, computationally complex, and hard to use. Moreover, these complex methods do not satisfy the convexity assumption in Theorem 10. A simpler and more common method is *linear value function approximation*, in which the value function of each state s is represented as a linear combination of *nonlinear features* $\phi(s)$. Linear value function approximation is easy to apply and analyze.

Linear value function approximation can be expressed in terms of matrices as follows. Let the matrix $\Phi : |\mathcal{S}| \times m$ represent the features for the state-space, where m is the number of features. The rows of the feature matrix Φ, also known as the *basis*, correspond to the features of the states $\phi(s)$. The feature matrix can be defined in one of the following two equivalent ways:

$$\Phi = \begin{pmatrix} - & \phi(s_1)^\mathsf{T} & - \\ - & \phi(s_2)^\mathsf{T} & - \\ & \vdots & \end{pmatrix} , \qquad \Phi = \begin{pmatrix} | & | & \\ \phi_1 & \phi_2 & \dots \\ | & | & \end{pmatrix} .$$

The value function v is then represented as $v = \Phi x$ and the set of representable functions is $\tilde{\mathcal{V}} = \text{colspan}\,(\Phi)$.

The goal of value function approximation is not simply to obtain a good value function $\tilde{v}$ but a policy with a small policy loss. Unfortunately, the policy loss of a greedy policy, as formulated in Theorem 9, depends non-trivially on the approximate value function $\tilde{v}$. Often, the only reliable method of precisely computing the policy loss is to simulate the policy, which can be very costly. The following theorem states the most common bound on the robust policy loss.

Theorem 11 *[Robust Policy Loss, Williams and Baird, 1994] Let π be a greedy policy with respect to a value function $\tilde{v}$. Then:*

$$\|v^\star - v_\pi\|_\infty \leq \frac{2}{1-\gamma} \|\tilde{v} - L\tilde{v}\|_\infty \ .$$

In addition, if $\tilde{v} \in \mathcal{K}$ then:

$$\|v^\star - v_\pi\|_\infty \leq \frac{1}{1-\gamma}\|\tilde{v} - L\tilde{v}\|_\infty \ .$$

The bounds in Theorem 11 are often overly conservative because they ignore the initial distribution and do not apply to the expected policy loss. We propose methods that minimize both the standard bounds in Theorem 11 and new tighter bounds on the expected policy loss in Theorem 12.

We are now ready to derive a new bound on the expected policy loss in its most general form. We show later how this bound relates to existing bounds and discuss its properties and special cases.

Theorem 12 *[Expected Policy Loss] Let $\pi \in \Pi$ be a greedy policy with respect to a value function $\tilde{v} \in \mathcal{V}$ and let the state occupancy frequencies of π be bounded as $\underline{u} \leq u_\pi \leq u$. Then:*

$$\begin{aligned}\sigma_e(\pi) = \|v^\star - v_\pi\|_{1,\alpha} &= \alpha^\mathsf{T} v^\star - \alpha^\mathsf{T}\tilde{v} + u_\pi^\mathsf{T}(\tilde{v} - L\tilde{v}) \\ &\leq \alpha^\mathsf{T} v^\star - \alpha^\mathsf{T}\tilde{v} + \underline{u}^\mathsf{T}[\tilde{v} - L\tilde{v}]_- + u^\mathsf{T}[\tilde{v} - L\tilde{v}]_+ \ ,\end{aligned}$$

where $[x]_+ = \max\{x, \mathbf{0}\}$ and $[x]_- = \min\{x, \mathbf{0}\}$ element-wise. In addition, when $\tilde{v} \in \mathcal{K}$, the bound is:

$$\|v^\star - v_\pi\|_{1,\alpha} \leq -\|v^\star - \tilde{v}\|_{1,\alpha} + \|\tilde{v} - L\tilde{v}\|_{1,u} \ , \tag{1}$$

$$\|v^\star - v_\pi\|_{1,\alpha} \leq -\|v^\star - \tilde{v}\|_{1,\alpha} + \frac{1}{1-\gamma}\|\tilde{v} - L\tilde{v}\|_\infty \ . \tag{2}$$

Proof Note that:

$$u_\pi^\mathsf{T}(\mathbf{I} - \gamma P_\pi) - \alpha^\mathsf{T} = \mathbf{0}^\mathsf{T} \ , \tag{3}$$

which follows directly from the definition of state-action occupancy frequencies. The bound is then derived as follows:

$$\begin{aligned}\|v^\star - v_\pi\|_\alpha \overset{\text{Theorem 3}}{=} \alpha^\mathsf{T} v^\star - \alpha^\mathsf{T} v_\pi &\overset{(3)}{=} \alpha^\mathsf{T} v^\star - \alpha^\mathsf{T} v_\pi + (u_\pi^\mathsf{T}(\mathbf{I} - \gamma P_\pi) - \alpha^\mathsf{T})\tilde{v} \\ &= \alpha^\mathsf{T} v^\star - r_\pi^\mathsf{T} u_\pi + (u_\pi^\mathsf{T}(\mathbf{I} - \gamma P_\pi) - \alpha^\mathsf{T})\tilde{v} \\ &= \alpha^\mathsf{T} v^\star - r_\pi^\mathsf{T} u_\pi + u_\pi^\mathsf{T}(\mathbf{I} - \gamma P_\pi)\tilde{v} - \alpha^\mathsf{T}\tilde{v} \\ &= \alpha^\mathsf{T} v^\star - \alpha^\mathsf{T}\tilde{v} + u_\pi^\mathsf{T}((\mathbf{I} - \gamma P_\pi)\tilde{v} - r_\pi) \\ &= \alpha^\mathsf{T} v^\star - \alpha^\mathsf{T}\tilde{v} + u_\pi^\mathsf{T}(\tilde{v} - L\tilde{v}) \\ &\leq \alpha^\mathsf{T} v^\star - \alpha^\mathsf{T}\tilde{v} + \underline{u}^\mathsf{T}[\tilde{v} - L\tilde{v}]_- + u^\mathsf{T}[\tilde{v} - L\tilde{v}]_+ \ .\end{aligned}$$

Inequality (1) then follows from Theorem 8, which implies that $\tilde{v} \geq v^\star$ and $v \geq Lv$. Inequality (2) follows using the trivial version of Holder's inequality as:

$$\begin{aligned}\alpha^\mathsf{T} v^\star - \alpha^\mathsf{T}\tilde{v} &\overset{\text{Theorem 8}}{=} -\|v^\star - \tilde{v}\|_{1,\alpha} \ , \\ u_\pi^\mathsf{T}(\tilde{v} - L\tilde{v}) &\overset{\text{Holder's}}{\leq} \|u_\pi\|_1 \|\tilde{v} - L\tilde{v}\|_\infty \overset{\text{Theorem 7}}{=} \frac{1}{1-\gamma}\|\tilde{v} - L\tilde{v}\|_\infty \ .\end{aligned}$$

■

Notice that the bounds in Theorem 12 can be minimized even without knowing the optimal $v^\star$. The optimal value function $v^\star$ is independent of the approximate value function $\tilde{v}$ and the greedy policy π depends only on $\tilde{v}$.

Algorithm 1: Approximate policy iteration, where $\mathcal{Z}(\pi)$ denotes a custom value function approximation for the policy π.

$\pi_0, k \leftarrow$ random, 1 ;
while $\pi_k \neq \pi_{k-1}$ **do**
  $\tilde{v}_k \leftarrow \mathcal{Z}(\pi_{k-1})$;
  $\pi_k(s) \leftarrow \arg\max_{a \in \mathcal{A}} r(s,a) + \gamma \sum_{s' \in \mathcal{S}} P(s,a,s') \tilde{v}_k(s) \quad \forall s \in \mathcal{S}$;
  $k \leftarrow k+1$;

Remark 13 *Theorem 12 generalizes the bounds established by de Farias (2002, Theorem 3.1), which state that for each $\tilde{v} \in \mathcal{K}$ and a greedy policy π:*

$$\|v^\star - v_\pi\|_{1,\alpha} \leq \frac{1}{1-\gamma} \|v^\star - \tilde{v}\|_{1,(1-\gamma)u_\pi} .$$

This bound is a special case of Inequality (1) *because $\alpha^\mathsf{T} v^\star - \alpha^\mathsf{T} \tilde{v} \leq 0$ and:*

$$\|\tilde{v} - L\tilde{v}\|_{1,u_\pi} \leq \|v^\star - \tilde{v}\|_{1,u_\pi} = \frac{1}{1-\gamma} \|v^\star - \tilde{v}\|_{1,(1-\gamma)u_\pi} ,$$

from $v^\star \leq L\tilde{v} \leq \tilde{v}$.

The methods that we propose require the following standard assumption.

Assumption 1 *All multiples of the constant vector $\mathbf{1}$ are representable in $\tilde{\mathcal{V}}$. That is, $k\mathbf{1} \in \tilde{\mathcal{V}}$ for all $k \in \mathbb{R}$.*

Notice that the representation set $\tilde{\mathcal{V}}$ satisfies Assumption 1 when the first column of Φ is $\mathbf{1}$. The impact of including the constant feature is typically negligible because adding a constant to the value function does not change the greedy policy.

Value function approximation algorithms are typically variations of the exact algorithms for solving MDPs. Hence, they can be categorized as approximate value iteration, approximate policy iteration, and approximate linear programming. The ideas behind approximate value iteration can be traced to Bellman (1957), which was followed by many additional research efforts (Bertsekas and Tsitsiklis, 1996; Sutton and Barto, 1998; Powell, 2007). Below, we only discuss approximate policy iteration and approximate linear programming, because they are the methods most closely related to our approach.

Approximate policy iteration (API) is summarized in Algorithm 1. The function $\mathcal{Z}(\pi)$ denotes the specific method used to approximate the value function for the policy π. The two most commonly used methods—*Bellman residual approximation* and *least-squares approximation* (Lagoudakis and Parr, 2003)—minimize the L_2 norm of the Bellman residual.

The approximations based on minimizing L_2 norm of the Bellman residual are common in practice since they are easy to compute and often lead to good results. Most theoretical analyses of API, however, assume minimization of the L_∞ norm of the Bellman residual:

$$\mathcal{Z}(\pi) \in \arg\min_{v \in \tilde{\mathcal{V}}} \|(\mathbf{I} - \gamma P_\pi)v - r_\pi\|_\infty .$$

L_∞-API is shown in Algorithm 1, where $\mathcal{Z}(\pi)$ is calculated by solving the following linear program:

$$\mathcal{Z}(\pi) = \min_{\sigma, v} \left\{ \sigma \,\middle|\, (\mathbf{I} - \gamma P_\pi)v + \mathbf{1}\sigma \geq r_\pi,\ -(\mathbf{I} - \gamma P_\pi)v + \mathbf{1}\sigma \geq -r_\pi, v \in \tilde{\mathcal{V}} \right\} .$$

We are not aware of convergence or divergence proofs of L_∞-API, and such analysis is beyond the scope of this paper. Theoretically, it is also possible to minimize the L_1 norm of the Bellman residual, but we are not aware of any detailed study of such an approximation.

In the above description of API, we assumed that the value function is approximated for all states and actions. This is impossible in practice due to the size of the MDP. Instead, API relies on a subset of states and actions, provided as samples. API is not guaranteed to converge in general and its analysis is typically in terms of limit behavior. The limit bounds are often very loose. We discuss the performance of API and how it relates to approximate bilinear programming in more detail in Section 7.

Approximate linear programming—a method for value function approximation—is based on the linear program formulation of exact MDPs:

$$\begin{array}{ll} \min\limits_{v} & \sum\limits_{s \in \mathcal{S}} c(s) v(s) \\ \text{s.t.} & v(s) - \gamma \sum\limits_{s' \in \mathcal{S}} P(s', s, a) v(s') \geq r(s, a) \quad \forall (s, a) \in (\mathcal{S}, \mathcal{A}) . \end{array} \tag{4}$$

The value c represents a distribution over the states, usually a uniform one. That is, $\sum_{s \in \mathcal{S}} c(s) = 1$. The linear program (4) is often too large to be solved precisely, so it is approximated by assuming that $v \in \tilde{\mathcal{V}}$ (de Farias and van Roy, 2003), yielding the following *approximate linear program*:

$$\begin{array}{ll} \min\limits_{v} & c^\mathsf{T} v \\ \text{s.t.} & Av \geq b , \qquad v \in \tilde{\mathcal{V}} . \end{array} \tag{5}$$

The matrix inequality $Av \geq b$ represents the inequality in (4) and is the following for actions $a_1, a_2, \ldots, a_n \in \mathcal{A}$:

$$\begin{pmatrix} \mathbf{I} - \gamma P_{a_1} \\ \mathbf{I} - \gamma P_{a_2} \\ \vdots \end{pmatrix} = A \quad \geq \quad b = \begin{pmatrix} r_{a_1} \\ r_{a_2} \\ \vdots \end{pmatrix} .$$

The constraint $v \in \tilde{\mathcal{V}}$ denotes the value function approximation. To actually solve this linear program for the simple linear approximation (when $\tilde{\mathcal{V}} = \text{colspan}(\Phi)$), the value function is represented as $v = \Phi x$, which leads to:

$$\begin{array}{ll} \min\limits_{x} & c^\mathsf{T} \Phi x \\ \text{s.t.} & A\Phi x \geq b . \end{array}$$

Appropriate constraints can be added for other choices of $\tilde{\mathcal{V}}$.

Assumption 1 guarantees that (5) is feasible. The following lemma follows directly from the definition of $\mathcal{K}$:

Lemma 14 *A value function v satisfies $Av \geq b$ if and only if $v \in \mathcal{K}$. In addition, if $v \in \mathcal{K}$, then $v \geq v^\star$.*

Theorem 14 implies that an optimal solution $\tilde{v}$ of (5) satisfies: $\tilde{v} \geq v^\star$ from Theorem 8. As a result, the objective of (5) represents the minimization of $\|v - v^\star\|_{1,c} = c^\mathsf{T}(v - v^\star)$ (de Farias, 2002).

Approximate linear programming is guaranteed to converge to a solution and minimize a weighted L_1 norm on the solution quality.

Theorem 15 (Theorem 4.1 in de Farias, 2002) *Given Assumption 1, let $\tilde{v}$ be the solution of* (5)*. If $c = \alpha$ then:*

$$\|v^\star - \tilde{v}\|_{1,\alpha} \leq \frac{2}{1-\gamma} \min_{v \in \tilde{\mathcal{V}}} \|v^\star - v\|_\infty = \frac{2}{1-\gamma} \min_x \|v^\star - \Phi x\|_\infty \ .$$

The difficulty with the solution of ALP is that it is hard to derive guarantees on the policy loss based on the bounds in terms of the L_1 norm; it is possible when the objective function c represents u, as Theorem 13 shows. In addition, the constant $1/(1-\gamma)$ may be very large when γ is close to 1.

Approximate linear programs are often formulated in terms of samples instead of the full formulation above. The performance guarantees are then based on analyzing the probability that a large number of constraints is violated. It is generally hard to translate the constraint violation bounds to bounds on the quality of the value function and the policy.

4. Bilinear Program Formulations

This section shows how to formulate value function approximation as a separable bilinear program. Bilinear programs are a generalization of linear programs that allows the objective function to include an additional bilinear term. A separable bilinear program consists of two linear programs with independent constraints and is fairly easy to solve and analyze in comparison to non-separable bilinear programs.

Definition 16 (Separable Bilinear Program) *A* separable *bilinear program in the normal form is defined as follows:*

$$\begin{array}{lll} \min\limits_{w,x \mid y,z} & s_1^\mathsf{T} w + r_1^\mathsf{T} x + x^\mathsf{T} C y + r_2^\mathsf{T} y + s_2^\mathsf{T} z & \\ s.t. & A_1 x + B_1 w = b_1 \ , & A_2 y + B_2 z = b_2 \ , \\ & w, x \geq \mathbf{0} \ , & y, z \geq \mathbf{0} \ . \end{array} \tag{6}$$

The objective of the bilinear program (6) is denoted as $f(w,x,y,z)$. We separate the variables using a vertical line and the constraints using different columns to emphasize the separable nature of the bilinear program. In this paper, we only use *separable* bilinear programs and refer to them simply as bilinear programs.

The goal in approximate dynamic programming and value function approximation is to find a policy that is close to optimal. The set of acceptable policies is typically restricted to be greedy with respect to *representable* value functions. We define this set of policies $\tilde{\Pi} \subseteq \Pi$ as:

$$\tilde{\Pi} = \{\pi \in \Pi \,\big|\, L_\pi v = Lv, \ v \in \tilde{\mathcal{V}}\} \ .$$

We propose approximate bilinear formulations that minimize the following bounds on robust and expected policy loss.

1. *Robust policy loss*: Minimize $\|v^\star - v_\pi\|_\infty$ by minimizing the bounds in Theorem 11:

$$\min_{\pi \in \tilde{\Pi}} \|v^\star - v_\pi\|_\infty \leq \min_{v \in \tilde{\mathcal{V}}} \frac{1}{1-\gamma} \|v - Lv\|_\infty \ .$$

2. *Expected policy loss*: Minimize $\|v^\star - v_\pi\|_{1,\alpha}$ by minimizing the bounds in Theorem 12:

$$\min_{\pi\in\tilde{\Pi}} \|v^\star - v_\pi\|_{1,\alpha} \leq \alpha^\mathsf{T} v^\star + \min_{v\in\tilde{\mathcal{V}}\cap\mathcal{K}} \left(-\alpha^\mathsf{T}\tilde{v} + \frac{1}{1-\gamma}\|v - Lv\|_\infty \right) .$$

The appropriateness of each formulation depends on the particular circumstances of the domain. For example, minimizing robust bounds is advantageous when the initial distribution is not known and the performance must be consistent under all circumstances. On the other hand, minimizing expected bounds on the value function is useful when the initial distribution is known.

In the formulations described below, we initially assume that samples of all states and actions are used. This means that the precise version of the operator L is available. When solving large problems, the number of samples is often much smaller, due to either subsampling or reduction based on the structure of the MDP. While sampling in linear programs results simply in removal of constraints, in approximate bilinear programs it also leads to a reduction in the number of certain variables, as described in Section 6.

The formulations below denote the value function approximation generically by $v \in \tilde{\mathcal{V}}$. That restricts the value functions to be representable using the features. Representable value functions v can be replaced by a set of variables x as $v = \Phi x$, which reduces the number of variables to the number of features.

4.1 Robust Policy Loss

The solution of the robust approximate bilinear program minimizes the L_∞ norm of the Bellman residual $\|v - Lv\|_\infty$ over the set of representable and transitive-feasible value functions. This minimization can be formulated as follows.

$$\begin{array}{lll} \min\limits_{\pi|\lambda,\lambda',v} & \pi^\mathsf{T}\lambda + \lambda' & \\ \text{s.t.} & B\pi = \mathbf{1}\,, & \lambda + \lambda'\mathbf{1} \geq Av - b \geq \mathbf{0}\,, \\ & \pi \geq \mathbf{0}\,, & \lambda, \lambda' \geq \mathbf{0}\,, \\ & & v \in \tilde{\mathcal{V}}\,. \end{array} \tag{7}$$

All the variables are vectors except λ', which is a scalar. The values A and b are identical to the values in (5). The variables λ correspond to all state-action pairs. These variables represent the Bellman residuals that are being minimized. This formulation offers the following guarantees.

Theorem 17 *Let* $(\tilde{\pi}, \tilde{v}, \tilde{\lambda}, \tilde{\lambda'})$ *be an optimal solution of* (7) *and let*

$$v' = \tilde{v} - \frac{\|\tilde{v} - L\tilde{v}\|_\infty}{2(1-\gamma)}\mathbf{1}\,.$$

Then:

$$\begin{aligned} \tilde{\pi}^\mathsf{T}\tilde{\lambda} + \tilde{\lambda'} = \|\tilde{v} - L\tilde{v}\|_\infty &= \min_{v\in\mathcal{K}\cap\tilde{\mathcal{V}}} \|v - Lv\|_\infty \\ \|v' - Lv'\|_\infty &= \min_{v\in\tilde{\mathcal{V}}} \|v - Lv\|_\infty \\ &\leq (1+\gamma) \min_{v\in\tilde{\mathcal{V}}} \|v - v^\star\|_\infty\,. \end{aligned}$$

In addition, there exists an optimal $\tilde{\pi} \in \tilde{\Pi}$.

It is important to note that the theorem states that solving the approximate bilinear program is equivalent to minimization over *all* representable value functions, not only the transitive-feasible ones. This follows by subtracting a constant vector $\mathbf{1}$ from $\tilde{v}$ to balance the lower bounds on the Bellman residual error with the upper ones as Theorem 20 shows. This reduces the Bellman residual by $1/2$ without affecting the policy. Finally, note that whenever $v^\star \in \tilde{\mathcal{V}}$, both ABP and ALP will return the optimal value function $v^\star$. The following corollary follows from Theorem 11 and Theorem 17 applied to v'.

Corollary 18 *For any optimal solution $\tilde{v}$ of* (7)*, the policy loss of the greedy policy $\tilde{\pi}$ is bounded by:*

$$\|v^\star - v_{\tilde{\pi}}\|_\infty = \frac{2}{1-\gamma} \min_{v \in \tilde{\mathcal{V}}} \|v - Lv\|_\infty \ .$$

To prove Theorem 17, we first define the following linear programs.

$$f_1(\pi, v) = \min_{\lambda, \lambda'} \left\{ \pi^\mathsf{T} \lambda + \lambda' \,\middle|\, \mathbf{1}\lambda' + \lambda \geq Av - b, \lambda \geq \mathbf{0} \right\} \ ,$$
$$f_2(v) = \min_{\pi} \left\{ f_1(\pi, v) \,|\, B\pi = \mathbf{1}, \pi \geq \mathbf{0} \right\} \ .$$

Assuming that $f^\star$ is the optimal solution of (7), then:

$$f^\star = \min_{\pi \in \Pi, v \in \tilde{\mathcal{V}} \cap \mathcal{K}} f_1(\pi, v) = \min_{v \in \tilde{\mathcal{V}} \cap \mathcal{K}} f_2(v) \ .$$

Lemma 19 *Let $v \in \mathcal{K}$ be a transitive-feasible value function and let π be a policy. Then:*

$$f_1(\pi, v) \geq \|v - L_\pi v\|_\infty \ , \tag{8}$$
$$f_2(v) = \|v - Lv\|_\infty \ . \tag{9}$$

In addition, inequality (8) *becomes an equality for any* deterministic *policy π, and there is a deterministic optimal policy that satisfies equality* (9).

Proof To prove (8), notice that for all $s \in \mathcal{S}$ we have that $\sum_{a \in \mathcal{A}} \pi(s,a) = 1$ and $\pi(s,a) \geq 0$. Then:

$$\begin{aligned}
f_1(\pi, v) &\overset{(8)}{=} \lambda' + \sum_{s \in \mathcal{S}, a \in \mathcal{A}} \lambda(s,a)\pi(s,a) \\
&\overset{\lambda(s,a) \geq 0}{\geq} \lambda' + \max_{s \in \mathcal{S}} \sum_{a \in \mathcal{A}} \lambda(s,a)\pi(s,a) \\
&= \max_{s \in \mathcal{S}} \sum_{a \in \mathcal{A}} \pi(s,a)(\lambda' + \lambda(s,a)) \\
&\geq \max_{s \in \mathcal{S}} \sum_{a \in \mathcal{A}} \pi(s,a) \sum_{s' \in \mathcal{S}} (\gamma P(s,a,s')v(s') + r(s,a)) \\
&= \|v - L_\pi v\|_\infty \ .
\end{aligned}$$

To show the equality for a deterministic policy, set $\lambda' = \|v - L_\pi v\|_\infty$ and $\lambda(s, \pi(s)) = \mathbf{0}$ with other elements of λ set arbitrarily. This can be readily shown to be an optimal solution.

To prove (9), note again that $v \in \mathcal{K}$, which implies that $v \geq Lv$. Then, using the fact that the policy defines an action for every state, we get:

$$\begin{aligned} f_2(v) &= \min_{\pi\in\Pi} \|v - L_\pi v\|_\infty = \min_{\pi\in\Pi} \max_{s\in\mathcal{S}} (v - L_\pi v)(s) \\ &= \max_{s\in\mathcal{S}} \min_{\pi\in\Pi} (v - L_\pi v)(s) \\ &= \max_{s\in\mathcal{S}} (v - \max_{\pi\in\Pi} L_\pi v)(s) \\ &= \max_{s\in\mathcal{S}} (v - Lv)(s) = \|v - Lv\|_\infty \ . \end{aligned}$$

The existence of an optimal deterministic solution then follows from the existence of a deterministic greedy policy with respect to a value function. ■

Now, we show that restricting the value functions to be transitive feasible is not limiting, because it does not restrict the set of greedy policies that are considered. To do that, we define the following sets:

$$\mathcal{V}_1 = \arg \min_{v\in\tilde{\mathcal{V}}\cap\mathcal{K}} \|v - Lv\|_\infty \ , \qquad \mathcal{V}_2 = \arg\min_{v\in\tilde{\mathcal{V}}} \|v - Lv\|_\infty \ .$$

Let Π_1 and Π_2 be sets of greedy policies with respect to $\mathcal{V}_1$ and $\mathcal{V}_2$. The sets $\mathcal{V}_1$ and $\mathcal{V}_2$ satisfy the following important property.

Lemma 20 *Given Assumption 1, let $v_1 \in \mathcal{V}_1$ and $v_2 \in \mathcal{V}_2$, we have the following equalities:*

$$\min_{s\in\mathcal{S}} (v_1 - Lv_1)(s) = 0 \ , \qquad -\min_{s\in\mathcal{S}} (v_2 - Lv_2)(s) = \max_{s\in\mathcal{S}} (v_2 - Lv_2)(s) \ .$$

Then, define:

$$v_1' = v_1 - \frac{\|v_1 - Lv_1\|_\infty}{2(1-\gamma)}\mathbf{1} \ , \qquad v_2' = v_2 + \frac{\|v_2 - Lv_2\|_\infty}{(1-\gamma)}\mathbf{1} \ .$$

for which the following holds:

$$\min_{s\in\mathcal{S}} (v_2' - Lv_2')(s) = 0 \ , \qquad -\min_{s\in\mathcal{S}} (v_1' - Lv_1')(s) = \max_{s\in\mathcal{S}} (v_1' - Lv_1')(s) \ .$$

Proof Assume, for the sake of deriving a contradiction, that $\min_{s\in\mathcal{S}} (v_1 - Lv_1)(s) = \varepsilon > 0$. Then, let $v_1 = v_1 - \varepsilon/(1-\gamma)\mathbf{1} \in \mathcal{K}$ which implies the following by Theorem 6:

$$\begin{aligned} \|v_1 - Lv_1\|_\infty &= \|v_1 - Lv_1 - \varepsilon\mathbf{1}\|_\infty = \max_{s\in\mathcal{S}} (v_1 - Lv_1 - \varepsilon\mathbf{1})(s) \\ &= \max_{s\in\mathcal{S}} (v_1 - Lv_1)(s) - \varepsilon = \|v_1 - Lv_1\|_\infty - \varepsilon \\ &< \|v_1 - Lv_1\|_\infty \ . \end{aligned}$$

This contradicts the optimality of v_1. The inequality for v_2 follows similarly. The rest of the lemma is a simple consequence of Theorem 6. ■

We are now ready to show that neither the set of greedy policies considered nor the policy loss bounds are affected by considering only transitive feasible functions in (7).

Proposition 21 *Given Assumption 1, the following holds:*

$$\begin{aligned} \mathcal{V}_1 &= \left\{ v_2 + \frac{\|v_2 - Lv_2\|_\infty}{(1-\gamma)}\mathbf{1} \,\middle|\, v_2 \in \mathcal{V}_2 \right\} , \\ \|v_1 - Lv_1\|_\infty &= 2\|v_2 - Lv_2\|_\infty \qquad \forall v_1 \in \mathcal{V}_1, \forall v_2 \in \mathcal{V}_2 , \\ \Pi_1 &= \Pi_2 . \end{aligned}$$

Proof To show that $\mathcal{V}_1 \subseteq \left\{ v_2 + \frac{\|v_2 - Lv_2\|_\infty}{(1-\gamma)}\mathbf{1} \,\middle|\, v_2 \in \mathcal{V}_2 \right\}$, assume a $v_1 \in \mathcal{V}_1$ and define:

$$v_2 = v_1 - \frac{\|v_1 - Lv_1\|_\infty}{2(1-\gamma)}\mathbf{1} .$$

Note that $v_2 \in \mathcal{V}$ from Assumption 1, and $2\|v_2 - Lv_2\|_\infty = \|v_1 - Lv_1\|_\infty$ from Theorem 20. To show that $v_2 \in \mathcal{V}_2$ by contradiction, assume that there exists $v_2 \in \mathcal{V}_2$ such that $\|v_2 - Lv_2\|_\infty < \|v_2 - Lv_2\|_\infty$ and let $v_1 = v_2 + \frac{\|v_2 - Lv_2\|_\infty}{(1-\gamma)}\mathbf{1}$. Using Theorem 20, we get:

$$\|v_1 - Lv_1\|_\infty = 2\|v_2 - Lv_2\|_\infty < 2\|v_2 - Lv_2\|_\infty = \|v_1 - Lv_1\|_\infty ,$$

which contradicts the optimality of v_1.

The inclusion $\mathcal{V}_1 \supseteq \left\{ v_2 - \frac{\|v - Lv\|_\infty}{2(1-\gamma)}\mathbf{1} \,\middle|\, v_2 \in \mathcal{V}_2 \right\}$ and $\Pi_1 \supseteq \Pi_2$ can be shown similarly. Finally, Theorem 6 implies that $\Pi_1 = \Pi_2$. ■

Proposition 22 *Given Assumption 1, the minimal Bellman residual for a representable value function can be bounded as follows:*

$$\min_{v \in \tilde{\mathcal{V}}} \|Lv - v\|_\infty \leq (1+\gamma) \min_{v \in \tilde{\mathcal{V}}} \|v - v^\star\|_\infty .$$

Proof Assume that $\hat{v}$ minimizes $\min_{v \in \tilde{\mathcal{V}}} \|v - v^\star\|_\infty \leq \varepsilon$. Then:

$$\begin{aligned} v^\star - \varepsilon\mathbf{1} \leq \quad & v \quad & \leq v^\star + \varepsilon\mathbf{1} , \\ Lv^\star - \gamma\varepsilon\mathbf{1} \leq \quad & Lv \quad & \leq Lv^\star + \gamma\varepsilon\mathbf{1} , \\ Lv^\star - \gamma\varepsilon\mathbf{1} - v \leq \quad & Lv - v \quad & \leq Lv^\star + \gamma\varepsilon\mathbf{1} - v , \\ Lv^\star - v^\star - (1+\gamma)\varepsilon\mathbf{1} \leq \quad & Lv - v \quad & \leq Lv^\star - v^\star + (1+\gamma)\varepsilon\mathbf{1} , \\ -(1+\gamma)\varepsilon\mathbf{1} \leq \quad & Lv - v \quad & \leq (1+\gamma)\varepsilon\mathbf{1} . \end{aligned}$$

■

Theorem 17 now easily follows from the results above.

Proof [Proof of Theorem 17] Let $f^\star$ be the optimal objective value of (7). Then we have from Theorem 19 that:

$$f^\star = \min_{\pi \in \Pi, v \in \tilde{\mathcal{V}} \cap \mathcal{K}} f_1(\pi, v) = \min_{v \in \tilde{\mathcal{V}} \cap \mathcal{K}} f_2(v) = \min_{v \in \tilde{\mathcal{V}} \cap \mathcal{K}} \|v - Lv\|_\infty .$$

The properties of v' follow directly from Theorem 21:

$$\tilde{v} \in \mathcal{V}_1 \Rightarrow v' \in \mathcal{V}_2 \Rightarrow \|v' - Lv'\|_\infty = \min_{v \in \tilde{\mathcal{V}}} \|v - Lv\|_\infty .$$

Note that the existence of an optimal deterministic policy in (7) follows from the existence of a deterministic optimal policy in f_2. The bound on the minimal Bellman residual follows from Theorem 22. ■

4.2 Expected Policy Loss

This section describes bilinear programs that minimize bounds on the expected policy loss for a given initial distribution $\|v - Lv\|_{1,\alpha}$. The initial distribution can be used to derive tighter bounds on the policy loss. We describe two formulations. They respectively minimize an L_∞ and a weighted L_1 norm on the Bellman residual.

The expected policy loss can be minimized by solving the following bilinear formulation.

$$\begin{array}{lll} \min\limits_{\pi|\lambda,\lambda',v} & \pi^\mathsf{T}\lambda + \lambda' - (1-\gamma)\alpha^\mathsf{T} v & \\ \text{s.t.} & B\pi = \mathbf{1}\,, & Av - b \geq \mathbf{0}\,, \\ & \pi \geq \mathbf{0}\,, & \lambda + \lambda'\mathbf{1} \geq Av - b\,, \\ & & \lambda, \lambda' \geq \mathbf{0}\,, \\ & & v \in \tilde{\mathcal{V}}\,. \end{array} \tag{10}$$

Notice that this formulation is identical to the bilinear program (7) with the exception of the term $-(1-\gamma)\alpha^\mathsf{T} v$.

Theorem 23 *Given Assumption 1, any optimal solution* $(\tilde{\pi}, \tilde{v}, \tilde{\lambda}, \tilde{\lambda'})$ *of* (10) *satisfies:*

$$\begin{aligned} \frac{1}{1-\gamma}\left(\tilde{\pi}^\mathsf{T}\tilde{\lambda} + \tilde{\lambda'}\right) - \alpha^\mathsf{T}\tilde{v} &= \frac{1}{1-\gamma}\|L\tilde{v} - \tilde{v}\|_\infty - \alpha^\mathsf{T}\tilde{v} \\ &= \min_{v \in \mathcal{K} \cap \tilde{\mathcal{V}}} \left(\frac{1}{1-\gamma}\|Lv - v\|_\infty - \alpha^\mathsf{T} v\right) \\ &= \min_{v \in \tilde{\mathcal{V}}} \left(\frac{1}{1-\gamma}\|Lv - v\|_\infty - \alpha^\mathsf{T} v\right)\,. \end{aligned}$$

In addition, there exists an optimal $\tilde{\pi} \in \tilde{\Pi}$.

The following bound on the policy loss follows using Theorem 12.

Corollary 24 *There exists an optimal solution* $\tilde{\pi}$ *that is greedy with respect to* $\tilde{v}$ *for which the policy loss is bounded by:*

$$\begin{aligned} \|v^\star - v_{\tilde{\pi}}\|_{1,\alpha} &\leq \left(\min_{v \in \tilde{\mathcal{V}} \cap \mathcal{K}} \frac{1}{1-\gamma}\|Lv - v\|_\infty - \|v^\star - v\|_{1,\alpha}\right) \\ &\leq \left(\min_{v \in \tilde{\mathcal{V}}} \frac{1}{1-\gamma}\|Lv - v\|_\infty - \alpha^\mathsf{T}(v - v^\star)\right)\,. \end{aligned}$$

Proof The proof of Theorem 23 is similar to the proof of Theorem 17 with the main difference being in the definition of functions f_1, and f_2:

$$f_1(\pi, v) = \min_{\lambda,\lambda'} \left\{ \pi^\mathsf{T}\lambda + \lambda' - (1-\gamma)\alpha^\mathsf{T} v \,\middle|\, \mathbf{1}\lambda' + \lambda \geq Av - b, \lambda \geq \mathbf{0} \right\} ,$$
$$f_2(v) = \min_{\pi} \{ f_1(\pi, v) \,|\, B\pi = \mathbf{1}, \pi \geq \mathbf{0} \} .$$

The following lemma can be proved identically to Theorem 19.

Lemma 25 *Let $v \in \mathcal{K}$ be a transitive-feasible value function and let π be a policy. Then:*

$$f_1(\pi, v) \geq \|v - L_\pi v\|_\infty - \|v^\star - v\|_{1,\alpha} , \tag{11}$$
$$f_2(v) = \|v - Lv\|_\infty - \|v^\star - v\|_{1,\alpha} . \tag{12}$$

In addition, (11) *holds with an equality for a* deterministic *policy π, and there is a deterministic optimal policy that satisfies* (12).

The fact that optimization over transitive-feasible value functions does not restrict the resulting policies is proved identically to Theorem 21 with sets $\mathcal{V}_1$ and $\mathcal{V}_2$ that satisfy the same equations. Notice also that the objective function of (10) does not change when subtracting a constant for $v' = v - k\mathbf{1}$ and $k \geq 0$:

$$\frac{1}{1-\gamma}\|v' - Lv'\|_\infty - \alpha^\mathsf{T} v' = \frac{1}{1-\gamma}\|v - Lv\|_\infty - \alpha^\mathsf{T} v ,$$

when $-\min_{s\in\mathcal{S}}(v' - Lv')(s) = \max_{s\in\mathcal{S}}(v' - Lv')(s)$ and $\min_{s\in\mathcal{S}}(v - Lv)(s) = 0$. ■

5. Solving Bilinear Programs

This section describes methods for solving approximate bilinear programs. Bilinear programs can be easily mapped to other global optimization problems, such as mixed integer linear programs (Horst and Tuy, 1996). We focus on a simple iterative algorithm for solving bilinear programs approximately, which also serves as a basis for many optimal algorithms. In addition, we provide a basic problem-specific mixed integer program formulation.

Solving a bilinear program is an NP-complete problem (Bennett and Mangasarian, 1993). The membership in NP follows from the finite number of basic feasible solutions of the individual linear programs, each of which can be checked in polynomial time. The NP-hardness is shown by a reduction from the SAT problem.

There are two main approaches for solving bilinear programs optimally. In the first approach, a relaxation of the bilinear program is solved. The solution of the relaxed problem represents a lower bound on the optimal solution. The relaxation is then iteratively refined, for example by adding cutting plane constraints, until the solution becomes feasible. This is a common method used to solve integer linear programs. The relaxation of the bilinear program is typically either a linear or semi-definite program (Carpara and Monaci, 2009).

In the second approach, feasible, but suboptimal, solutions of the bilinear program are calculated approximately. The approximate algorithms are usually some variation of Algorithm 2. The bilinear

Algorithm 2: Iterative algorithm for solving (6)

$(x_0, w_0) \leftarrow$ random ;
$(y_0, z_0) \leftarrow \arg\min_{y,z} f(w_0, x_0, y, z)$;
$i \leftarrow 1$;
while $y_{i-1} \neq y_i$ *or* $x_{i-1} \neq x_i$ **do**
    $(y_i, z_i) \leftarrow \arg\min_{\{y,z | A_2 y + B_2 z = b_2,\ y,z \geq \mathbf{0}\}} f(w_{i-1}, x_{i-1}, y, z)$;
    $(x_i, w_i) \leftarrow \arg\min_{\{x,w | A_1 x + B_1 w = b_1,\ x,w \geq \mathbf{0}\}} f(w, x, y_i, z_i)$;
    $i \leftarrow i + 1$
return $f(w_i, x_i, y_i, z_i)$

program formulation is then refined—using concavity cuts (Horst and Tuy, 1996)—to eliminate previously computed feasible solutions and solved again. This procedure can be shown to find the optimal solution by eliminating all suboptimal feasible solutions.

The most common and simplest approximate algorithm for solving bilinear programs is Algorithm 2. This algorithm is shown for the general bilinear program (6), where $f(w,x,y,z)$ represents the objective function. The minimizations in the algorithm are linear programs which can be easily solved. Interestingly, as we will show in Section 7, Algorithm 2 applied to ABP generalizes a version of API.

While Algorithm 2 is not guaranteed to find an optimal solution, its empirical performance is often remarkably good (Mangasarian, 1995). Its basic properties are summarized by the following proposition.

Proposition 26 (Theorem 2.2 in Bennett and Mangasarian, 1993) *Algorithm 2 is guaranteed to converge, assuming that the linear program solutions are in a vertex of the optimality simplex. In addition, the global optimum is a fixed point of the algorithm, and the objective value monotonically improves during execution.*

The proof is based on the finite count of the basic feasible solutions of the individual linear programs. Because the objective function does not increase in any iteration, the algorithm will eventually converge.

As mentioned above, any separable bilinear program can be also formulated as a mixed integer linear program (Horst and Tuy, 1996). Such formulation is not practical in our setting, because its size grows quadratically with the size of ABP and its linear relaxations were very loose in our experiments. Below, we present a more compact and structured mixed integer linear program formulation, which relies on the property of ABP that there is always a solution with an optimal deterministic policy (see Theorem 17).

We only show the formulation of the robust approximate bilinear program (7); the same approach applies to all other formulations that we propose. To formulate the mixed integer linear program, assume a given upper bound $\tau \in \mathbb{R}$ for the optimal solution $\lambda^\star$ and all $s \in \mathcal{S}$ and $a \in \mathcal{A}$

such that $\tau \geq \lambda^{\star}(s,a)$. The mixed integer linear program formulation that corresponds to (7) is:

$$\begin{array}{ll} \min\limits_{z,\pi,\lambda,\lambda',v} & \mathbf{1}^{\mathsf{T}} z + \lambda' \\ \text{s.t.} & z \geq \lambda - \tau(\mathbf{1} - \pi) \ , \\ & B\pi = \mathbf{1} \ , \\ & \lambda + \lambda' \mathbf{1} \geq Av - b \geq \mathbf{0} \ , \\ & \lambda, \lambda' \geq \mathbf{0} \ , \qquad v \in \tilde{\mathcal{V}} \ , \qquad \pi \in \{0,1\}^{|\mathcal{S}||\mathcal{A}|} \ . \end{array} \tag{13}$$

The following theorem states the correctness of this formulation:

Theorem 27 *Let* $(\pi_1, \lambda_1, \lambda_1')$ *be an optimal (greedy-policy) solution of* (7) *and let* $\tau \geq \lambda_1$. *Then:*

$$\left(\pi_1, \lambda_1, \lambda_1', z' = \min_{z \geq \lambda_1 - (\tau - \pi_1)} \mathbf{1}^{\mathsf{T}} z \right)$$

is an optimal solution of (13) *and vice versa*. *When in addition* f_1 *and* f_2 *are the optimal objective values of* (7) *and* (13), *then* $f_1 = f_2$.

Proof First, we show that $(\pi_1, \lambda_1, \lambda_1', z = \min_{z \geq \lambda_1 - (\tau - \pi_1)} \mathbf{1}^{\mathsf{T}} z)$ is feasible in (13) and has the same objective value. Since π_1 is a greedy policy (see Theorem 17), then $\pi_1 \in \{0,1\}^{\mathcal{S} \times \mathcal{A}}$. That is π_1 is feasible in (13). Let then:

$$z_2(s,a) = \begin{cases} \lambda(s,a) & \text{if } \pi_1(s,a) = 1 \\ 0 & \text{otherwise} \end{cases} .$$

To show that z_2 is feasible in (13), analyze the following two cases:

$$\begin{aligned} \pi(s,a) = 1 : & \qquad z_2(s,a) + \tau(s,a)(1 - \pi_1(s,a)) = z_2(s,a) = \lambda_1(s,a) \ , \\ \pi(s,a) = 0 : & \qquad z_2(s,a) + \tau(s,a)(1 - \pi_1(s,a)) \geq \tau(s,a) \geq \lambda_1(s,a) \ . \end{aligned}$$

The objective values must then be identical based on a simple algebraic manipulation. The reverse direction—showing that for any solution of (13) there is a solution of (7) with the same objective value—follows similarly. ■

This mixed integer linear program formulation is much simpler than a general MILP formulation of a bilinear program (Horst and Tuy, 1996).

The performance of the proposed solution methods strongly depends on the actual structure of the problem. As usual with NP-hard problems, there is very little understanding of the theoretical properties that could guarantee faster solution methods. Experimental results, however, show that the ABP-specific formulation can solve problems that are orders of magnitude larger than those that can be solved by the general MILP formulation of ABP.

6. Sampling Guarantees

Typically, the number of states in an MDP is too large to be explicitly enumerated, making it hard to solve even when the value function is restricted to be representable. The usual approach is to sample a limited number of states, actions, and their transitions in order to approximately calculate

the value function. This section shows basic properties of the samples that are sufficient to establish solution quality guarantees with incomplete samples. To derive sampling bounds, we assume in this section that the representable value functions are regularized.

First, we formally define samples and then show how to use them to compute a solution. The samples of the simplest type are defined as follows.

Definition 28 One-step simple samples *are defined as:*

$$\tilde{\Sigma} \subseteq \{(s,a,(s_1 \dots s_n),r(s,a)) \,|\, s,s' \in \mathcal{S},\ a \in \mathcal{A}\}\ ,$$

where $s_1 \dots s_n$ are selected i.i.d. from the distribution $P(s,a,\cdot)$.

Note that $\tilde{\Sigma}$ represents an arbitrary subset of states and actions and may or may not be sampled from a distribution. More informative samples include the full distribution $P(s,a,\cdot)$ instead of samples from the distribution. While these samples are often unavailable in practice, they are useful in the theoretical analysis of sampling issues.

Definition 29 One-step samples with expectation *are defined as follows:*

$$\Sigma \subseteq \{(s,a,P(s,a,\cdot),r(s,a)) \,|\, s \in \mathcal{S},\ a \in \mathcal{A}\}\ ,$$

where $P(s,a,\cdot)$ is the distribution over the next states.

The membership of a state in the samples is denoted simply as $s \in \tilde{\Sigma}$ or $(s,a) \in \Sigma$ with the remaining variables, such as $r(s,a)$ considered to be available implicitly.

The sampling models may vary significantly in different domains. The focus of this work is on problems with either a fixed set of available samples or a domain model. Therefore, we do not analyze methods for gathering samples. We also do not assume that the samples come from previous executions, but rather from a deliberate sample-gathering process.

The samples are used to approximate the Bellman operator and the set of transitive-feasible value functions as the following definitions describe.

Definition 30 *The* sampled Bellman operator *and the corresponding set of sampled transitive-feasible functions are defined as:*

$$(L(v))(s) = \max_{\{a \,|\, (s,a) \in \Sigma\}} r(s,a) + \gamma \sum_{s' \in \mathcal{S}} P(s,a,s')v(s') \qquad \forall s \in \Sigma$$
$$\mathcal{K} = \left\{v \,\middle|\, (s,a,P(s,a),r(s,a)) \in \Sigma,\ v(s) \geq (Lv)(s)\right\}\ .$$

The less-informative set of samples $\tilde{\Sigma}$ can be used as follows.

Definition 31 *The* estimated Bellman operator *and the corresponding set of estimated transitive-feasible functions are defined as:*

$$(\tilde{L}(v))(s) = \max_{\{a \,|\, (s,a) \in \tilde{\Sigma}\}} r(s,a) + \gamma \frac{1}{n} \sum_{i=1}^{n} v(s_i) \qquad \forall s \in \tilde{\Sigma}$$
$$\tilde{\mathcal{K}} = \left\{v \,\middle|\, (s,a,(s_1 \dots s_n),r(s,a)) \in \tilde{\Sigma},\ v(s) \geq (\tilde{L}v)(s)\right\}\ .$$

Notice that operators $\tilde{L}$ and L map value functions to a subset of all states—only states that are sampled. The values for other states are not defined here; they would be defined in a problem-specific way as, for example, the proof of Theorem 32 shows.

The samples can also be used to create an approximation of the initial distribution, or the distribution of visitation frequencies of a given policy. The estimated initial distribution α is defined as:

$$\alpha(s) = \begin{cases} \alpha(s) & (s,\cdot,\cdot,\cdot) \in \Sigma \\ 0 & \text{otherwise} \end{cases} .$$

Most existing sampling bounds for approximate linear programming focus on bounding the probability that a large number of constraints is violated when assuming a distribution over the constraints (de Farias and van Roy, 2004). The difficulty with this approach is that the number of violated constraints does not easily translate to bounds on the quality of the value function, or the policy. In addition, the constraint distribution assumed in the bounds of de Farias and van Roy (2004) is often somewhat arbitrary with no implication on solution quality.

Our approach, on the other hand, is to define properties of the sampled operators that guarantee that the sampling error bounds are small. These bounds do not rely on distributions over constraints and transform directly to bounds on the policy loss. To define bounds on the sampling behavior, we propose the following assumptions. The first assumption limits the error due to missing transitions in the sampled Bellman operator L.

Assumption 2 (Constraint Sampling Behavior) *There exists $\varepsilon_p \geq 0$ such that for all $v \in \tilde{\mathcal{V}}$:*

$$Lv - \varepsilon_p \mathbf{1} \leq Lv \leq Lv .$$

Notice that Assumption 2 implies that:

$$\mathcal{K} \subseteq \mathcal{K} \subseteq \mathcal{K}(\varepsilon_p) .$$

The second assumption quantifies the error on the estimation of the transitions of the estimated Bellman operator $\tilde{L}$.

Assumption 3 (Constraint Estimation Behavior) *There exists $\varepsilon_s \geq 0$ such that for all $v \in \tilde{\mathcal{V}}$:*

$$Lv - \varepsilon_s \mathbf{1} \leq \tilde{L}v \leq Lv + \varepsilon_s \mathbf{1} .$$

Notice that Assumption 3 implies that:

$$\mathcal{K}(-\varepsilon_s) \subseteq \tilde{\mathcal{K}} \subseteq \mathcal{K}(\varepsilon_s) .$$

Assumptions 2 and 3 are intentionally generic, so that they apply to a wide range of scenarios. They can be easily satisfied, for example, by making the following Lipschitz continuity assumptions on state features, transitions and rewards.

Assumption 4 *Let $k : \mathcal{S} \to \mathbb{R}^n$ be a map of the state-space to a normed vector space. Then for all $s_1, s_2, s_3 \in \mathcal{S}$ and all features (columns) $\phi_i \in \Phi$, we define K_r, K_P, and K_ϕ such that*

$$\begin{aligned} |r(s_1) - r(s_2)| &\leq K_r \, \|k(s_1) - k(s_2)\| , \\ |p(s_3|s_1,a) - p(s_3|s_2,a)| &\leq K_P \, \|k(s_1) - k(s_2)\| \quad \forall a \in \mathcal{A} , \\ |\phi_i(s_1) - \phi_i(s_2)| &\leq K_\phi \, \|k(s_1) - k(s_2)\| . \end{aligned}$$

This assumption can be used to provide bounds on the sampling error as follows; for more details on tighter and more general bounds see Petrik (2010).

Proposition 32 *Given Assumptions 1 and 4 and that* $\tilde{\mathcal{V}} = \{\Phi x + l\mathbf{1} \,|\, \|x\|_1 \leq \psi, l \in \mathbb{R}\}$*, then Assumption 2 holds with:*

$$\varepsilon_p = \left(K_r + \psi(K_\phi + \gamma K_P)\right) \max_{s \in \mathcal{S}} \min_{s \in \Sigma} \|k(s) - k(s)\| \ .$$

Note that $\tilde{\mathcal{V}}$ as defined in Theorem 32 satisfies Assumption 1.

Proof Assume that there exists a constant q such that:

$$\max_{s \in \mathcal{S}} \min_{s \in \Sigma} \|k(s) - k(s)\| \leq q \ .$$

Also, define a function $\chi : \mathcal{S} \to \mathcal{S}$ that maps each state to the closest sample as follows:

$$\chi(s) = \arg\min_{s \in \Sigma} \|k(s) - k(s)\| \ .$$

We will use the following simple extension of Holder's inequality to prove the proposition.

Lemma 33 *The following holds for any* $v \in \tilde{\mathcal{V}} = \{\Phi x + l\mathbf{1} \,|\, \|x\|_1 \leq \psi, l \in \mathbb{R}\}$ *and any* y *such that* $\mathbf{1}^\mathsf{T} y = 0$.

$$|y^\mathsf{T} v| \leq |y|^\mathsf{T} |v| \leq \psi \|\Phi y\|_\infty \ .$$

Assumption 4 directly implies the following inequalities:

$$\begin{aligned} \|\phi(\chi(s)) - \phi(s)\|_\infty &\leq q K_\phi \ , \\ |r(\chi(s)) - r(s)| &\leq q K_r \ , \\ \|P(\chi(s), a)^\mathsf{T} \phi_i - P(s,a)^\mathsf{T} \phi_i\|_\infty &\leq q K_p \quad \forall a \in \mathcal{A} \ . \end{aligned}$$

The proposition now follows using simple algebraic manipulation as:

$$\begin{aligned} &\max_{s \in \mathcal{S}} |(v - Lv)(s) - (v - Lv)(\chi(s))| \\ &\qquad \leq \max_{s \in \mathcal{S}, a \in \mathcal{A}} |(v - \gamma P_a v - r_a)(s) - (v - \gamma P_a v - r_a)(\chi(s))| \\ &\qquad \leq \max_{s \in \mathcal{S}, a \in \mathcal{A}} |\mathbf{1}_s^\mathsf{T} (\Phi x - \gamma P_a \Phi x - r_a) - \mathbf{1}_{\chi(s)}^\mathsf{T} (\Phi x - \gamma P_a \Phi x - r_a)| \\ &\qquad \leq \max_{s \in \mathcal{S}, a \in \mathcal{A}} |(\mathbf{1}_s^\mathsf{T} - \mathbf{1}_{\chi(s)}^\mathsf{T}) \Phi x| + |(\mathbf{1}_s^\mathsf{T} - \mathbf{1}_{\chi(s)}^\mathsf{T}) \gamma P_a \Phi x| + \\ &\qquad\quad + |(\mathbf{1}_s^\mathsf{T} - \mathbf{1}_{\chi(s)}^\mathsf{T}) r_a| \\ &\qquad \overset{\text{Theorem 33}}{\leq} \max_{s \in \mathcal{S}, a \in \mathcal{A}} \|(\mathbf{1}_s^\mathsf{T} - \mathbf{1}_{\chi(s)}^\mathsf{T}) \Phi\|_\infty \psi + \\ &\qquad\quad + \|(\mathbf{1}_s^\mathsf{T} - \mathbf{1}_{\chi(s)}^\mathsf{T}) \gamma P_a \Phi\|_\infty \psi + \|(\mathbf{1}_s^\mathsf{T} - \mathbf{1}_{\chi(s)}^\mathsf{T}) r_a\|_\infty \\ &\qquad \leq q K_r + q\psi(K_\phi + \gamma K_p) \ , \end{aligned}$$

where the last inequality follows from Assumption 4. ■

In practice, the estimated Bellman operator is used to formulate the approximate bilinear program. Then, the matrices used in the sampled approximate bilinear program (7) are defined as follows for all $(s_i, a_j) \in \tilde{\Sigma}$.

$$\tilde{A}\Phi = \begin{pmatrix} - & \phi(s_i)^\mathsf{T} - \gamma \frac{1}{m} \sum_{s' \in s'_1 \ldots s'_m} P(s_i, a_j, s') \phi(s')^\mathsf{T} & - \\ - & \vdots & - \end{pmatrix}, \qquad \tilde{b} = \begin{pmatrix} r(s_i, a_j) \\ \vdots \end{pmatrix},$$
$$\tilde{B}(s', (s_i, a_j)) = \mathrm{I}\{s' = s_i\} \quad \forall s' \in \tilde{\Sigma}.$$

The ordering over states in the definitions above is also assumed to be consistent. The sampled version of the bilinear program (7) is then:

$$\begin{array}{lll} \min\limits_{\pi | \lambda, \lambda', x} & \pi^\mathsf{T} \lambda + \lambda' & \\ \text{s.t.} & \tilde{B}\pi = \mathbf{1}, & \tilde{A}\Phi x - b \geq \mathbf{0}, \\ & \pi \geq \mathbf{0}, & \lambda + \lambda' \mathbf{1} \geq \tilde{A}\Phi x - \tilde{b}, \\ & & \lambda, \lambda' \geq \mathbf{0}. \end{array} \tag{14}$$

The size of the bilinear program (14) scales with the number of samples and features, not with the size of the full MDP, because the variables λ and π are defined only for state-action pairs in $\tilde{\Sigma}$. That is, $|\pi| = |\lambda| = |\{(s,a) \in \Sigma\}|$. The number of constraints in (14) is approximately three times the number of variables λ. Finally, the number of variables x corresponds to the number of approximation features.

Theorem 17 shows that sampled robust ABP minimizes $\|v - \tilde{L}v\|_\infty$ or $\|v - Lv\|_\infty$. We are now ready to derive sampling bounds on these values that rely on Assumptions 2 and 3 defined above.

Theorem 34 *Let the optimal solutions to the sampled and precise Bellman residual minimization problems be:*

$$v_1 \in \min_{v \in \tilde{\mathcal{V}}} \|v - Lv\|_\infty, \qquad v_2 \in \min_{v \in \tilde{\mathcal{V}}} \|v - Lv\|_\infty, \qquad v_3 \in \min_{v \in \tilde{\mathcal{V}}} \|v - \tilde{L}v\|_\infty.$$

Value functions v_1, v_2, v_3 correspond to solutions of instances of robust approximate bilinear programs for the given samples. Also let $\hat{v}_i = v_{\pi_i}$, where π_i is greedy with respect to v_i. Then, given Assumptions 1 to 3, the following holds:

$$\|v^\star - \hat{v}_1\|_\infty \leq \frac{2}{1-\gamma} \min_{v \in \tilde{\mathcal{V}}} \|v - Lv\|_\infty,$$
$$\|v^\star - \hat{v}_2\|_\infty \leq \frac{2}{1-\gamma} \left(\min_{v \in \tilde{\mathcal{V}}} \|v - Lv\|_\infty + \varepsilon_p \right),$$
$$\|v^\star - \hat{v}_3\|_\infty \leq \frac{2}{1-\gamma} \left(\min_{v \in \tilde{\mathcal{V}}} \|v - Lv\|_\infty + \varepsilon_p + 2\varepsilon_s \right).$$

These bounds show that it is possible to bound policy loss due to incomplete samples. As mentioned above, existing bounds on constraint violation in approximate linear programming (de Farias and van Roy, 2004) typically do not easily lead to policy loss bounds.

Sampling guarantees for other bilinear program formulations are very similar. Because they also rely on an approximation of the initial distribution and the policy loss, they require additional assumptions on the uniformity of state samples.

Proof We show bounds on $\|v_i - Lv_i\|_\infty$; the theorem can then be inferred from Theorem 17, which establishes that ABP minimizes the Bellman residual. The first inequality follows directly from Theorem 17. The second inequality can be derived as:

$$\begin{aligned} v_2 - Lv_2 &\overset{\text{Assumption 2}}{\leq} v_2 - Lv_2 \\ &\overset{(\star)}{\leq} v_1 - Lv_1 \\ &\leq v_1 - Lv_1 + \varepsilon_p \mathbf{1} \,. \end{aligned}$$

The third inequality can be derived as:

$$\begin{aligned} v_3 - Lv_3 &\overset{\text{Assumption 2}}{\leq} v_3 - Lv_3 + \varepsilon_p \mathbf{1} \\ &\overset{\text{Assumption 3}}{\leq} v_3 - \tilde{L}v_3 + \varepsilon_s \mathbf{1} + \varepsilon_p \mathbf{1} \\ &\overset{(\star)}{\leq} v_1 - \tilde{L}v_1 + \varepsilon_s \mathbf{1} + \varepsilon_p \mathbf{1} \\ &\overset{\text{Assumption 3}}{\leq} v_1 - Lv_1 + 2\varepsilon_s \mathbf{1} + \varepsilon_p \mathbf{1} \,. \end{aligned}$$

The star $(\star)$ in the inequalities refers to the fact that $v_i \geq Lv_i$ and that v_i's minimize the corresponding Bellman residuals. ■

To summarize, this section identifies basic assumptions on the sampling behavior and shows that approximate bilinear programming scales well in the face of uncertainty caused by incomplete sampling. More detailed analysis will need to focus on identifying problem-specific assumptions and sampling modes that guarantee the basic conditions, namely satisfying Assumptions 2 and 3. Such analysis is beyond the scope of this paper.

7. Discussion and Related ADP Methods

This section describes connections between approximate bilinear programming and two closely related approximate dynamic programming methods: ALP, and L_∞-API, which are commonly used to solve factored MDPs (Guestrin et al., 2003). Our analysis sheds light on some of their observed properties and leads to a new *convergent* form of approximate policy iteration.

Approximate bilinear programming addresses some important drawbacks of ALP:

1. ALP provides value function bounds with respect to L_1 norm, which does not guarantee small policy loss;
2. ALP's solution quality depends significantly on the heuristically-chosen objective function c in (5) (de Farias, 2002);
3. The performance bounds involve a constant $1/(1-\gamma)$ which can be very large when γ is close to 1; and
4. Incomplete constraint samples in ALP easily lead to unbounded linear programs.

The downside of using approximate bilinear programming is, of course, the higher computational complexity.

The first and the second issues in ALP can be addressed by choosing a problem-specific objective function c (de Farias, 2002). Unfortunately, all existing bounds require that c is chosen based on the optimal ALP solution for c. This is impossible to compute in practice. Heuristic values for c are used instead. Robust approximate bilinear program (7), on the other hand, has no such parameters.

The fourth issue in approximate linear programs arises when the constraints need to be sampled. The ALP may become unbounded with incomplete samples because its objective value is defined using the L_1 norm on the value function, and the constraints are defined using the L_∞ norm of the Bellman residual. In approximate bilinear programs, the Bellman residual is used in both the constraints and objective function. The objective function of ABP is then bounded below by 0 for an arbitrarily small number of samples.

The NP-completeness of ABP compares unfavorably with the polynomial complexity of ALP. However, most other approximate dynamic programming algorithms are not guaranteed to converge to a solution in finite time. As we show below, the exponential time complexity of ABP is unavoidable (unless P = NP).

Proposition 35 (Mangasarian, 1995) *A bilinear program can be solved in NP time.*

The proof is straightforward. There is an optimal solution of the bilinear program such that the solutions of the individual linear programs are basic feasible. The set of all basic feasible solutions is finite, because the feasible regions of w, x and y, z are independent. The value of a basic feasible solution can be calculated in polynomial time.

The following theorem shows that the computational complexity of the ABP formulation is asymptotically the same as the complexity of tightly approximating the value function.

Theorem 36 *Suppose that* $0 < \gamma < 1$ *and* $\varepsilon > 0$. *Then the problem of determining whether the following inequalities hold is NP-complete:*

$$\min_{v \in \mathcal{K} \cap \tilde{\mathcal{V}}} \|Lv - v\|_\infty < \varepsilon \ , \qquad \min_{v \in \tilde{\mathcal{V}}} \|Lv - v\|_\infty < \varepsilon \ .$$

The problem remains NP-complete even when Assumption 1 is satisfied. In addition, it is also NP-complete to determine:

$$\min_{v \in \tilde{\mathcal{V}}} \|Lv - v\|_\infty - \|v^\star - v\|_{1,\alpha} < \varepsilon \ , \qquad \min_{v \in \tilde{\mathcal{V}}} \|Lv - v\|_{1,u} - \|v^\star - v\|_{1,\alpha} < \varepsilon \ ,$$

assuming that $u \geq \mathbf{0}$ *and* $\mathbf{1}^\mathsf{T} u = 1$.

As the theorem states, the value function approximation does not become computationally simpler even when Assumption 1 holds—a universal assumption in the paper. Notice that ALP can determine whether $\min_{v \in \mathcal{K} \cap \tilde{\mathcal{V}}} \|Lv - v\|_\infty = 0$ in polynomial time.

Proof The membership in NP follows from Theorem 17 and Theorem 35. We show NP-hardness by a reduction from the 3SAT problem. We first do not make Assumption 1. We show that the theorem holds for $\varepsilon = 1$. The appropriate ε can be obtained by simply scaling the rewards in the MDP.

The main idea is to construct an MDP and an approximation basis, such that the approximation error is small whenever the SAT problem is satisfiable. The values of the states will correspond to

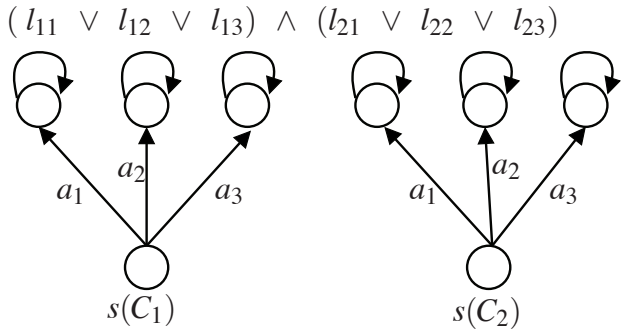


Figure 1: MDP constructed from the corresponding SAT formula.

the truth values of the literals and clauses. The approximation features ϕ will be used to constrain the values of literals that share the same variable. The MDP constructed from the SAT formula is depicted in Figure 1.

Consider a SAT problem with clauses C_i:

$$\bigwedge_{i=1,\ldots,n} C_i = \bigwedge_{i=1,\ldots,n} (l_{i1} \vee l_{i2} \vee l_{i3}) \ ,$$

where l_{ij} are literals. A literal is a variable or the negation of a variable. The variables in the SAT problem are $x_1 \ldots x_m$. The corresponding MDP is constructed as follows. It has one state $s(l_{ij})$ for every literal l_{ij}, one state $s(C_i)$ for each clause C_i and an additional state s. That is:

$$\mathcal{S} = \{s(C_i) \,|\, i = 1,\ldots,n\} \cup \{s(l_{ij}) \,|\, i = 1,\ldots,n, j = 1,\ldots,3\} \cup \{s\} \ .$$

There are 3 actions available in each state $s(C_i)$, which determine the literal of the clause whose value is true. There is only a single action available in states $s(l_{ij})$ and s. All the MDP's transitions are deterministic. The transition $t(s,a) = (s',r)$ is from the state s to s', when action a is taken, and the reward received is r. The transitions are as follows:

$$\begin{aligned} t(s(C_i), a_j) &= (s(l_{ij}), 1-\gamma) \ , \\ t(s(l_{ij}), a) &= (s(l_{ij}), -(1-\gamma)) \ , \\ t(s,a) &= (s, 2-\gamma) \ . \end{aligned}$$

Notice that the rewards depend on the discount factor γ, for notational convenience.

There is one approximation feature for every variable x_k such that:

$$\begin{aligned} \phi_k(s(C_i)) &= 0 \ , \\ \phi_k(s) &= 0 \ , \\ \phi_k(s(l_{ij})) &= \begin{cases} 1 & \text{if } l_{ij} = x_k \\ -1 & \text{if } l_{ij} = \neg x_k \end{cases} \ . \end{aligned}$$

An additional feature in the problem ϕ is defined as follows:

$$\begin{aligned} \phi(s(C_i)) &= 1 \ , \\ \phi(s(l_{ij})) &= 0 \ , \\ \phi(s) &= 1 \ . \end{aligned}$$

The purpose of state s is to ensure that $v(s(c_i)) \geq 2-\gamma$, as we assume in the remainder of the proof.

First, we show that if the SAT problem is satisfiable, then $\min_{v \in \tilde{\mathcal{V}} \cap \mathcal{K}} \|Lv - v\|_\infty < 1$. The value function $\tilde{v} \in \mathcal{K}$ is constructed as a linear sum of the features as: $v = \Phi y$, where $y = (y_1, \ldots, y_m, y)$. Here y_k corresponds to ϕ_k and y corresponds to ϕ. The coefficients y_k are constructed from the truth value of the variables as follows:

$$y_k = \begin{cases} \gamma & \text{if } x_k = \text{true} \\ -\gamma & \text{if } x_k = \text{false} \end{cases} ,$$
$$y = 2-\gamma \, .$$

Now define the *deterministic* policy π as:

$$\pi(s(C_i)) = a_j \text{ where } l_{ij} = \text{true} \, .$$

The true literals are guaranteed to exist from the satisfiability. This policy is greedy with respect to $\tilde{v}$ and satisfies that $\|L_\pi \tilde{v} - \tilde{v}\|_\infty \leq 1-\gamma^2$.

The Bellman residuals for all actions and states, given a value function v, are defined as:

$$v(s) - \gamma v(s') - r \, ,$$

where $t(s,a) = (s', r)$. Given the value function $\tilde{v}$, the residual values are:

$$t(s(C_i), a_j) = (s(l_{ij}), 1-\gamma): \quad \begin{cases} 2-\gamma-\gamma^2+(1-\gamma) = 1-\gamma^2 & \text{if} \quad l_{ij} = \text{true} \\ 2-\gamma+\gamma^2+(1-\gamma) = 1+\gamma^2 & \text{if} \quad l_{ij} = \text{false} \end{cases} ,$$
$$t(s(l_{ij}), a) = (s(l_{ij}), (1-\gamma)): \quad \begin{cases} \gamma-\gamma^2+1-\gamma = 1-\gamma^2 & \text{if } l_{ij} = \text{true} \\ -\gamma+\gamma^2+1-\gamma = (1-\gamma)^2 > 0 & \text{if } l_{ij} = \text{false} \end{cases} ,$$
$$t(s,a) = (s, 1-\gamma): \quad (1-\gamma)+\gamma-1 = 0 \, .$$

It is now clear that π is greedy and that:

$$\|L\tilde{v} - \tilde{v}\|_\infty = 1-\gamma^2 < 1 \, .$$

We now show that if the SAT problem is not satisfiable then $\min_{v \in \mathcal{K} \cap \tilde{\mathcal{V}}} \|Lv - v\|_\infty \geq 1 - \frac{\gamma^2}{2}$. Now, given a value function v, there are two possible cases for each $v(s(l_{ij}))$: 1) a positive value, and 2) a non-positive value. Two literals that share the same variable will have the same sign, since there is only one feature per each variable.

Assume now that there is a value function $\tilde{v}$. There are two possible cases we analyze: 1) all transitions of a greedy policy are to states with positive value, and 2) there is at least one transition to a state with a non-positive value. In the first case, we have that

$$\forall i \, \exists j, \; \tilde{v}(s(l_{ij})) > 0 \, .$$

That is, there is a function $q(i)$, which returns the positive literal for the clause j. Now, create a satisfiable assignment of the SAT problem as follows:

$$x_k = \begin{cases} \text{true} & \text{if } l_{iq(i)} = x_k \\ \text{false} & \text{if } l_{iq(i)} = \neg x_k \end{cases} ,$$

with other variables assigned arbitrary values. Given this assignment, all literals with states that have a positive value will be also positive. Since every clause contains at least one positive literal, the SAT is satisfiable, which is a contradiction with the assumption. Therefore, there is at least one transition to a state with a non-positive value.

Let C_1 represent the clause with a transition to a literal l_{11} with a non-positive value, without loss of generality. The Bellman residuals at the transitions from these states will be:

$$b_1 = \tilde{v}(s(l_{11})) - \gamma\tilde{v}(s(l_{11})) + (1-\gamma) \geq 0 - 0 + (1-\gamma) = 1-\gamma \ ,$$
$$b_1 = \tilde{v}(s(C_1)) - \gamma\tilde{v}(s(l_{11})) - (1-\gamma) \geq 2 - \gamma - 0 - 1 + \gamma = 1 \ .$$

Therefore, the Bellman residual $\tilde{v}$ is bounded as:

$$\|L\tilde{v} - \tilde{v}\|_\infty \geq \max\{b_1, b_2\} \geq 1 \ .$$

Since we did not make any assumptions on $\tilde{v}$, the claim holds for all representable and transitive-feasible value functions. Therefore, $\min_{v \in \tilde{\mathcal{V}} \cap \mathcal{K}} \|Lv - v\|_\infty \leq 1-\gamma^2$ is and only if the 3SAT problem is feasible.

We now show that the problem remains NP-complete even when Assumption 1 holds. Define a new state s_1 with the following transition:

$$t(s_2, a) = (s_2, -\frac{\gamma}{2}) \ .$$

All previously introduced features ϕ are zero on the new state. That is $\phi_k(s_1) = \phi(s_1) = 0$. The new constant feature is: $\hat{\phi}(s) = 1$ for all states $s \in \mathcal{S}$, and the matching coefficient is denoted as y_1. When the formula is satisfiable, then clearly $\min_{v \in \tilde{\mathcal{V}} \cap \mathcal{K}} \|Lv - v\|_\infty \leq 1-\gamma^2$ since the basis is now richer and the Bellman error on the new transition is less than $1-\gamma^2$ when $y_1 = 0$.

Now we show that when the formula is not satisfiable, then:

$$\min_{v \in \tilde{\mathcal{V}} \cap \mathcal{K}} \|Lv - v\|_\infty \geq 1 - \frac{\gamma^2}{2} \ .$$

This can be scaled to an appropriate ε by scaling the rewards. Notice that

$$0 \leq y_1 \leq \frac{\gamma}{2} \ .$$

When $y_1 < 0$, the Bellman residual on transitions $s(C_i) \to s(l_{ij})$ may be decreased by increasing y_1 while adjusting other coefficients to ensure that $v(s(C_i)) = 2-\gamma$. When $y_1 > \frac{\gamma}{2}$ then the Bellman residual from the state s_1 is greater than $1 - \frac{\gamma^2}{2}$. Given the bounds on y_1, the argument for $y_k = 0$ holds and the minimal Bellman residual is achieved when:

$$v(s(C_i)) - \gamma v(s(l_{ij})) - (1-\gamma) = v(s(s_1)) - \gamma v(s(s_1)) + \frac{\gamma}{2} \ ,$$
$$2 - \gamma - \gamma y_1 - (1-\gamma) = y_1 - \gamma y_1 + \frac{\gamma}{2} \ ,$$
$$y_1 = \frac{\gamma}{2} \ .$$

Therefore, when the SAT problem is unsatisfiable, the Bellman residual is at least $1 - \frac{\gamma^2}{2}$.

The NP-completeness of $\min_{v\in\tilde{\mathcal{V}}} \|Lv - v\|_\infty < \varepsilon$ follows trivially from the fact that transitive-feasibility does not restrict the solution quality. The proof for $\|v - Lv\|_\infty - \alpha^\mathsf{T} v$ is almost identical. The difference is a new state $\hat{s}$, such that $\phi(\hat{s}) = \mathbf{1}$ and $\alpha(\hat{s}) = 1$. In that case $\alpha^\mathsf{T} v = 1$ for all $v \in \tilde{\mathcal{V}}$. The additional term thus has no effect on the optimization.

The proof can be similarly extended to the minimization of $\|v - Lv\|_{1,u}$. Define $u(C_i) = 1/n$ and $u(l_{ij}) = 0$. Then the SAT problem is satisfiable if an only if $\|v - Lv\|_{1,u} = 1 - \gamma^2$. Note that u, as defined above, is not an upper bound on the occupancy frequencies u_π. It is likely that the proof could be extended to cover the case $u \geq u_\pi$ by more carefully designing the transitions from C_i. In particular, there needs to be high probability of returning to C_i and $u(l_{ij} > 0$. ■

Approximate bilinear programming can also improve on API with L_∞ minimization (L_∞-API for short), which is a popular method for solving factored MDPs (Guestrin et al., 2003). Minimizing the L_∞ approximation error is theoretically preferable, since it is compatible with the existing bounds on policy loss (Guestrin et al., 2003). The bounds on value function approximation in API are typically (Munos, 2003):

$$\limsup_{k\to\infty} \|v^\star - \hat{v}_k\|_\infty \leq \frac{2\gamma}{(1-\gamma)^2} \limsup_{k\to\infty} \|\tilde{v}_k - v_k\|_\infty \,,$$

where $\hat{v}_k$ is the value function of policy π_k which is greedy with respect to $\tilde{v}_k$. These bounds are looser than the bounds on solutions of ABP by at least a factor of $1/(1-\gamma)$. Often the difference may be up to $1/(1-\gamma)^2$ since the error $\|\tilde{v}_k - v_k\|_\infty$ may be significantly larger than $\|\tilde{v}_k - L\tilde{v}_k\|_\infty$. Finally, the bounds cannot be easily used, because they only hold in the limit.

We propose *Optimistic Approximate Policy Iteration* (OAPI), a modification of API. OAPI is shown in Algorithm 1, where $\mathcal{Z}(\pi)$ is calculated using the following program:

$$\begin{array}{ll} \min\limits_{\sigma,v} & \sigma \\ \text{s.t.} & Av \geq b \quad (\equiv (\mathbf{I} - \gamma P_a)v \geq r_a \;\; \forall a \in \mathcal{A}) \\ & -(\mathbf{I} - \gamma P_\pi)v + \mathbf{1}\sigma \geq -r_\pi \,, \\ & v \in \tilde{\mathcal{V}} \,. \end{array} \tag{15}$$

In fact, OAPI corresponds to Algorithm 2 applied to ABP because the linear program (15) corresponds to (7) with a fixed π. Then, using Theorem 26, we get the following corollary.

Corollary 37 *Optimistic approximate policy iteration converges in finite time. In addition, the Bellman residual of the generated value functions monotonically decreases.*

OAPI differs from L_∞-API in two ways: 1) OAPI constrains the Bellman residuals by 0 from below and by σ from above, and then it minimizes σ. L_∞-API constrains the Bellman residuals by σ from both above and below. 2) OAPI, like API, uses only the current policy for the upper bound on the Bellman residual, but uses *all* the policies for the lower bound on the Bellman residual. Next we show that the optimal solutions of (16) and (17) are closely related.

L_∞-API cannot return an approximate value function that has a lower Bellman residual than ABP, given the optimality of ABP described in Theorem 17. However, even OAPI—an approximate ABP algorithm—is guaranteed to perform comparably to L_∞-API, as the following theorem states.

Theorem 38 *Assume that L_∞-API converges to a policy π and a value function v. Then, define:*

$$v' = v + \frac{1}{1-\gamma}\|v - L_\pi v\|_\infty \mathbf{1} .$$

The pair π and v' is a fixed point of OAPI when ties are broken appropriately.

Notice that while the optimistic and standard policy iterations can converge to the same solutions, the steps in their computation may not be identical. In addition, there may be multiple points of convergence with the solution depending on the initialization.

Proof First, note that the value function optimization in API and OAPI corresponds to the following optimization problems:

$$\min_{v\in\tilde{\mathcal{V}}} \|L_\pi v - v\|_\infty = \min_{\sigma,v} \left\{ \sigma \;\middle|\; \begin{array}{c} (\mathbf{I}-\gamma P_\pi)v + \mathbf{1}\sigma \geq r_\pi \\ -(\mathbf{I}-\gamma P_\pi)v + \mathbf{1}\sigma \geq -r_\pi \end{array}, v \in \tilde{\mathcal{V}} \right\} , \tag{16}$$

$$\min_{v\in\tilde{\mathcal{V}}\cap\mathcal{K}} \|L_\pi v - v\|_\infty = \min_{\sigma,v} \left\{ \sigma \;\middle|\; \begin{array}{c} (\mathbf{I}-\gamma P_a)v \geq r_a \quad \forall a \in \mathcal{A} \\ -(\mathbf{I}-\gamma P_\pi)v + \mathbf{1}\sigma \geq -r_\pi \end{array}, v \in \tilde{\mathcal{V}} \right\} . \tag{17}$$

Given that π is greedy with respect to v and that v minimizes the Bellman residual of π, the following equalities hold:

$$\begin{aligned} L_\pi v &\geq Lv , \\ \|v - L_\pi v\|_\infty &\leq \|v - L_\pi v\|_\infty \quad \forall v \in \tilde{\mathcal{V}} , \\ -\min_{s\in\mathcal{S}}(v - L_\pi v)(s) &= \max_{s\in\mathcal{S}}(v - L_\pi v)(s) . \end{aligned}$$

Then, $v' \in \mathcal{K}$ from the first and third properties, since $v' \geq L_\pi v' \geq Lv'$. The value function v' is therefore feasible in OAPI. In addition, we have that $\|v' - L_\pi v'\|_\infty = 2\|v - L_\pi v\|_\infty$.

For the policy π to be a fixed point in OAPI, it needs to minimize the Bellman residual with respect to v'. This is easy to show as follows:

$$\begin{aligned} L_\pi v &\geq L_\pi v , \\ v - L_\pi v &\leq v - L_\pi v , \\ \mathbf{0} \leq v' - L_\pi v' &\leq v' - L_\pi v' , \\ \|v' - L_\pi v'\|_\infty &\leq \|v' - L_\pi v'\|_\infty . \end{aligned}$$

For the value function v' to be a fixed point in OAPI, it needs to minimize the Bellman residual with respect to all representable and transitive-feasible value functions. To show a contradiction, assume that there exists $v' \in \tilde{\mathcal{V}} \cap \mathcal{K}$ such that for some $\varepsilon > 0$:

$$\|v' - L_\pi v'\|_\infty \leq \|v' - L_\pi v'\|_\infty - \varepsilon .$$

Define also a value function z as follows:

$$z = v' - \left(\frac{1}{2(1-\gamma)}\|v - L_\pi v\|_\infty + \frac{\varepsilon}{2} \right) \mathbf{1} .$$

We now show that the Bellman residual of z is less than that of v:

$$\begin{aligned}
& \|v' - L_\pi v'\|_\infty && \leq \|v' - L_\pi v'\|_\infty - \varepsilon \,, \\
0 \leq \quad & \max_{s\in\mathcal{S}}(v' - L_\pi v')(s) && \leq \max_{s\in\mathcal{S}}(v' - L_\pi v')(s) - \varepsilon \,, \\
-\|v - L_\pi v\|_\infty + \frac{\varepsilon}{2} \leq \quad & \max_{s\in\mathcal{S}}(v' - L_\pi v')(s) - \|v - L_\pi v\|_\infty + \tfrac{\varepsilon}{2} && \leq -\|v - L_\pi v\|_\infty + \frac{\varepsilon}{2} \,, \\
-\|v - L_\pi v\|_\infty + \frac{\varepsilon}{2} \leq \quad & \|z - L_\pi z\|_\infty && \leq -\|v - L_\pi v\|_\infty + \frac{\varepsilon}{2} \,.
\end{aligned}$$

Therefore, $\|z - L_\pi z\|_\infty < \|v - L_\pi v\|_\infty$, which is a contradiction. ■

To summarize, OAPI guarantees convergence, while matching the performance of L_∞-API. The convergence of OAPI is achieved because given a non-negative Bellman residual, the greedy policy also minimizes the Bellman residual. Because OAPI ensures that the Bellman residual is always non-negative, it can progressively reduce it. In comparison, the greedy policy in L_∞-API does not minimize the Bellman residual, and therefore L_∞-API does not always reduce it. Theorem 38 also explains why API provides better solutions than ALP, as observed in Guestrin et al. (2003). From the discussion above, ALP can be seen as an L_1-norm approximation of a single iteration of OAPI. L_∞-API, on the other hand, performs many such ALP-like iterations.

8. Experimental Results

In this section, we validate the approach by applying it to simple reinforcement learning benchmark problems. We consider three different problem domains, each designed to empirically test a different property of the algorithm.

First, in Section 8.1, we compare the policy loss of various approximate bilinear programming formulations with the policy loss of approximate policy iteration and approximate linear programming. These experiments are on a problem that is sufficiently small to compute the optimal value function. Second, in Section 8.2, we compare the solution quality in terms of the Bellman residual for a number of applicable algorithms. Finally, in Section 8.3 we apply ABP with L_1 relaxation to a common inverted pendulum benchmark problem and solve it using the proposed mixed integer linear formulation.

Note that our analysis shows that the solution of ABP using OAPI corresponds to the solutions of API. The optimal solutions of ABP are, therefore, also at least equivalently good in terms of the Bellman residual bounds. However, the actual empirical performance of these methods will depend significantly on the specific problem; our experimental results mostly demonstrate that the proposed methods compute value functions that minimize Bellman residual bounds and result in good policies.

ABP is an off-policy approximation method like LSPI (Lagoudakis and Parr, 2003) or ALP. Thus samples can be gathered independently of the control policy. But it is necessary that multiple actions are sampled for each state to enable the selection of different policies.

8.1 Simple Chain Problem

First, we demonstrate and analyze the properties of ABP on a simple chain problem with 200 states, in which the transitions move to the right or left (2 actions) by one step with a centered Gaussian

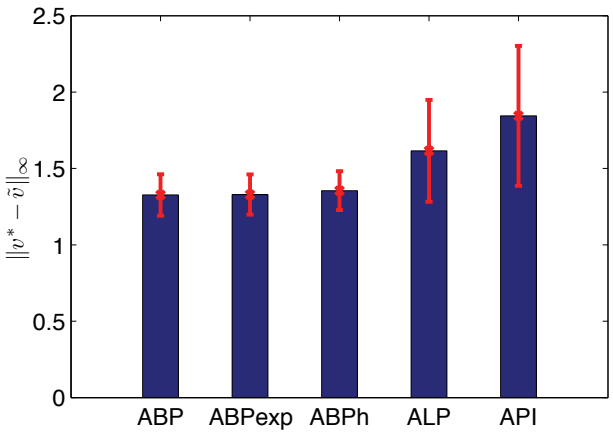


Figure 2: L_∞ Bellman residual for the chain problem

noise of standard deviation 3. The rewards were set to $\sin(i/20)$ for the right action and $\cos(i/20)$ for the left action, where i is the index of the state. This problem is small enough to calculate the optimal value function and to control the approximation features. The approximation basis in this problem is represented by piece-wise linear features, of the form $\phi(s_i) = [i-c]_+$, for c from 1 to 200. The discount factor in the experiments was $\gamma = 0.95$ and the initial distribution was $\alpha(130) = 1$. We verified that the solutions of the bilinear programs were always close to optimal, albeit suboptimal.

We experimented with the full state-action sample and randomly chose the features. All results are averages over 50 runs with 15 features. In the results, we use ABP to denote a close-to-optimal solution of robust ABP, ABPexp for the bilinear program (10), and ABPh for a formulation that minimizes the average of ABP and ABPexp. API denotes approximate policy iteration that minimizes the L_2 norm.

Figure 2 shows the Bellman residual attained by the methods. It clearly shows that the robust bilinear formulation most reliably minimizes the Bellman residual. The other two bilinear formulations are not much worse. Notice also the higher standard deviation of ALP and API. Figure 3 shows the expected policy loss, as specified in Theorem 9, for the calculated value functions. It confirms that the ABP formulation outperforms the robust formulation, since its explicit objective is to minimize the expected loss. Similarly, Figure 4 shows the robust policy loss. As expected, it confirms the better performance of the robust ABP formulation in this case.

Note that API and ALP may achieve lower policy loss on this particular domain than the ABP formulations, even though their Bellman residual is significantly higher. This is possible because ABP simply minimizes bounds on the policy loss. The analysis of tightness of policy loss bounds is beyond the scope of this paper.

8.2 Mountain Car Benchmark Problem

In the mountain-car benchmark, an underpowered car needs to climb a hill (Sutton and Barto, 1998). To do so, it first needs to back up to an opposite hill to gain sufficient momentum. The car receives a reward of 1 when it climbs the hill. The discount factor in the experiments was $\gamma = 0.99$.

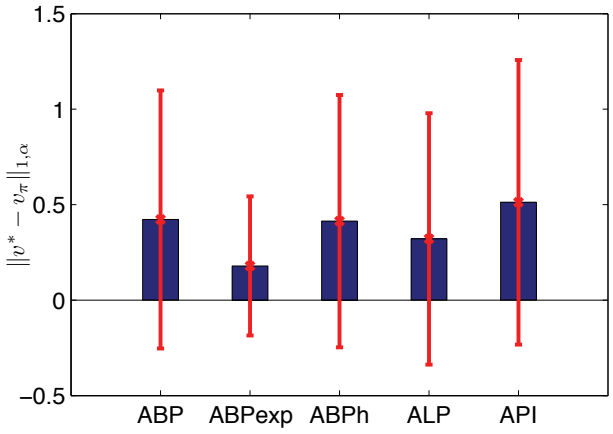


Figure 3: Expected policy loss for the chain problem

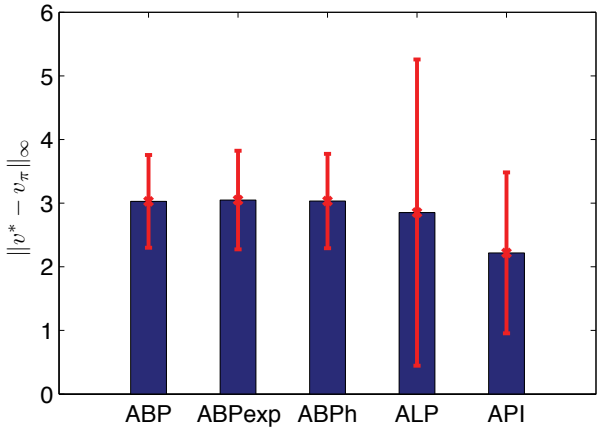


Figure 4: Robust policy loss for the chain problem

Note that the state space in this problem is infinite. It is, therefore, necessary to sample states. The states are sampled uniformly from the feasible state space and the ABP formulation is created as described in Section 6.

The experiments are designed to determine whether OAPI reliably minimizes the Bellman residual in comparison with API and ALP. We use a uniformly-spaced linear spline to approximate the value function. The constraints were based on 200 uniformly sampled states with all 3 actions per state. We evaluated the methods with 100 and 144 approximation features, which correspond to the number of linear segments.

The results of robust ABP (in particular OAPI), ALP, API with L_2 minimization, and LSPI are depicted in Table 1. The results are shown for both L_∞ norm and uniformly-weighted L_2 norm. The run-times of all these methods are comparable, with ALP being the fastest. Since API (LSPI) is not guaranteed to converge, we ran it for at most 20 iterations, which was an upper bound on the number of iterations of OAPI. The results demonstrate that ABP minimizes the L_∞ Bellman residual

(a) L_∞ error of the Bellman residual

Features	100	144
OAPI	0.21 (0.23)	0.13 (0.1)
ALP	13. (13.)	3.6 (4.3)
LSPI	9. (14.)	3.9 (7.7)
API	0.46 (0.08)	0.86 (1.18)

(b) L_2 error of the Bellman residual

Features	100	144
OAPI	0.2 (0.3)	0.1 (1.9)
ALP	9.5 (18.)	0.3 (0.4)
LSPI	1.2 (1.5)	0.9 (0.1)
API	0.04 (0.01)	0.08 (0.08)

Table 1: Bellman residual of the final value function. The values are averages over 5 executions, with the standard deviations shown in parentheses.

much more consistently than the other methods. Note, however, that all the considered algorithms would have performed significantly better with a finer approximation.

8.3 Inverted Pendulum Benchmark Problem

The goal in the inverted pendulum benchmark problem is to balance an inverted pole by accelerating a cart in either of two directions (Wang et al., 1996; Lagoudakis and Parr, 2003). There are three actions in this domain that represent applying the force of $u = -50N$, $u = 0N$, and $u = 50N$ to the cart with a uniform noise between $-10N$ and $10N$. The angle of the inverted pendulum is θ and its update equation is:

$$\ddot{\theta} = \frac{g\sin(\theta) - \alpha ml(\dot{\theta})^2\sin(2\theta)/2 - \alpha\cos(\theta)u}{4l/3 - \alpha ml\cos^2(\theta)} .$$

Here the constants are: $g = 9.8$, $m = 2.0$, $M = 8.0$, $\alpha = 1/(m+M)$. The simulation step is set to 0.1 and we use linear interpolation for simplicity.

We used the standard features for this benchmark problem; a set of radial basis functions arranged in a grid over the 2-dimensional state space with centers μ_i and a constant term required by Assumption 1. The features for a state $s = (\theta, \dot{\theta})$ are defined as:

$$\left(1, \exp -\frac{\|s-\mu_1\|_2^2}{2}, \exp -\frac{\|s-\mu_2\|_2^2}{2}, \dots\right) .$$

We considered 100 centers for radial basis functions arranged in a 10 by 10 grid for $\theta \in [-\pi/2, \pi/2]$ and $\dot{\theta} \in [-5, 5]$.

We used L_1 norm regularization to apply the sampling bounds and to compare the approach with regularized approximate linear programming. Assuming that ϕ_0 represents the constant feature, the set of representable value functions is defined as:

$$\tilde{\mathcal{V}} = \left\{ \sum_{i=0}^{100} \phi_i x_i \;\middle|\; \sum_{i=1}^{100} |x_i| \le \psi \right\} .$$

Note that the constant feature is not included in the regularization. The regularization bound was set apriori to $\psi = 100$. Subsequent tests showed that ABP performed almost identically with the regularization bound for values $\psi \in [50, 200]$.

Transition samples were collected in advance—using the same procedure as LSPI—from random episodes, starting in randomly perturbed states very close to the equilibrium state $(0, 0)$ and

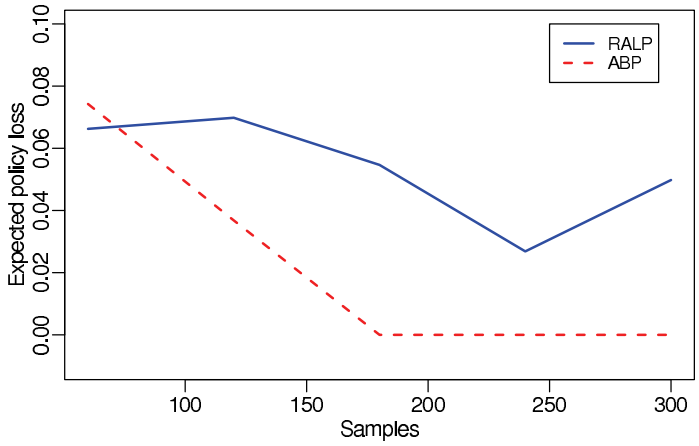


Figure 5: Policy loss as a function of the number of samples.

following a random policy. The average length of such episodes was about 6 steps. We computed the transitions for each sampled state and all the actions by sampling each transition 20 times.

We compare the solution quality to regularized approximate linear programming (RALP), which has been show to perform well on a range of benchmarks (Petrik et al., 2010). We evaluated only the formulation that minimizes the robust objective. The mixed integer linear program formulation for ABP was optimized using CPLEX 12.1. We set the time cutoff to be 60s. In this time interval, most solutions were computed to about 10% optimality gap.

Figure 5 compares the expected policy loss of ABP and RALP on the inverted pendulum benchmark as a function of the number of state transitions sampled. In every iteration, both ABP and RALP were run with the same samples. The policy loss was evaluated on 50 episodes, each at most 50 steps long. The performance of the optimal policy was assumed to be 0 and the policy loss of 0 essentially corresponds to balancing the pole for 2500 steps.

The experimental results on the inverted pendulum demonstrate that ABP may significantly outperform RALP. Both RALP and ABP have a large sampling error when the number of samples is small. This could be addressed by appropriately setting the regularization bound as our sampling bounds indicate; we kept the regularization bound fixed for all sample counts for the sake of simplicity. With a larger number of samples, ABP significantly outperforms RALP, which significantly outperforms LSPI for similar features (Petrik et al., 2010).

9. Conclusion and Future Work

We propose and analyze approximate bilinear programming, a new value-function approximation method, which provably minimizes bounds on policy loss. ABP returns the *optimal* approximate value function with respect to the Bellman residual bounds, despite being formulated with regard to transitive-feasible value functions. We also show that there is no asymptotically simpler formulation, since finding the closest value function and solving a bilinear program are both NP-complete problems. Finally, the formulation leads to the development of OAPI, a new convergent form of API which monotonically improves the objective value function.

While we only discuss simple solvers for ABP, a deeper study of bilinear solvers may lead to more efficient optimal solution methods. ABPs have a small number of essential variables (that

determine the value function) and a large number of constraints, which can be leveraged by some solvers (Petrik and Zilberstein, 2007). In addition, the L_∞ error bound provides good theoretical guarantees, but it may be too conservative in practice; a similar formulation based on L_2 norm minimization may be more practical.

Note that, as for example LSPI, approximate bilinear programming is especially applicable to MDPs with discrete (and small) action spaces. This requirement is limiting in solving many resource management problems in which the resource is a continuous variable. While it is always possible to discretize the action space, this is not feasible when the action space is multidimensional. Therefore, extending these methods to problems with continuous action spaces is an important issue that needs to be addressed in future work.

We believe that the proposed formulation will help deepen the understanding of value function approximation and the characteristics of existing solution methods, and potentially lead to the development of more robust and more widely-applicable reinforcement learning algorithms.

Acknowledgments

This work was supported by the Air Force Office of Scientific Research under Grant No. FA9550-08-1-0171. We also thank the anonymous reviewers for their comments that helped to improve the paper significantly.

References

Pieter Abbeel, Varun Ganapathi, and Andrew Y. Ng. Learning vehicular dynamics, with application to modeling helicopters. In *Advances in Neural Information Processing Systems*, pages 1–8, 2006.

Daniel Adelman. A price-directed approach to stochastic inventory/routing. *Operations Research*, 52:499–514, 2004.

Richard Bellman. *Dynamic Programming*. Princeton University Press, 1957.

Kristin P. Bennett and O. L. Mangasarian. Bilinear separation of two sets in n-space. *Computation Optimization and Applications*, 2, 1993.

Dimitri P. Bertsekas and Sergey Ioffe. Temporal differences-based policy iteration and applications in neuro-dynamic programming. Technical Report LIDS-P-2349, LIDS, 1997.

Dimitri P. Bertsekas and John N. Tsitsiklis. *Neuro-Dynamic Programming*. Athena Scientific, 1996.

Alberto Carpara and Michele Monaci. Bidimensional packing by bilinear programming. *Mathematical Programming Series A*, 118:75–108, 2009.

Daniela P. de Farias. *The Linear Programming Approach to Approximate Dynamic Programming: Theory and Application*. PhD thesis, Stanford University, 2002.

Daniela P. de Farias and Ben van Roy. The linear programming approach to approximate dynamic programming. *Operations Research*, 51:850–856, 2003.

Daniela P. de Farias and Benjamin van Roy. On constraint sampling in the linear programming approach to approximate dynamic programming. *Mathematics of Operations Research*, 29(3): 462–478, 2004.

Carlos Guestrin, Daphne Koller, Ronald Parr, and Shobha Venkataraman. Efficient solution algorithms for factored MDPs. *Journal of Artificial Intelligence Research*, 19:399–468, 2003.

Reiner Horst and Hoang Tuy. *Global optimization: Deterministic approaches*. Springer, 1996.

Michail G. Lagoudakis and Ronald Parr. Least-squares policy iteration. *Journal of Machine Learning Research*, 4:1107–1149, 2003.

Olvi L. Mangasarian. The linear complementarity problem as a separable bilinear program. *Journal of Global Optimization*, 12:1–7, 1995.

Remi Munos. Error bounds for approximate policy iteration. In *International Conference on Machine Learning*, pages 560–567, 2003.

Marek Petrik. *Optimization-based Approximate Dynamic Programming*. PhD thesis, University of Massachusetts Amherst, 2010.

Marek Petrik and Shlomo Zilberstein. Anytime coordination using separable bilinear programs. In *Conference on Artificial Intelligence*, pages 750–755, 2007.

Marek Petrik and Shlomo Zilberstein. Constraint relaxation in approximate linear programs. In *International Conference on Machine Learning*, pages 809–816, 2009.

Marek Petrik, Gavin Taylor, Ron Parr, and Shlomo Zilberstein. Feature selection using regularization in approximate linear programs for Markov decision processes. In *International Conference on Machine Learning*, pages 871–878, 2010.

Warren B. Powell. *Approximate Dynamic Programming*. Wiley-Interscience, 2007.

Martin L. Puterman. *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. John Wiley & Sons, Inc., 2005.

Kenneth O. Stanley and Risto Miikkulainen. Competitive coevolution through evolutionary complexification. *Journal of Artificial Intelligence Research*, 21:63–100, 2004.

Richard S. Sutton and Andrew Barto. *Reinforcement Learning*. MIT Press, 1998.

Istvan Szita and Andras Lorincz. Learning Tetris using the noisy cross-entropy method. *Neural Computation*, 18(12):2936–2941, 2006.

Hua O. Wang, Kazuo Tanaka, and Meichael F. Griffin. An approach to fuzzy control of nonlinear systems: Stability and design issues. *IEEE Transactions on Fuzzy Systems*, 4:14–23, 1996.

Ronald J. Williams and Leemon C. Baird. Tight performance bounds on greedy policies based on imperfect value functions. In *Yale Workshop on Adaptive and Learning Systems*, 1994.

Journal of Machine Learning Research 12 (2011) 3065-3069 Submitted 10/10; Revised 8/11; Published 10/11

The Stationary Subspace Analysis Toolbox

Jan Saputra Müller SAPUTRA@CS.TU-BERLIN.DE
Paul von Bünau PAUL.BUENAU@TU-BERLIN.DE
Frank C. Meinecke FRANK.MEINECKE@TU-BERLIN.DE
Franz J. Király FRANZ.J.KIRALY@TU-BERLIN.DE
Klaus-Robert Müller KLAUS-ROBERT.MUELLER@TU-BERLIN.DE
Machine Learning Group
Department of Computer Science
Berlin Institute of Technology (TU Berlin)
Franklinstr. 28/29, 10587 Berlin, Germany

Editor: Cheng Soon Ong

Abstract

The Stationary Subspace Analysis (SSA) algorithm linearly factorizes a high-dimensional time series into stationary and non-stationary components. The SSA Toolbox is a platform-independent efficient stand-alone implementation of the SSA algorithm with a graphical user interface written in Java, that can also be invoked from the command line and from Matlab. The graphical interface guides the user through the whole process; data can be imported and exported from comma separated values (CSV) and Matlab's `.mat` files.

Keywords: non-stationarities, blind source separation, dimensionality reduction, unsupervised learning

1. Introduction

Discovering and understanding temporal changes in high-dimensional time series is a central task in data analysis. In particular, when the observed data is a mixture of latent factors that cannot be measured directly, visual inspection of multivariate time series is not informative to discern stationary and non-stationary contributions. For example, a single non-stationary factor can be spread out among all channels and make the whole data appear non-stationary, even when all other sources are perfectly stationary. Conversely, a non-stationary component with low power can remain hidden among stronger stationary sources. In electroencephalography (EEG) analysis (Niedermeyer and Lopes da Silva, 2005), for instance, the electrodes on the scalp record a mixture of the activity from a multitude of sources located inside the brain, which we cannot measure individually with non-invasive methods. Thus, in order to distinguish the activity of stationary and non-stationary brain sources, we need to separate their contributions in the measured EEG signals (von Bünau et al., 2010).

To that end, in the Stationary Subspace Analysis (SSA) model (von Bünau et al., 2009), the observed data $x(t) \in \mathbb{R}^D$ is assumed to be generated as a linear mixture of d stationary sources $\mathbf{s}^{\mathrm{s}}(t)$ and $D-d$ non-stationary sources $\mathbf{s}^{\mathrm{n}}(t)$,

$$x(t) = As(t) = \begin{bmatrix} A^{\mathrm{s}} & A^{\mathrm{n}} \end{bmatrix} \begin{bmatrix} s^{\mathrm{s}}(t) \\ s^{\mathrm{n}}(t) \end{bmatrix},$$

where A is an invertible mixing matrix. Note that the sources $s(t)$ are *not* assumed to be independent or uncorrelated. A time series is considered stationary if its mean and covariance are constant over time, that is, a time series $u(t)$ is called stationary if

$$\mathbb{E}[u(t_1)] = \mathbb{E}[u(t_2)] \text{ and } \mathbb{E}[u(t_1)u(t_1)^\top] = \mathbb{E}[u(t_2)u(t_2)^\top],$$

at all pairs of time points $t_1, t_2 \geq 0$. This is a variant of *weak stationarity* (Priestley, 1983), where we do not consider the time structure.

The SSA algorithm (von Bünau et al., 2009; Hara et al., 2010; Kawanabe et al., 2011) finds the demixing matrix that separates the stationary and non-stationary sources given samples from $x(t)$ by solving a non-convex optimization problem. This yields an estimate for the mixing matrix, and the stationary and non-stationary sources.

2. Capabilities of the SSA Toolbox

The SSA Toolbox is a platform-independent implementation of the SSA algorithm with a convenient graphical user interface. The latest release is available from the SSA website.[1] It can be used in the following environments.

- As a *stand-alone application* with a graphical user interface.
- From the operating system's *command line*.
- From *Matlab* via an efficient in-memory interface through the wrapper script `ssa.m`.
- As a *library* from your Java own application.

In the following, we give an overview of the main features of the SSA Toolbox.

2.1 Platforms

The SSA Toolbox is platform-independent: it is written in the Java programming language with bytecode backwards-compatible until JVM version 1.5 (released in 2004); native libraries are included for all major platforms with a pure-Java fallback.

2.2 Data Import/Export

The stand-alone application can read data and write results from comma separated values (CSV) and from Matlab's `.mat` file format.[2]

2.3 Efficiency

The efficiency of the toolbox is mainly due to the underlying matrix libraries. The user can choose between `COLT`,[3] written in pure Java, and the high-performance library jblas[4] (Braun et al., 2010), which wraps the state-of-the-art BLAS and LAPACK implementations included as native binaries for Windows, Linux and MacOS in 32 and 64 bit.

1. See `http://www.stationary-subspace-analysis.org/toolbox`.
2. We use the `JMatIO` library, see `http://sourceforge.net/projects/jmatio`.
3. See `http://acs.lbl.gov/software/colt/`.
4. See `http://www.jblas.org`.

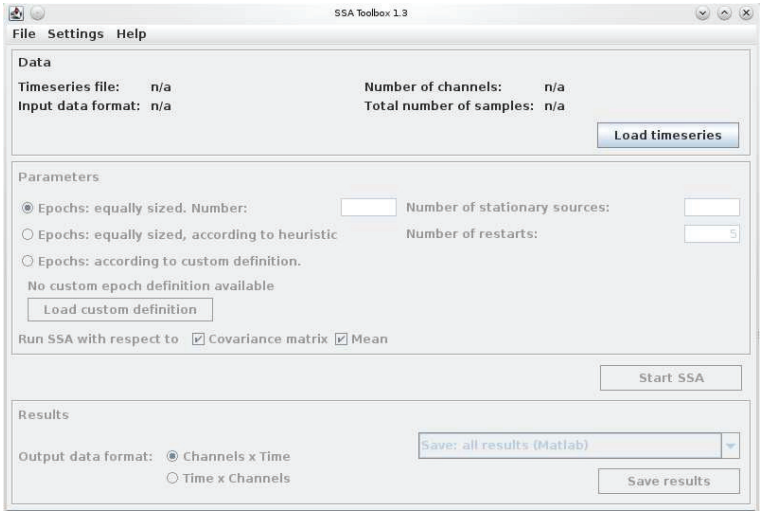


Figure 1: Graphical user interface of the SSA Toolbox. From top to bottom, the panels correspond to the steps data import, parameter specification, and export of results. The window also includes a log panel at the bottom, which is not shown here.

2.4 User Interface

The graphical user interface of the stand-alone application provides step-by-step guidance through the whole process: from data import, specification of parameters to the export of results. The toolbox also suggests sensible parameter values based on heuristics. The log panel, not pictured in Figure 1, shows instructive error and diagnostic messages.

```
>> [ X, A ] = ssa_toydata(10, 2, 2);    % generate data
>> [ Ps, Pn, As, An ] = ssa(X, 2);      % apply SSA
>> err=subspace_error(An, A(:,[3 4]));  % measure the error
>> s1=Ps*X{1};     % Project to s-sources in epoch 1
```

Figure 2: Application of the SSA Toolbox from within Matlab to a synthetic data set with two stationary and two non-stationary sources.

2.5 Matlab Interface

The implementation of the SSA algorithm can also be accessed directly from Matlab, using the wrapper script `ssa.m`, see Figure 2. Data and results are passed in-memory between Java and Matlab and all messages are relayed to the Matlab prompt.

2.6 Documentation

The user manual explains the SSA algorithm, the use of the toolbox, interpretation of results and answers frequently asked questions. It also includes a section for developers that provides an overview of the source code and a description of the unit tests.

2.7 Examples

The toolbox comes with example data in CSV and `.mat` format, a Matlab script for generating synthetic data sets (documented in the manual, and a self-contained Matlab demo `ssa_demo.m`).

2.8 Developer Access, License and Unit Tests

The source code is provided under the BSD license and is available in a separate archive for each released version. The latest version of the source code is available from github,[5] a free hosting services for the git version control system. The source code is fully documented according to the Javadoc conventions and accompanied by a set of unit tests, which are described in the developer section of the user manual.

Acknowledgments

We acknowledge support by the Bernstein Cooperation (German Federal Ministry of Education and Science), FKZ 01 GQ 0711; the Bernstein Focus Neurotechnology and the EU PASCAL2 NoE.

References

Mikio Braun, Johannes Schaback, Jan Saputra Mueller, and Matthias Jugel. jblas, 2010. `http://mloss.org/software/view/180/`.

Satoshi Hara, Yoshinobu Kawahara, Takashi Washio, and Paul von Bünau. Stationary subspace analysis as a generalized eigenvalue problem. In *Proceedings of the 17th International Conference on Neural Information Processing: Theory and Algorithms - Volume Part I*, ICONIP'10, pages 422–429, Berlin, Heidelberg, 2010. Springer-Verlag.

Motoaki Kawanabe, Wojciech Samek, Paul von Bünau, and Frank Meinecke. An information geometrical view of stationary subspace analysis. In *Artificial Neural Networks and Machine Learning ICANN 2011*, volume 6792 of *Lecture Notes in Computer Science*, pages 397–404. Springer Berlin / Heidelberg, 2011. URL `http://dx.doi.org/10.1007/978-3-642-21738-8_51`.

Ernst Niedermeyer and Feranando H. Lopes da Silva. *Electroencephalography: Basic Principles, Clinical Applications, and Related Fields*. Lippincott, Williams and Wilkins, 530 Walnut Street, Philadephia, PA 19106 USA, 2005.

Maurice B. Priestley. *Spectral Analysis and Time Series*. Academic Press, 1983.

5. See `http://github.org`.

Paul von Bünau, Frank C. Meinecke, Franz J. Király, and Klaus-Robert Müller. Finding stationary subspaces in multivariate time series. *Phys. Rev. Lett.*, 103(21):214101, Nov 2009. doi: 10.1103/PhysRevLett.103.214101.

Paul von Bünau, Frank C. Meinecke, Simon Scholler, and Klaus-Robert Müller. Finding stationary brain sources in EEG data. In *Proceedings of the 32nd Annual Conference of the IEEE EMBS*, pages 2810–2813, 2010.

Journal of Machine Learning Research 12 (2011) 3071-3096 Submitted 10/10; Revised 7/11; Published 11/11

In All Likelihood, Deep Belief Is Not Enough

Lucas Theis LUCAS.THEIS@TUEBINGEN.MPG.DE
Sebastian Gerwinn SEBASTIAN.GERWINN@TUEBINGEN.MPG.DE
Fabian Sinz FABIAN.SINZ@TUEBINGEN.MPG.DE
Matthias Bethge MATTHIAS.BETHGE@TUEBINGEN.MPG.DE
Werner Reichardt Centre for Integrative Neuroscience
Bernstein Center for Computational Neuroscience
Max Planck Institute for Biological Cybernetics
Spemannstraße 41, 72076 Tübingen, Germany

Editor: Yoshua Bengio

Abstract

Statistical models of natural images provide an important tool for researchers in the fields of machine learning and computational neuroscience. The canonical measure to quantitatively assess and compare the performance of statistical models is given by the likelihood. One class of statistical models which has recently gained increasing popularity and has been applied to a variety of complex data is formed by deep belief networks. Analyses of these models, however, have often been limited to qualitative analyses based on samples due to the computationally intractable nature of their likelihood. Motivated by these circumstances, the present article introduces a consistent estimator for the likelihood of deep belief networks which is computationally tractable and simple to apply in practice. Using this estimator, we quantitatively investigate a deep belief network for natural image patches and compare its performance to the performance of other models for natural image patches. We find that the deep belief network is outperformed with respect to the likelihood even by very simple mixture models.

Keywords: deep belief network, restricted Boltzmann machine, likelihood estimation, natural image statistics, potential log-likelihood

1. Introduction

When dealing with natural images, the choice of image representation is often crucial for achieving a good performance. Good generative models or classifiers, for example, can be easier to realize in terms of more complex features such as edges than in terms of raw pixel intensities. Several facts point to the advantage of using hierarchical representations for encoding natural images over non-hierarchical ones. Multiple layers of representations allow for the use of simpler transformations, each solving only a subproblem, as well as for a more efficient implementation by enabling the reuse of low-level features in the realization of higher-level features. Further motivation for hierarchical representations comes from the hierarchical organization of the brain (Felleman and van Essen, 1991) and the hierarchical organization of the concepts surrounding us. The idea of using hierarchical image representations in supervised as well as unsupervised tasks is now several decades old (e.g., Selfridge, 1958; Fukushima, 1980; LeCun et al., 1989). However, only the emergence of recent training methods has made them competitive across many supervised learning tasks and led to a renewed surge of interest in hierarchical image representations. Despite this success

in supervised tasks, no probabilistic model has been conclusively shown to exhibit state-of-the-art performance as a *generative* model and at the same time benefit strongly from hierarchical image representations.

The most prominent example of the recent research in hierarchical image modeling is the research into a class of hierarchical generative models called *deep belief networks*. Deep belief networks were introduced by Hinton and Salakhutdinov (2006); Hinton et al. (2006) together with a greedy learning rule as an approach to the long-standing challenge of training *deep* neural networks, that is, hierarchical neural networks such as multi-layer perceptrons. The existence of an efficient learning rule has made them become attractive not only for pretraining multi-layer perceptrons, but also for density estimation and other inherently unsupervised learning tasks. In supervised tasks, they have been shown to learn representations which outperform many competing representations when employed, for example, in character recognition (Hinton et al., 2006) or speech recognition (Mohamed et al., 2009). In unsupervised tasks, they have been applied to a wide variety of complex data sets such as patches of natural images (Osindero and Hinton, 2008; Ranzato et al., 2010a; Ranzato and Hinton, 2010; Lee and Ng, 2007), motion capture recordings (Taylor et al., 2007) and images of faces (Susskind et al., 2008). When applied to natural images, deep belief networks have been shown to develop biologically plausible features (Lee and Ng, 2007) and samples from the model were shown to adhere to certain statistical regularities also found in natural images (Osindero and Hinton, 2008). Examples of natural image patches and features learned by a deep belief network are presented in Figure 1.

An important measure to assess the generative performance of a probabilistic model is the likelihood. The likelihood allows us to objectively compare the density estimation performance of different models. Given two model instances with equal a priori probability, the ratio of their likelihoods with respect to a set of data samples tells us everything we need to know to decide which of the two models is more likely to have generated the data set. Further motivation for the likelihood stems from coding theory. For densities p and q, the negative expected log-likelihood represents the *cross-entropy* term of the Kullback-Leibler (KL) divergence,

$$D_{\mathrm{KL}}[p(x)||q(x)] = -\sum_x p(x)\log q(x) - \mathbf{H}[p(x)],$$

which is always non-negative and zero if and only if p and q are identical. The cross-entropy represents the coding cost of encoding samples drawn from p with a code that would be optimal for samples drawn from q. Correspondingly, the KL-divergence represents the additional coding cost created by using an optimal code which assumes the distribution of the samples to be q instead of p.

The expected negative log-likelihood, or cross-entropy, quantifies the amount of correlations captured by a statistical model. A model with a minimal cross-entropy would, at least in principle, be able to predict missing information from partially observed input in an optimal manner. In this sense, the likelihood can be understood as a measure of scene understanding if a model is applied to natural scenes.

Finally, the likelihood allows us to directly examine the success of training when maximum likelihood learning is employed. Even when the ultimate goal is classification, deep belief networks and related unsupervised feature learning approaches are optimized with respect to the likelihood. Evaluating the likelihood is therefore also important to assess the success of pretraining and for fine-tuning hyperparameters. Unfortunately, the likelihood of deep belief networks is in general computationally intractable to evaluate.

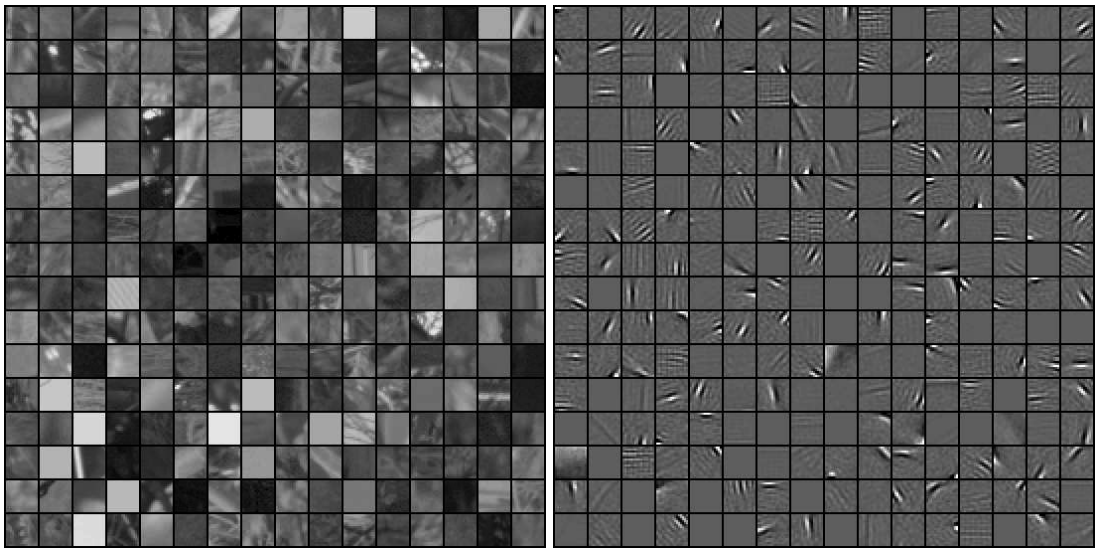


Figure 1: *Left:* Natural image patches sampled from the van Hateren dataset (van Hateren and van der Schaaf, 1998). *Right:* Filters learned by a deep belief network trained on whitened image patches.

In this article, we set out to test the performance of a deep belief network by evaluating its likelihood. After reviewing the relevant aspects of deep belief networks, we will derive a new consistent estimator for their likelihood and demonstrate the estimator's applicability in practice. We will investigate a particular deep belief network's capability to model the statistical regularities found in natural image patches. We will show that the deep belief network under study is not particularly good at capturing the statistics of natural image patches as it is outperformed with respect to the likelihood even by very simple mixture models. We will furthermore show that adding layers to the network has only a small effect on the overall performance of the model if the first layer is trained well enough and offer possible explanations for this observation by analyzing a best-case scenario of the greedy learning procedure commonly used for training deep belief networks.

2. Models

In this section we will review the statistical models used in the remainder of this article and discuss some of their properties relevant for estimating the likelihood of deep belief networks (DBNs). Throughout this section, the goal of applying statistical models is assumed to be the approximation of a particular distribution of interest, the *data distribution*. We will denote this distribution by $\tilde{p}$.

2.1 Boltzmann Machines

A *Boltzmann machine* is a potentially fully connected *undirected graphical model* with binary random variables. Its probability mass function is a Boltzmann distribution over 2^k binary states

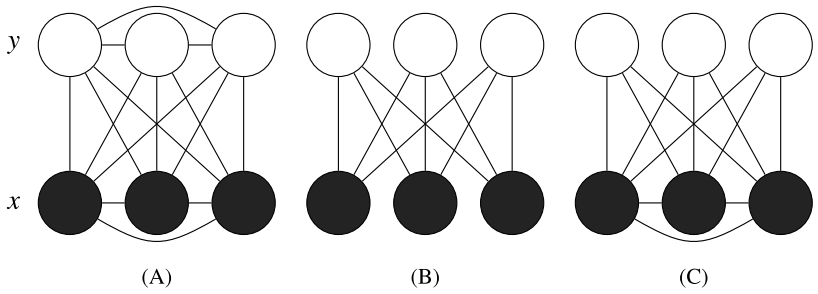


Figure 2: Boltzmann machines with different constraints on their connectivity. Filled nodes denote visible variables, unfilled nodes denote hidden variables. *A:* A fully connected Boltzmann machine. *B:* A restricted Boltzmann machine. *C:* A semi-restricted Boltzmann machine, which in contrast to RBMs also allows connections between the visible units.

$s \in \{0,1\}^k$ which is defined in terms of an *energy function* E,

$$q(s) = \frac{1}{Z}\exp(-E(s)), \quad Z = \sum_s \exp(-E(s)),$$

where E is given by

$$E(s) = -\frac{1}{2}s^\top W s - b^\top s = -\frac{1}{2}\sum_{i,j} s_i w_{ij} s_j - \sum_i s_i b_i$$

and depends on a symmetric weight matrix $W \in \mathbb{R}^{k\times k}$ with zeros on the diagonal, $w_{ii} = 0$ for all $i = 1,...,k$, and bias terms $b \in \mathbb{R}^k$. Z is called *partition function* and ensures the normalization of q. In the following, unnormalized distributions will be marked with an asterisk:

$$q^*(s) = Zq(s) = \exp(-E(s)).$$

Of particular interest for building DBNs are *latent variable Boltzmann machines*, that is, Boltzmann machines for which the states s are only partially observed (Figure 2). We will refer to states of observed or visible random variables as x and to states of unobserved or hidden random variables as y, such that $s = (x,y)$.

Maximum likelihood (ML) learning can be implemented by following the gradient of the log-likelihood. In Boltzmann machines, this gradient is conceptually simple yet computationally hard to evaluate. The gradient of the expected log-likelihood with respect to some parameter θ of the energy function is (e.g., Salakhutdinov, 2009):

$$\mathbf{E}_{\tilde{p}(x)}\left[\frac{\partial}{\partial\theta}\log q(x)\right] = \mathbf{E}_{q(x,y)}\left[\frac{\partial}{\partial\theta}E(x,y)\right] - \mathbf{E}_{\tilde{p}(x)q(y|x)}\left[\frac{\partial}{\partial\theta}E(x,y)\right]. \tag{1}$$

The first term on the right-hand side of this equation is the expected gradient of the energy function when both hidden and visible states are sampled from the model, while the second term is the expected gradient of the energy function when the hidden states are drawn from the conditional distribution of the model, given a visible state drawn from the data distribution, $\tilde{p}(x)$.

Evaluating these expectations, however, is computationally intractable for all but the simplest models. Even approximating the expectations with Monte Carlo methods is typically very slow (Long and Servedio, 2010). Two measures can be taken to make learning in Boltzmann machines feasible: constraining the Boltzmann machine in some way, or replacing the likelihood with a simpler objective function. The latter approach led to the introduction of the contrastive divergence (CD) learning rule (Hinton, 2002) which represents a tractable approximation to ML learning: In CD learning, the expectation over the model distribution $q(x,y)$ is replaced by an expectation over

$$q_{\mathrm{CD}}(x,y) = \sum_{x_0,y_1} \tilde{p}(x_0)q(y_1 \mid x_0)q(x \mid y_1)q(y \mid x),$$

from which samples are obtained by taking a sample x_0 from the data distribution, updating the hidden units, updating the visible units, and finally updating the hidden units again, while in each step keeping the respective set of other variables fixed. This corresponds to a single sweep of Gibbs sampling through all random variables of the model plus an additional update of the hidden units. If instead n sweeps of Gibbs sampling are used, the learning procedure is generally referred to as CD(n) learning. In the limit of large n, ML learning is regained (Salakhutdinov, 2009). An improved sampling scheme is offered by *persistent contrastive divergence* (PCD), in which the Markov chain is initialized not with a sample from the data distribution, but with the state of the Markov chain at the previous update of the gradient (Younes, 1989; Tieleman, 2008).

2.2 Restricted Boltzmann Machines

The first expectation on the right-hand side of Equation 1 can be made analytically tractable by constraining the energy function such that no direct interaction between two visible units or two hidden units is possible (Smolensky, 1986; Hinton, 2002),

$$E(x,y) = -x^\top W y - b^\top x - c^\top y.$$

Such a model is called a restricted Boltzmann machine (RBM). The corresponding graph has no connections between the visible units and no connections between the hidden units (Figure 2). Importantly, the unnormalized marginal distributions $q^*(x)$ and $q^*(y)$ can now be computed analytically by integrating out the respective other set of variables. The unnormalized marginal distribution of the visible units becomes

$$q^*(x) = \exp(b^\top x) \prod_j (1 + \exp(w_j^\top x + c_j)). \tag{2}$$

Two related models also used in this article are the *Gaussian RBM* (GRBM) (Salakhutdinov, 2009) and the *semi-restricted Boltzmann machine* (SRBM) (Osindero and Hinton, 2008). The GRBM employs continuous visible units and binary hidden units and can thus be used to model continuous data. Its energy function is given by

$$E(x,y) = \frac{1}{2\sigma^2}||x-b||^2 - \frac{1}{\sigma}x^\top W y - c^\top y.$$

A somewhat more general definition allows a different σ for each individual visible unit (Salakhutdinov, 2009). The conditional distribution $q(x \mid y)$ of a GRBM is a multivariate Gaussian distribution,

$$q(x \mid y) = \mathcal{N}(x; \sigma W y + b, \sigma^2 I).$$

Each binary state of the hidden units encodes one mean, while σ controls the variance of each Gaussian and is the same for all hidden units. The GRBM can therefore be interpreted as a mixture of an exponential number of Gaussian distributions with fixed, isotropic covariance and parameter sharing constraints.

In an SRBM, only the hidden units are constrained to have no direct connections to each other while the visible units are unconstrained (Figure 2). Analytic expressions are therefore only available for $q^*(x)$ but not for $q^*(y)$ and the visible units are no longer independent given a state for the hidden units.

2.3 Deep Belief Networks

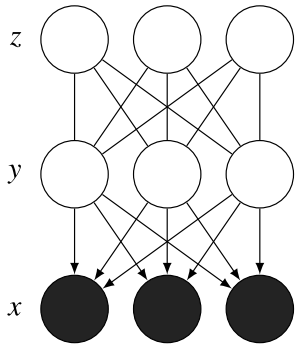


Figure 3: A graphical model representation of a two-layer deep belief network composed of two RBMs. The connections of the first layer are directed.

DBNs (Hinton and Salakhutdinov, 2006) are hierarchical generative models composed of several layers of RBMs or one of their generalizations. Let $q(x,y)$ and $r(y,z)$ be the densities of two RBMs over visible states x and hidden states y and z. Then the joint probability mass function of a two-layer DBN is defined to be

$$p(x,y,z) = q(x \mid y)r(y,z).$$

The resulting model is best described not as a deep Boltzmann machine but as a graphical model with undirected connections between y and z and directed connections between x and y (Figure 3). This definition can be recursively extended to DBNs with three or more layers by replacing $r(y,z)$ with another DBN. DBNs with an arbitrary number of layers have been shown to be universal approximators even if the number of hidden units in each layer is fixed to the number of visible units (Sutskever and Hinton, 2008). DBNs are easily generalized by allowing more general models as layers, such as the GRBM and the SRBM.

The learning procedure introduced by Hinton et al. (2006) for training DBNs makes two approximations to ML learning. The first approximation is made by training the DBN in a greedy manner: After the first layer of the model has been trained to approximate the data distribution, its parameters are fixed and only the parameters of the second layer are optimized. If θ is a parameter

of the second-layer density r, the gradient of the DBN's log-likelihood with respect to θ is

$$\frac{\partial}{\partial\theta}\log p(x) = \sum_y p(y \mid x)\frac{\partial}{\partial\theta}\log r(y). \tag{3}$$

However, exact sampling from the posterior distribution $p(y \mid x)$ is difficult. In order to make the training feasible, the posterior distribution is replaced by the factorial distribution $q(y \mid x)$. Training the DBN in this manner optimizes a variational lower bound on the log-likelihood (Hinton et al., 2006),

$$\sum_y q(y \mid x)\log r(y) \leq \log p(x) + const, \tag{4}$$

where *const* is constant in θ, which is a parameter of r. Taking the derivative of the left-hand side of Equation 4 with respect to θ yields (3) with the posterior distribution $p(y \mid x)$ replaced by $q(y \mid x)$. The greedy learning procedure can be generalized to more layers by training each additional layer to approximate the distribution obtained by conditionally sampling from each layer in turn, starting with the lowest layer.

After finishing the greedy training, Hinton et al. (2006) suggested to use the *wake-sleep algorithm* (Hinton et al., 1995) to fine-tune the parameters. Like the greedy algorithm, the wake-sleep algorithm optimizes a lower bound on the log-likelihood of the model, but in contrast optimizes all parameters concurrently. In addition, a separate set of RBMs is used and adapted to more closely approximate the DBN's posterior distribution over hidden units.

3. Likelihood Estimation

In this section, we will discuss the problem of estimating the likelihood of a two-layer DBN with joint density

$$p(x,y,z) = q(x \mid y)r(y,z). \tag{5}$$

That is, for a given visible state x, to estimate the value of

$$p(x) = \sum_{y,z} q(x \mid y)r(y,z).$$

We will later generalize this problem to more layers. As before, $q(x,y)$ and $r(y,z)$ refer to the densities of two RBMs.

Two difficulties arise when dealing with this problem in the context of DBNs. First, $r(y,z)$ depends on a partition function Z_r whose exact evaluation requires integration over an exponential number of states. Second, despite being able to integrate analytically over z, even computing just the unnormalized likelihood still requires integration over an exponential number of hidden states y,

$$p^*(x) = \sum_y q(x \mid y)r^*(y).$$

After briefly reviewing previous approaches to resolving these difficulties, we will propose an unbiased estimator for $p^*(x)$, its contribution being a possible solution to the second problem, and discuss how to construct a consistent estimator for $p(x)$ based on this result.

3.1 Importance Sampling

Since our estimator relies on importance sampling, we briefly review it here. Importance sampling is a Monte Carlo integration method for unbiased estimation of expectations (MacKay, 2003) and is based on the following observation: Let s be a density with $s(x) > 0$ whenever $q^*(x) > 0$ and let $w(x) = \frac{q^*(x)}{s(x)}$, then

$$\sum_x q^*(x) f(x) = \sum_x s(x) \frac{q^*(x)}{s(x)} f(x) = \mathbf{E}_{s(x)} \left[w(x) f(x) \right]$$

for any function f. s is called a *proposal distribution* for q and $w(x)$ is called *importance weight*. If $f(x) = 1$ for all x, we get

$$\mathbf{E}_{s(x)} \left[w(x) \right] = \sum_x s(x) \frac{q^*(x)}{s(x)} = Z_q. \tag{6}$$

Estimates of the partition function Z_q can therefore be obtained by drawing samples $x^{(n)}$ from a proposal distribution and averaging the resulting importance weights $w(x^{(n)})$. It was pointed out in Minka (2005) that minimizing the variance of the importance sampling estimate of the partition function (6) is equivalent to minimizing an α-divergence[1] between the proposal distribution s and the true distribution q. Therefore, for an estimator to work well in practice, s should be both close to q and easy to sample from.

3.2 Previous Work

The following is a brief summary of existing approaches to approximating the likelihood of RBMs and DBNs.

3.2.1 Annealed Importance Sampling

Salakhutdinov and Murray (2008) have shown how *annealed importance sampling* (AIS) (Neal, 2001) can be used to estimate the partition function of an RBM. AIS tries to circumvent some of the problems associated with finding a suitable proposal distribution. For a sequence of proposal distributions $s_1, ..., s_n$ and corresponding transition operators $T_1, ..., T_n$, one can show that

$$Z_q = \sum_x s_n(x_{n-1}) T_{n-1}(x_{n-2}; x_{n-1}) \cdots T_1(x_0; x_1) \frac{s^*_{n-1}(x_{n-1})}{s_n(x_{n-1})} \cdots \frac{q^*(x_0)}{s^*_1(x_0)},$$

where the sum integrates over all $x = (x_0, ..., x_{n-1})$. Hence, in order to estimate the partition function, we can draw independent samples x_{n-1} from a simple distribution s_n, use the transition operators to generate samples $x_{n-2}, ..., x_0$ from increasingly complex distributions, and average the resulting product of fractions given in the preceding equation. For details on how to choose the intermediate distributions and transition operators, see Salakhutdinov and Murray (2008).

1. With $\alpha = 2$. α-divergences are a generalization of the KL-divergence.

3.2.2 ESTIMATING LOWER BOUNDS

In Salakhutdinov and Murray (2008) it was also shown how estimates of a lower bound on the log-likelihood,

$$\log p(x) \geq \sum_y q(y \mid x) \log \frac{r^*(y) q(x \mid y)}{q(y \mid x)} - \log Z_r \tag{7}$$

$$= \sum_y q(y \mid x) \log r^*(y) q(x \mid y) + \mathbf{H}[q(y \mid x)] - \log Z_r, \tag{8}$$

can be obtained, provided the partition function Z_r is given. This is the same lower bound as the one optimized during greedy learning (4). Since $q(y \mid x)$ is factorial, the entropy $\mathbf{H}[q(y \mid x)]$ can be computed analytically. The only term which still needs to be estimated is the first term on the right-hand side of Equation 8. This was achieved in Salakhutdinov and Murray (2008) by averaging over samples drawn from $q(y \mid x)$.

3.2.3 CONSISTENT ESTIMATES

In Murray and Salakhutdinov (2009), a rather elaborate sampling scheme was devised to give unbiased estimates for the inverse posterior probability $\frac{1}{p(y|x)}$ of some fixed hidden state y. These estimates were then used to get unbiased estimates of $p^*(x)$ by taking advantage of the fact $p^*(x) = \frac{p^*(x,y)}{p(y|x)}$. The corresponding partition function was estimated using AIS, giving rise to a consistent estimator. While the estimator's Markov chain was constructed such that arbitrarily short runs of the Markov chain result in unbiased estimates of $p^*(x)$, even a single step of the Markov chain is slow compared to sampling from $q(y \mid x)$, as it was done for the estimation of the lower bound (7).

3.3 A New Estimator for DBNs

The estimator we will introduce in this section shares the same formal properties as the estimator proposed in Murray and Salakhutdinov (2009), but will use samples drawn from $q(y \mid x)$. This will make it conceptually as simple and as easy to apply in practice as the estimator for the lower bound (7), while providing us with consistent estimates of $p(x)$.

Let $p(x,y,z)$ be the joint density of a DBN as defined in Equation 5. By applying Bayes' theorem, we obtain

$$\begin{aligned} p(x) &= \sum_y q(x \mid y) r(y) \\ &= \sum_y q(y \mid x) \frac{q(x)}{q(y)} r(y) \\ &= \sum_y q(y \mid x) \frac{q^*(x)}{q^*(y)} \frac{r^*(y)}{Z_r}. \end{aligned} \tag{9}$$

A natural choice for an estimator of $p(x)$ is therefore

$$\begin{aligned} \hat{p}_N(x) &= \frac{1}{N} \sum_n \frac{q^*(x)}{q^*(y^{(n)})} \frac{r^*(y^{(n)})}{Z_r} \\ &= q^*(x) \frac{1}{Z_r N} \sum_n \frac{r^*(y^{(n)})}{q^*(y^{(n)})} \end{aligned} \tag{10}$$

where $y^{(n)} \sim q(y^{(n)} \mid x)$ for $n = 1,...,N$. For RBMs, the unnormalized marginals $q^*(x), q^*(y)$ and $r^*(y)$ can be computed analytically (2). Also note that the partition function Z_r only has to be calculated once for all visible states we wish to evaluate. We are not aware of any work which tried to use this estimator to estimate the likelihood of DBNs. In the following, we will discuss and investigate its properties.

Under the assumption that the partition function Z_r is known, $\hat{p}_N(x)$ provides an unbiased estimate of $p(x)$ since the sample average is always an unbiased estimate of the expectation. However, Z_r is generally intractable to compute exactly so that approximations become necessary. If in the estimate (10) the partition function Z_r is replaced by an unbiased estimate $\hat{Z}_r$, then the overall estimate will tend to overestimate the true likelihood,

$$\begin{aligned}\mathbf{E}\left[\frac{\hat{p}_N^*(x)}{\hat{Z}_r}\right] &= \mathbf{E}\left[\frac{1}{\hat{Z}_r}\right]\mathbf{E}\left[\hat{p}_N^*(x)\right] \\ &\geq \frac{1}{\mathbf{E}\left[\hat{Z}_r\right]}p^*(x) = p(x),\end{aligned}$$

where $\hat{p}_N^*(x) = Z_r\hat{p}_N(x)$ is an unbiased estimate of the unnormalized density. The second step is a consequence of Jensen's inequality and the averages are taken with respect to $\hat{p}_N(x)$ and $\hat{Z}_r$, which are independent; x is held fix.

While the estimator loses its unbiasedness for unbiased estimates of the partition function, it still retains its consistency. Since $\hat{p}_N^*(x)$ is unbiased for all $N \in \mathbb{N}$, it is also asymptotically unbiased,

$$\operatorname*{plim}_{N\to\infty} \hat{p}_N^*(x) = p^*(x).$$

Furthermore, if $\hat{Z}_{r,N}$ for $N \in \mathbb{N}$ is a consistent sequence of estimators for the partition function, it follows that

$$\operatorname*{plim}_{N\to\infty} \frac{\hat{p}_N^*(x)}{\hat{Z}_{r,N}} = \frac{\operatorname*{plim}_{N\to\infty} \hat{p}_N^*(x)}{\operatorname*{plim}_{N\to\infty} \hat{Z}_{r,N}} = \frac{p^*(x)}{Z_r} = p(x).$$

Unbiased and consistent estimates of Z_r can be obtained using AIS (Salakhutdinov and Murray, 2008). Note that although the estimator tends to overestimate the true likelihood in expectation and is unbiased in the limit, it is still possible for it to underestimate the true likelihood most of the time. This behavior can occur if the distribution of estimates is heavily skewed.

An important question which remains is whether the estimator is also good in terms of statistical efficiency. The more efficient an estimator, the less samples are required to obtain reliable estimates of the likelihood. If we cast Equation 9 into the form of Equation 6, we see that our estimator performs importance sampling with proposal distribution $q(y \mid x)$ and target distribution $p(y \mid x)$, where $p(x)$ can be seen as the partition function of an unnormalized distribution $p(x,y)$. As mentioned earlier, the efficiency of importance sampling estimates of partition functions depends on how well the proposal distribution approximates the true distribution. Therefore, for the proposed estimator to work well in practice, $q(y \mid x)$ should be close to $p(y \mid x)$. Note that a similar assumption is made when optimizing the variational lower bound during greedy learning. The lower bound (4) can be shown to be equal to

$$\sum_y q(y \mid x)\log r(y) = \log p(x) - D_{KL}(q(y \mid x)||p(y \mid x)) + const, \tag{11}$$

Estimate $p(x_0)$

1: $\hat{p} \leftarrow q_1^*(x_0)$
2: **for** $l = 1$ **to** $L-1$ **do**
3: $\quad x_l \sim q_l(x_l \mid x_{l-1})$
4: $\quad \hat{p} \leftarrow \hat{p} \cdot \frac{q_{l+1}^*(x_l)}{q_l^*(x_l)}$
5: **end for**
6: **return** $\hat{p}/Z_L$

Figure 4: Pseudocode for generating an unbiased estimate $\hat{p}$ of the probablity of x_0 under a DBN with L layers. Layer l here has unnormalized density q_l^*. Note that in practice, an average over multiple such estimates will generally be taken. Also note that Z_L will typically have to be estimated as well.

where again *const* is constant in the parameters of r. Training the DBN by optimizing the above lower bound therefore minimizes the KL-divergence between the true posterior and the approximating posterior, thereby making it a better proposal distribution. We will further address the question of statistical efficiency empirically in the experimental section.

The definition of the estimator for two-layer DBNs readily extends to DBNs with L layers. If $p(x)$ is the marginal density of a DBN whose layers are RBMs with densities $q_1(x_0,x_1),...,$ $q_L(x_{L-1},x_L)$ and partition functions $Z_1,...,Z_L$, and if we refer to the states of the random vectors in each layer by $x_0,...,x_L$, where x_0 contains the visible states and x_L contains the states of the top hidden layer, then

$$\begin{aligned} p(x_0) &= \sum_{x_1,...,x_L} q_L(x_{L-1},x_L) \prod_{l=1}^{L-1} q_l(x_{l-1} \mid x_l) \\ &= \sum_{x_1,...,x_{L-1}} q_L(x_{L-1}) \prod_{l=1}^{L-1} q_l(x_l \mid x_{l-1}) \frac{q_l^*(x_{l-1})}{q_l^*(x_l)} \\ &= q_1^*(x_0) \frac{1}{Z_L} \sum_{x_1,...,x_{L-1}} \prod_{l=1}^{L-1} q_l(x_l \mid x_{l-1}) \frac{q_{l+1}^*(x_l)}{q_l^*(x_l)}. \end{aligned}$$

In order to estimate this term, hidden states $x_1,...,x_{L-1}$ are generated in a feed-forward manner using the conditional distributions $q_l(x_l \mid x_{l-1})$. The weights $\frac{q_{l+1}^*(x_l)}{q_l^*(x_l)}$ are computed along the way, then multiplied together and finally averaged over all drawn states. Intuitively, the estimation process can be imagined as first assigning a basic value using the first layer and then correcting this value based on how the densities of each pair of consecutive layers relate to each other. Pseudocode for this procedure is given in Figure 4.

3.4 Potential Log-Likelihood

In this section, we will discuss a concept which appears in Roux and Bengio (2008) and which we will dub the *potential log-likelihood*. By considering a best-case scenario, the potential log-likelihood can give us an idea of the log-likelihood that can at best be achieved by training additional

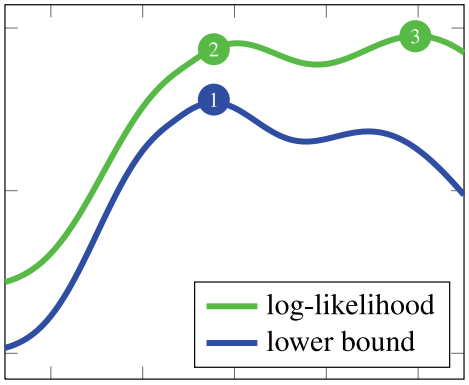


Figure 5: A cartoon explaining the potential log-likelihood. For each choice for the second layer of a DBN, we obtain a different value for the log-likelihood and its lower bound. From all possible choices, we take the distribution which maximizes the lower bound (1) and compute its log-likelihood (2). We call this particular log-likelihood the *potential log-likelihood*. It is still possible that the log-likelihood is larger for other distributions (3). However, it is unlikely that such a distribution will be found, as the second layer is optimized with respect to the lower bound.

layers using the greedy learning of Hinton and Salakhutdinov (2006). Its usefulness will become apparent in the experimental section.

Let $q(x,y)$ be the distribution of an already trained RBM or one of its generalizations, and let $r(y)$ be a second distribution—not necessarily the marginal distribution of any Boltzmann machine. As in section 2.3, $r(y)$ serves to replace the prior distribution over the hidden variables, $q(y)$, and to thereby improve the marginal distribution over x, $\sum_y q(x \mid y)r(y)$. As above, let $\tilde{p}(x)$ denote the data distribution. Our goal is to increase the expected log-likelihood of the model distribution with respect to r,

$$\sum_x \tilde{p}(x) \log \sum_y q(x \mid y) r(y). \tag{12}$$

In applying the greedy learning procedure, we try to reach this goal by optimizing a lower bound on the log-likelihood (4), or equivalently, by minimizing the following KL-divergence:

$$D_{\mathrm{KL}}\left[\sum_x \tilde{p}(x) q(y \mid x) || r(y)\right] = -\sum_x \tilde{p}(x) \sum_y q(y \mid x) \log r(y) + const,$$

where *const* is constant in r.

The KL divergence is minimal if $r(y)$ is equal to

$$\sum_x \tilde{p}(x) q(y \mid x) \tag{13}$$

for every y. Since RBMs are universal approximators (Roux and Bengio, 2008), this distribution could in principle be approximated arbitrarily well by a single, potentially very large RBM. Assume therefore that we have found this distribution, that is, we have maximized the lower bound with respect to all possible distributions r. Then, the distribution for the DBN which we obtain by replacing r in (12) with (13) is given by

$$\begin{aligned}\sum_y q(x \mid y) \sum_{x_0} \tilde{p}(x_0) q(y \mid x_0) &= \sum_{x_0} \tilde{p}(x_0) \sum_y q(x \mid y) q(y \mid x_0) \\ &= \sum_{x_0} \tilde{p}(x_0) q_0(x \mid x_0),\end{aligned}$$

where we have used the *reconstruction distribution*

$$q_0(x \mid x_0) = \sum_y q(x \mid y) q(y \mid x_0),$$

which can be sampled from by conditionally sampling a state for the hidden units, and then, given the state of the hidden units, conditionally sampling a *reconstruction* of the visible units. The log-likelihood we achieve with this lower-bound optimal distribution is given by

$$\sum_x \tilde{p}(x) \log \sum_{x_0} \tilde{p}(x_0) q_0(x \mid x_0).$$

We will refer to this log-likelihood as the *potential log-likelihood*. Note that the potential log-likelihood is not a true upper bound on the log-likelihood that can be achieved with greedy learning, as suboptimal solutions with respect to the lower bound might still give rise to higher log-likelihoods. However, if such a solution was found, it would have been rather by accident than by design. The situation is depicted in the cartoon in Figure 5.

4. Experiments

We performed two types of experiments. In the first, we tested the performance of our estimator. In the second, we further investigated the capabilities of a DBN architecture proposed by Osindero and Hinton (2008) as a model for natural images. The model consists of a GRBM in the first layer and SRBMs in the subsequent layers.

For all but the topmost layer, our estimator requires evaluation of the unnormalized marginal density over hidden states (see Figure 4). In an SRBM, however, integrating out the visible states is analytically intractable. Fortunately, an RBM with the same hidden-to-visible connections and the same bias weights but without the lateral connections turned out to be an efficient enough proposal distribution. We estimated the unnormalized SRBM marginals using

$$q^*(y) = \sum_x q^*(x,y) = \sum_x q'(x \mid y) \frac{q^*(x,y)}{q'(x \mid y)} \approx \frac{1}{N} \sum_n \frac{q^*(x^{(n)},y)}{q'(x^{(n)} \mid y)},$$

where q' is the density of an RBM which approximates the density of the SRBM, q.

4.1 Testing the Estimator

As a first test, we considered a three-layer model for which the likelihood is still tractable. It employed 15 hidden units in each of the first two layers, 50 hidden units in the third layer and was

layers	true neg. log-likelihood	est. neg. log-likelihood
1	2.0482569	2.0484440
2	2.0478912	2.0477577
3	2.0478672	2.0474685

Table 1: True and estimated negative log-likelihood in bits per component of a small DBN trained on 4 by 4 image patches. The estimated likelihood closely matches the true likelihood. Adding more layers to the network did not help to improve the performance if the GRBM employed only few hidden units.

trained on 4 by 4 pixel image patches taken from the van Hateren image data set (van Hateren and van der Schaaf, 1998). The image patches were preprocessed using a standard battery of preprocessing steps including a log-transformation, a centering step and a whitening step. Additionally, the DC component was projected out and only the remaining 15 components of each patch were used for training (for a more detailed description of the preprocessing, see Eichhorn et al., 2009). Brute-force and estimated results are given in Table 1.

We also compared results obtained with our estimator applied to larger models to results obtained by Murray and Salakhutdinov (2009). Using the same preprocessing of the image patches and the same hyperparameters as in Murray and Salakhutdinov (2009), we trained a two-layer model with 2000 hidden units in the first layer and 500 hidden units in the second layer on 20 by 20 pixel van Hateren image patches. In contrast to the preprocessing of the smaller image patches, the DC component was not removed. We used 150000 patches for training and 50000 patches for testing and obtained a negative log-likelihood of 2.047 bits per component on the test set, compared to 2.032 bits reported by Murray and Salakhutdinov (2009).

We further evaluated a two-layer model with 500 hidden units in the first and 2000 hidden units in the second layer trained on a binarized version of the MNIST data set. For the first layer we took the parameters from Salakhutdinov (2009), which were available online. We then tried to match the training of the second-layer RBM with 2000 hidden units using the information given in Salakhutdinov (2009) and Murray and Salakhutdinov (2009). Evaluating the model on the whole test set, we obtained a result of 0.357 bits per component compared to 0.358 bits reported by Murray and Salakhutdinov (2009).

We measured the statistical efficiency of our estimator in terms of the *effective sample size* (ESS) (Kong et al., 1994). For N samples, target density p and proposal density q, the ESS is defined to be

$$\frac{N}{1+\mathbf{Var}_q(y)\left[\frac{p(y)}{q(y)}\right]} = \frac{N}{1+D_2[p(y)||q(y)]}.$$

D_2 is the α-divergence with $\alpha = 2$ and is also known as the χ^2-divergence. In our case, the target distribution is the conditional distribution over the hidden states of a DBN given a state for the visible units. If p equals q, then the variance of the importance weights is zero, so that a single sample from the posterior would suffice to get a perfect estimate of the unnormalized probability of any data point. In that case, the ESS would be N.

Note that the success of training DBNs is also influenced by the ESS. The tightness of the lower bound optimized during training depends on the goodness of the same approximation, although it is measured in terms of the KL-divergence rather than the χ^2-divergence (see Equation 11).

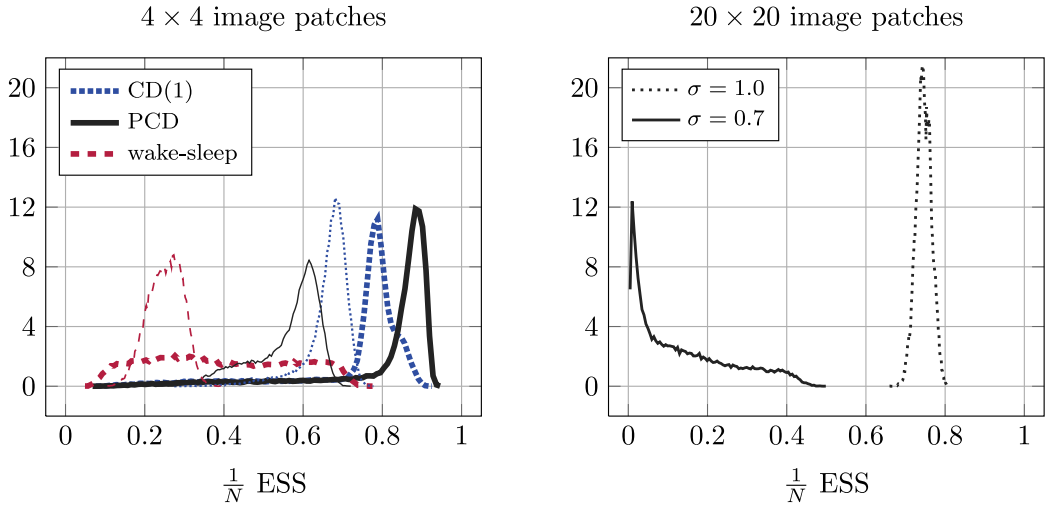


Figure 6: *Left:* Distributions of relative effective sample sizes (ESS) for two-layer DBNs (thick lines) and three-layer DBNs (thin lines) trained on 4 by 4 pixel image patches using different training rules. The larger the ESS for a given data point, the more efficient is our estimator in determining its probability. *Right:* Ditributions of relative ESSs for two two-layer models trained on 20 by 20 pixel image patches. Both models were trained with PCD but using different hyperparameters. While the GRBM yielded a better performance for smaller σ, it also caused the evaluation and the training of subsequent layers to be more difficult.

Because the distributions we are interested in are conditional distributions which are dependent on the state of the visible units, we obtain a different ESS for each data point. Hence, we get a distribution over ESSs for each model (Figure 6). We found that the quality of the proposal distribution was strongly dependent on the parameters of the model. Training a model on the larger image patches using the same hyperparameters as used by Osindero and Hinton (2008) and Murray and Salakhutdinov (2009), for example, led to a distribution of ESSs which was sharply peaked around 0.75. This made it easier to train subsequent layers and to evaluate the two-layer model. Using a smaller value for the Gaussian noise, on the other hand, led to a better likelihood but also a worse approximation to the posterior distribution. Consequently, getting accurate estimates of the log-likelihood took longer (right plot in Figure 7).

On a single core of an AMD Opteron 6174 machine with 2.20 GHz and with an implementation written in Python, it took us about 10 minutes to get accurate results for 50 data points taken from the MNIST data set. Murray and Salakhutdinov (2009) report that they needed about 50 minutes on a Pentium Xeon 3.00 GHz to get stable results for 50 data points. However, a fair comparison of computational efficiency is difficult, even when the comparison is performed with the same machine and the same programming framework is used. For the larger models trained on 20 by 20 van Hateren image patches, it took us less than a second and up to a few minutes to get reasonably accurate estimates for 50 samples (Figure 7). Note that even for the case where the distribution of ESSs is peaked around a value very close to zero, evaluating a large number of training samples was

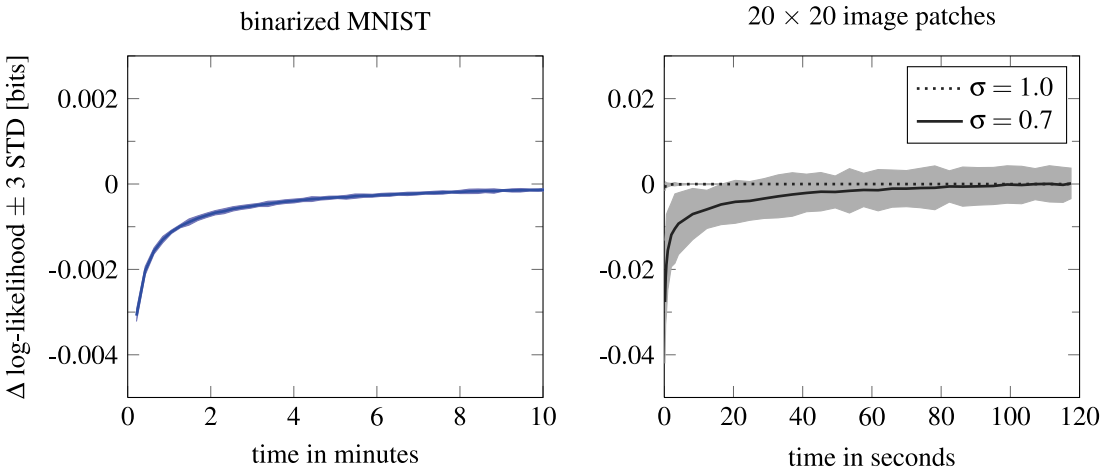


Figure 7: *Left:* The log-likelihood in bits per component was estimated for 50 test samples using different numbers of proposal samples. The difference between each estimate and the most accurate estimate was then measured. The plots show the mean difference and its standard deviation relative to the time it takes to compute the estimates. The bias in the estimates is due to the logarithm. *Right:* The same plot for a different model. The time it takes to get accurate estimates depends on the parameters of the model. Using the parameters obtained with the hyperparameters suggested in (Osindero and Hinton, 2008), already a single sample yielded good estimates of the log-likelihood (about 0.2 seconds). But even for the case where our estimator showed the least statistical efficiency, it only took a few minutes to evaluate 50 samples on a single CPU.

more than feasible. To further reduce computation time, the evaluation can easily be parallelized by splitting the test set into batches which are evaluated separately.

One likely reason for the efficiency of our estimator is the optimization of the lower bound during training (see discussion around Equation 11). This means that if the model is trained with a different learning algorithm, it should become less efficient. We found that this was indeed the case if models were further optimized using the wake-sleep algorithm (Figure 6).

4.2 Model Comparison

We compared the DBN's performance to the performance of complete ICA (Comon, 1994; Eichhorn et al., 2009) as well as several mixture distributions. Perhaps closest in interpretation to the GRBM as well as to the DBN is the mixture of isotropic Gaussian distributions (MoIG) with identical covariances and varying means. Recall that the GRBM can be interpreted as a mixture of a very large number of isotropic Gaussian distributions with weight sharing constraints. After the parameters of the GRBM have been fixed, adding layers to the network only changes the prior distribution over the Gaussians, but does not alter their means. Other models taken into account are mixtures of Gaussians with unconstrained covariance but zero mean (MoG) and mixtures of elliptically contoured distributions with zero mean (MoEC) (Bethge and Hosseini, 2007), of which a special case is the more well-known mixture of Gaussian scale mixtures (e.g., Guerrero-Colon et al., 2008).

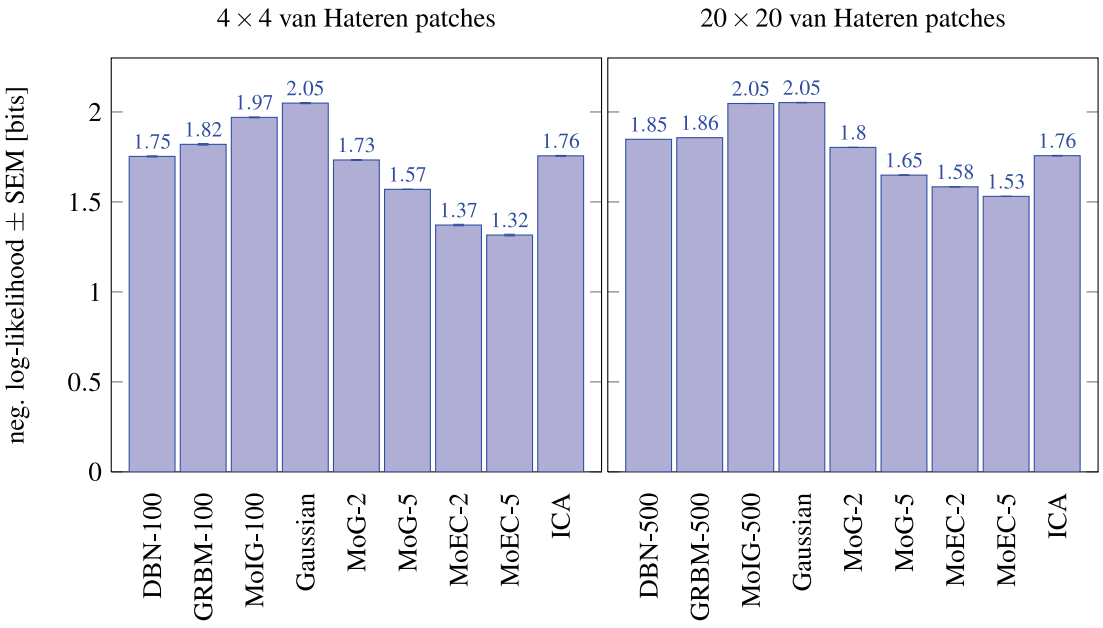


Figure 8: A comparison of different models. For each model, the estimated negative log-likelihood in bits per data component is shown, averaged over 10 independent trials with independent training and test sets. For the GRBM and DBN trained on 20 by 20 image patches, the performance obtained from a single trial is shown. The number behind each model hints either at the number of hidden units or at the number of mixture components used. DBNs were trained using the greedy learning rule and fine-tuned using the wake-sleep algorithm. Boltzmann machines were trained with PCD. Larger values correspond to worse performance.

Using PCD, we trained a three-layer model with 100 hidden units per layer on the smaller 4 by 4 image patches. As expected, both the GRBM and the DBN performed better than the mixture of isotropic Gaussians. Strikingly, however, the DBN was outperformed even by the mixture of Gaussians with just two components. The overall results suggest that mixture models with freely varying covariance are better suited for modeling the statistics of natural images than mixture models with constrained covariance (Figure 8), which is consistent with previous observations that having a flexible model of the covariance structure is important (e.g., Karklin and Lewicki, 2009; Ranzato et al., 2010a).

Adding a second layer to the network only helped very little and we found that the better the performance achieved by the GRBM, the smaller the improvement contributed by subsequent layers. For the hyperparameters that led to the best performance, adding a third layer had no effect on the likelihood and in some cases even led to a decrease in performance (Figure 9). This was despite the apparently reasonable approximation to the posterior distribution over the hidden states (Figure 6). The hyperparameters were selected by performing separate grid searches for the different layers (for details, see Appendix A). PCD on average yielded a slightly worse performance than CD(5) or CD(10), indicating that the Markov chain used during training to sample from the model converged

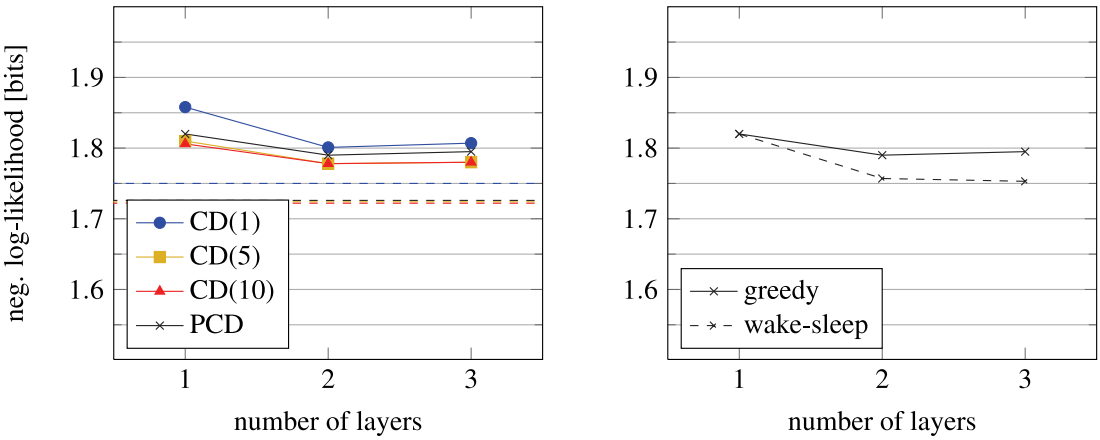


Figure 9: *Left:* Estimated performance of three DBN-100 models trained on 4×4 van Hateren image patches using the greedy learning rule. Each point represents an average over 10 models trained on different training sets. Smaller values correspond to better performance. Error bars were too small to be visible and were therefore left out. The dashed lines indicate the estimated negative potential log-likelihood. *Right:* Performance after fine-tuning the model parameters with the two-layer and three-layer models with the wake-sleep algorithm.

quickly to the true distribution. For other sets of hyperparameters we found that adding a third layer was still able to yield some improvement, but the overall performance of the three-layer model achieved with these hyperparameters was worse.

We also trained a two-layer DBN with 500 hidden units in the first layer and 2000 hidden units in the third layer on 20 by 20 pixel image patches. Using PCD instead of CD(1) and a smaller value for σ led to a performance which is about 0.18 bits per pixel better than the performance reported by Murray and Salakhutdinov (2009). However, the performance achieved by the model was again worse than the performance achieved by other, simpler models. As for the smaller model, the improvement in performance of the GRBM led to a smaller improvement gained by adding a second layer to the model. Despite the much larger second layer, the performance gain induced by the second layer was even less than for the smaller models. Both patch sizes led to the same ordering of the models and resulted in an overall similar picture (Figure 8).

The improvement of the DBN over the GRBM trained with PCD is about 0.05 bits per component for the smaller model and 0.01 bits for the larger model. An important question is why this improvement is so small. Insight into this question can be gained by evaluating the potential log-likelihood, as it represents a practical limit to the performance which can be achieved by means of the approximate greedy learning procedure and could in principle be evaluated even before training any additional layers. If the potential log-likelihood of a GRBM is close to its log-likelihood, adding layers is a priori unlikely to prove useful. Unfortunately, exact evaluation of the potential log-likelihood is intractable, as it involves two nested integrals with respect to the data distribution,

$$\int \tilde{p}(x) \log \int \tilde{p}(x_0) q_0(x \mid x_0) dx_0 dx. \tag{14}$$

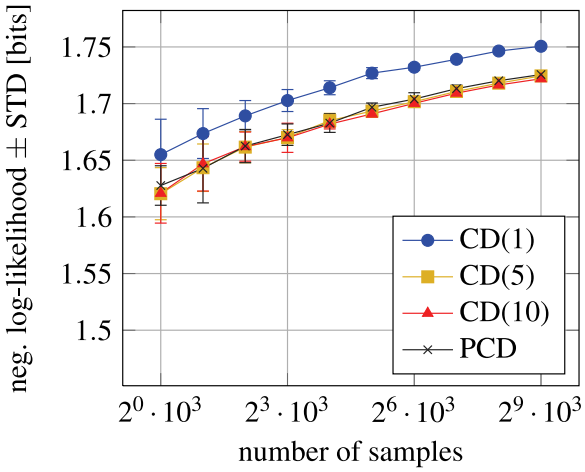


Figure 10: Estimated negative potential log-likelihood of the GRBM. Each plot represents an average over 10 GRBMs trained on different training sets. Error bars indicate one standard deviation. After 512000 samples to approximate the integrals of the potential log-likelihood, the estimates have still not converged, suggesting that the true potential log-likelihood is even worse.

Nevertheless, optimistic estimates of this quantity can still tell us something about the DBN's capability to improve over the GRBM. We estimated the potential log-likelihood using the same set of data samples to approximate both integrals, thereby encouraging optimistic estimation. Note that estimating the potential log-likelihood in this manner is similar to evaluating the log-likelihood of a kernel density estimate on the training data, although the reconstruction distribution $q_0(x \mid x_0)$ might not correspond to a valid kernel. Also note that by taking more and more data samples, the estimate of the potential log-likelihood should become more and more accurate. Figure 10 therefore suggests that the negative potential log-likelihood of a GRBM trained with PCD is at least 1.72 or larger, which is still worse than the performance of, for example, the mixture of Gaussian distributions with 5 components.

The potential log-likelihood is a joint property of the trained first-layer model and the greedy learning procedure. If the greedy algorithm is responsible for the weak performance of the model, it should be possible to get better results by using a non-greedy strategy. We therefore tried to improve the model's performance by fine-tuning the weights with the wake-sleep algorithm (Hinton et al., 1995). We trained the model for another 100 epochs using the same set of hyperparameters but with smaller learning rates. This led to a small improvement (Figure 9), but the likelihood remained behind the performance of the mixture of Gaussians with two components. By applying the same strategy to the larger model, we were unable to get any further improvements. If the learning rate was chosen large enough to cause a measurable change in performance, the performance of the model decreased.

5. Discussion

In this paper, we have introduced a new estimator for the likelihood of DBNs. We have shown that it can be very efficient for models trained with the greedy learning algorithm and efficient enough to evaluate models fine-tuned using the wake-sleep algorithm. However, as we have also seen, the estimator's performance depends on the particular parameters of the model and the way the model was trained. The estimator is unbiased for the unnormalized likelihood, but only asymptotically unbiased for the log-likelihood. When evaluating the log-likelihood, this means that the number of samples from the proposal distribution has to be chosen large enough to ensure that the effects of the bias are small. Since the number of proposal samples is the only parameter of our estimator and the evaluation can generally be performed quickly, a good way to do this is to test the estimator for different numbers of proposal samples on a smaller test set.

Using our estimator, we have shown that DBNs based on GRBMs and SRBMs are not well suited for capturing the statistics of natural images. Since the family of DBNs includes a very large number of models if we allow arbitrary Boltzmann machines in the definition, this of course does not imply that no DBN is able to model natural images well. In fact, other models like the mcRBM or mPoT (Ranzato and Hinton, 2010; Ranzato et al., 2010b) promise to be much better models of natural images than the GRBM and can be used to construct DBNs. One goal of this paper, however, was to see if a deep network with simple layer modules could perform well as a generative model for natural images. While more complex Boltzmann machines are likely to achieve a better likelihood, it is not clear why they should also be good choices as layer modules for constructing DBNs. In particular, it is not clear why they should also lead to large improvements through the addition of layers. In our experiments, better performances achieved with the first layer were always accompanied by a decrease in the performance gained by adding layers. Note that a model which achieves a high likelihood could still have a potential log-likelihood very close to it. This would make a model a better model for natural images, but not the best choice as a layer module for DBNs trained with the greedy learning algorithm.

In Ranzato et al. (2011) it was shown that adding a second-layer RBM to the mPoT model leads to qualitatively different and visually more appealing samples. However, as we have shown in this work, the appearance of samples can be misleading when judging the generative performance of a probabilistic model. For the DBN based on GRBMs and SRBMs we arrive at a different conclusion about the model's capabilities than Osindero and Hinton (2008), who based their conclusions mainly on statistics derived from model samples. We believe that it is therefore worth the effort to come up with statistical estimators which directly target the likelihood or related quantities.

The relationship between generative models and models used for object recognition is not yet fully understood. Hence, the implications of our results on the object recognition performance of features learned by a DBN are also not clear. Most DBNs used to learn features for solving supervised tasks rarely exceed more than a few layers (e.g., Ranzato and Hinton, 2010 show that the effects of adding layers to a GRBM or mcRBM on object recognition performance can be small or even adverse). This observation might be related to the observations made here, that adding layers does not help much to improve the likelihood on natural image patches. A possible explanation for the small contribution of each layer would be that too little information is carried by the hidden representations of each layer about the respective visible states. This is consistent with a small potential log-likelihood and would affect both object recognition and generative performance. If the visible states can be perfectly reconstructed from the hidden states, that is, when $q_0(x \mid x_0) =$

$\delta(x - x_0)$, the potential log-likelihood reduces to the negative entropy of the data—the maximum value for the likelihood that any model can achieve.

A much less frequently used hierarchical model is *hierarchical ICA*, for which a greedy learning algorithm was introduced by Chen and Gopinath (2001) and which can be seen as a special case of projection pursuit density estimation. In ICA, all the information about a visible state is retained in the hidden representation, so that the potential log-likelihood is optimal for this layer module. As shown in Figure 8, already a single ICA layer can compete with a DBN based on RBMs. Furthermore, adding layers to the network is guaranteed to improve the likelihood of the model (Chen and Gopinath, 2001) and not just a lower bound as with the greedy learning algorithm for DBNs. Hosseini and Bethge (2009) have shown that adding layers does indeed give significant improvements of the likelihood, although it was also shown that hierarchical ICA cannot compete with other, non-hierarchical models of natural images. An interesting question is whether representations learned by this model can also compete or outperform those learned by DBNs in supervised tasks. Comparing the supervised performance of features learned with hierarchical ICA and the model discussed here could shed further light on the importance of the likelihood when training DBNs for supervised tasks.

A lot of research has been devoted to creating new layer modules (e.g., Roux et al., 2010; Ranzato and Hinton, 2010; Lee and Ng, 2007; Welling et al., 2005) and finding better approximations to ML learning for training these (e.g., Gutmann and Hyvärinen, 2010; Sohl-Dickstein et al., 2009; Tieleman, 2008). To our knowledge, much less work has been done to improve the general strategy for training DBNs since its introduction. Currently, the lower layers are trained in a way which is independent of whether additional layers will be added to the network. A better performance could be achieved by devising alternative learning schemes in which the lower layers are optimized to better assist the upper layers, for example by making sure that the potential log-likelihood stays large. Directly regularizing with respect to the potential log-likelihood is difficult due to the nested integrals (see Equation 14). An improvement could nevertheless indirectly be achieved by maximizing the information that is carried in the hidden representations about the states of the visible units.

An alternative strategy would be to use non-greedy learning strategies such as the wake-sleep algorithm, for which the potential log-likelihood no longer plays a critical role. We showed that in one case, the wake-sleep algorithm was able to further improve the likelihood. In another case, we were unable to get any improvement. While more extensive tests are necessary before final conclusions about its effectiveness should be made, its potential to improve the likelihood of the DBN discussed here seems rather limited.

A very recent paper by Ngiam et al. (2011) introduced an alternative approach to hierarchical modeling of natural images. Instead of stacking probabilistic models on top of each other, the approach is based on a single Boltzmann machine in which the energy function itself is hierarchically organized. This allows them to jointly train all parameters of the network and also to evaluate the likelihood much more easily. The paper reports the likelihood of several instances of the model and shows that the model is able to take advantage of multiple layers. Unfortunately, it missed to also report the likelihood of other, more well known models of natural images. This makes it hard to judge the performance of the model, as the absolute value of the likelihood is highly dependent on the preprocessing of the data.

Despite good arguments for why hierarchical models should excel at modeling natural images, no hierarchical model has been convincingly shown to yield state-of-the-art generative performance

and outperform other, much simpler models, such as the mixture of Gaussians, in terms of the log-likelihood.

Acknowledgments

We would like to thank the anonymous reviewers for helpful comments and suggestions. We also thank Iain Murray for providing us with a binarized version of the MNIST data set. This work is supported by the German Ministry of Education, Science, Research and Technology through the Bernstein award to Matthias Bethge (BMBF, FKZ: 01GQ0601) and the Max Planck Society.

Appendix A.

In all experiments, we used stochastic gradient descent with a batch size of 100 data points to train DBNs. For the 4 by 4 pixel image patches, we used $5 \cdot 10^4$ training and $5 \cdot 10^4$ test samples in each trial. For the case of 20 by 20 pixel image patches, we used $15 \cdot 10^4$ training and $5 \cdot 10^4$ test samples.

When PCD was used for training, the persistent Markov chain was updated using one Gibbs sampling step before each computation of the gradient.

For the GRBM with 100 hidden units trained on 4 by 4 pixel image patches, we used a σ of 0.65 and trained the models for 200 epochs. Together with CD(1) we used a learning rate of 10^{-2} and weight decay of 10^{-2} times the learning rate. With PCD, CD(5) and CD(10) we used a learning rate of $5 \cdot 10^{-3}$. For all parameters and all models, a momentum parameter of 0.9 used.

We initialized the second-layer SRBM so that its marginal distribution matched that of the GRBM and trained it for 200 epochs. During training, approximate samples from the conditional distribution of the visible units were obtained using 20 parallel mean field updates with a damping parameter of 0.2 (Welling and Hinton, 2002). For all sampling schemes, we used a learning rate of $5 \cdot 10^{-3}$ for hidden-to-visible connections and a learning rate of $5 \cdot 10^{-4}$ for lateral connections. The weight decay was set to $5 \cdot 10^{-3}$. The third layer was randomly initialized and had to be trained for 500 epochs before convergence was reached. It employed the same hyperparameters as the second layer.

The hyperparameters were selected by performing several grid searches. After the hyperparameters of the GRBM had been selected, we performed a second grid-search for the second-layer SRBM and used the same hyperparameters for the third-layer SRBM. We performed separate grid searches for CD(1) and PCD, but used the hyperparameters found for PCD also with CD(5) and CD(10). Each grid search comprised 75 hyperparameter combinations for the GRBM and 45 hyperparameter combinations for the SRBM. After the hyperparameters had been selected, we retrained the model.

For the larger model applied to 20 by 20 pixel image patches, we trained both layers for 200 epochs using PCD. For the GRBM, we used a σ of 0.7, a learning rate of 10^{-3} and weight decay of 10^{-2}. For the SRBM, we used a learning rate of 10^{-3} for the hidden-to-visible connections and $5 \cdot 10^{-4}$ for the lateral connections. We used 30 parallel mean field updates with a damping rate of 0.2 to sample the visible units. The hyperparameters of the GRBM were selected by performing a grid search over 66 hyperparameter combinations. For the SRBM we tried 18 different combinations. Due to the high computational cost of training the two-layer model, we took the parameters which

achieved the best result in the grid search for our comparison and evaluated it using a separate test set.

Partition functions were estimated using AIS. For the smaller model we used 10^3 intermediate distributions with a linear annealing schedule, 100 samples in the first and 10^3 samples in the second and third layer. For the larger model we used a linear annealing schedule, $2 \cdot 10^4$ intermediate distributions and 100 importance samples in both layers. The transition operator for the SRBM was implemented using Gibbs sampling. Conditioned on the hidden units, the visible units were updated in a random order.

To estimate the unnormalized log-probability of each data point with respect to the smaller models, we used 200 samples from the proposal distribution of our estimator for the two-layer, and 500 samples for the three-layer model. For the larger model we used 2000 samples, although less would have been sufficient.

For the fine-tuning of the smaller models with the wake-sleep algorithm, we reduced the learning rates to $\frac{1}{4}$ of the original learning rates. Using larger learning rates led to a decrease in performance. We used the same hyperparameters for the DBN representing the proposal distribution. We trained the models for 100 epochs. Training the models for 200 epochs led to no further improvements. We controlled for the estimator's bias by testing the estimators behavior on a smaller test set for different numbers of proposal samples. We used 4000 samples from the proposal distribution of our estimator to estimate the log-likelihood. Using only 500 samples led to essentially the same results, but the small difference in performance between the two-layer and three-layer model was less visible.

To estimate the unnormalized hidden marginals of SRBMs, we used 100 proposal samples from an approximating RBM with the same hidden-to-visible connections and same bias weights.

Lastly, note that the performance of the GRBM and the DBN might still be improved by taking a larger number of hidden units. A post-hoc analysis revealed that the GRBM does indeed not overfit but continues to improve its performance if the variance is decreased while increasing the number of hidden units. This is of course also true for the other models we evaluated, whose performance can still be improved if we allow them to use more parameters.

Code for training and evaluating deep belief networks using our estimator can be found under

`http://www.bethgelab.org/code/theis2011/`.

References

M. Bethge and R. Hosseini. Method and device for image compression. Patent WO/2009/146933, 2007.

S. S. Chen and R. A. Gopinath. Gaussianization. *Advances in Neural Information Processing Systems 13*, 2001.

P. Comon. Independent component analysis, a new concept? *Signal processing*, 36(3):287–314, 1994.

J. Eichhorn, F. Sinz, and M. Bethge. Natural image coding in V1: How much use is orientation selectivity? *PLoS Computational Biology*, 5(4), 2009.

D. J. Felleman and D. C. van Essen. Distributed hierarchical processing in the primate cerebral cortex. *Cerebral Cortex*, 1991.

K. Fukushima. Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. *Biological Cybernetics*, 1980.

J. A. Guerrero-Colon, E. P. Simoncelli, and J. Portilla. Image denoising using mixtures of gaussian scale mixtures. *Proceedings of the 15th IEEE International Conference on Image Processing*, 2008.

M. Gutmann and A. Hyvärinen. Noise-contrastive estimation: A new estimation principle for unnormalized statistical models. *Proceedings of the 13th International Conference on Artificial Intelligence and Statistics*, 2010.

G. E. Hinton. Training products of experts by minimizing contrastive divergence. *Neural Computation*, 14(8):1771–1800, 2002.

G. E. Hinton and R. Salakhutdinov. Reducing the dimensionality of data with neural networks. *Science*, 313:504–507, Jan 2006.

G. E. Hinton, P. Dayan, B. Frey, and R. Neal. The "wake-sleep" algorithm for unsupervised neural networks. *Science*, Jan 1995.

G. E. Hinton, S. Osindero, and Y. Teh. A fast learning algorithm for deep belief nets. *Neural Computation*, 18(7):1527–1554, Jul 2006.

R. Hosseini and M. Bethge. Hierachical models of natural images. *Frontiers in Computational Neuroscience*, 2009.

Y. Karklin and M. S. Lewicki. Emergence of complex cell properties by learning to generalize in natural scenes. *Nature*, 2009.

A. Kong, J. S. Liu, and W. H. Wong. Sequential imputations and bayesian missing data problems. *Journal of the American Statistical Association*, 89(425):278–288, 1994.

Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel. Backpropagation applied to handwritten zip code recognition. *Neural Computation*, 1989.

H. Lee and A. Ng. Sparse deep belief net model for visual area v2. *Advances in Neural Information Processing Systems 19*, 2007.

P. Long and R. Servedio. Restricted boltzmann machines are hard to approximately evaluate or simulate. *Proceedings of the 27th International Conference on Machine Learning*, 2010.

D. J. C. MacKay. *Information Theory, Inference, and Learning Algorithms*. Cambridge University Press, 2003.

T. Minka. Divergence measures and message passing. *Microsoft Research Technical Report (MSR-TR-2005-173)*, 2005.

A. Mohamed, G. Dahl, and G. E. Hinton. Deep belief networks for phone recognition. *NIPS 22 workshop on deep learning for speech recognition*, 2009.

I. Murray and R. Salakhutdinov. Evaluating probabilities under high-dimensional latent variable models. *Advances in Neural Information Processing Systems 21*, 2009.

R. M. Neal. Annealed importance sampling. *Statistics and Computing*, 11(2):125–139, Jan 2001.

J. Ngiam, Z. Chen, P. Koh, and A. Y. Ng. Learning deep energy models. *Proceedings of the 28th International Conference on Machine Learning*, 2011.

S. Osindero and G. E. Hinton. Modeling image patches with a directed hierarchy of markov random fields. *Advances in Neural Information Processing Systems 20*, 2008.

M. Ranzato and G. E. Hinton. Modeling pixel means and covariances using factorized third-order boltzmann machines. *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8, May 2010.

M. Ranzato, A. Krizhevsky, and G. E. Hinton. Factored 3-way restricted boltzmann machines for modeling natural images. *Proceedings of the 13th International Conference on Artificial Intelligence and Statistics*, 2010a.

M. A. Ranzato, V. Mnih, and G. E. Hinton. Generating more realistic images using gated mrfs. *Advances in Neural Information Processing Systems 23*, 2010b.

M. A. Ranzato, J. Susskind, V. Mnih, and G. E. Hinton. On deep generative models with applications to recognition. *IEEE Conference on Computer Vision and Pattern Recognition*, 2011.

N. Le Roux and Y. Bengio. Representational power of restricted boltzmann machines and deep belief networks. *Neural Computation*, 20(6):1631–1649, 2008.

N. Le Roux, N. Heess, J. Shotton, and J. Winn. Learning a generative model of images by factoring appearance and shape. *Microsoft Research Technical Report (MSR-TR-2010-7)*, 2010.

R. Salakhutdinov. *Learning Deep Generative Models*. PhD thesis, Dept. of Computer Science, University of Toronto, Sep 2009.

R. Salakhutdinov and I. Murray. On the quantitative analysis of deep belief networks. *Proceedings of the 25th International Conference on Machine Learning*, 25, Apr 2008.

O. G. Selfridge. Pandemonium: A paradigm for learning. *Mechanisation of thought processes: Proceedings of a symposium held at the National Physical Laboratory*, pages 115–122, 1958.

P. Smolensky. Information processing in dynamical systems: Foundations of harmony theory. *Parallel Distributed Processing: Explorations in the Microstructure of Cognition*, 1:194–281, Jan 1986.

J. Sohl-Dickstein, P. Battaglino, and M. R. DeWeese. Minimum probability flow learning. *pre-print*, 2009. arXiv:0906.4779.

J. M. Susskind, G. E. Hinton, J. R. Movellan, and A. K. Anderson. Generating facial expressions with deep belief nets. *Affective Computing, Emotion Modelling, Synthesis and Recognition*, pages 421–440, 2008.

I. Sutskever and G. E. Hinton. Deep, narrow sigmoid belief networks are universal approximators. *Neural Computation*, 20(11):2629–2636, Nov 2008.

G. W. Taylor, G. E. Hinton, and S. Roweis. Modeling human motion using binary latent variables. *Advances in Neural Information Processing Systems 19*, 2007.

T. Tieleman. Training restricted boltzmann machines using approximations to the likelihood gradient. *Proceedings of the 25th International Conference on Machine Learning*, 2008.

J. H. van Hateren and A. van der Schaaf. Independent component filters of natural images compared with simple cells in primary visual cortex. *Proceedings of the Royal Society B: Biological Sciences*, 265(1394), Mar 1998.

M. Welling and G. E. Hinton. A new learning algorithm for mean field boltzmann machines. *International Joint Conference on Neural Networks*, 2002.

M. Welling, M. Rosen-Zvi, and G. E. Hinton. Exponential family harmoniums with an application to information retrieval. *Advances in Neural Information Processing Systems 17*, 2005.

L. Younes. Parametric inference for imperfectly observed gibbsian fields. *Probability Theory and Related Fields*, 1989.

Journal of Machine Learning Research 12 (2011) 3097-3118 Submitted 10/10; Revised 7/11; Published 11/11

Efficient and Effective Visual Codebook Generation Using Additive Kernels

Jianxin Wu JXWU@NTU.EDU.SG
Wei-Chian Tan SGWEICHIAN@GMAIL.COM
School of Computer Engineering
Nanyang Technological University
Singapore, 639798, Singapore

James M. Rehg REHG@CC.GATECH.EDU
Center for Behavior Imaging
School of Interactive Computing
Georgia Institute of Technology
Atlanta, GA 30332, USA

Editor: Ben Taskar

Abstract

Common visual codebook generation methods used in a bag of visual words model, for example, k-means or Gaussian Mixture Model, use the Euclidean distance to cluster features into visual code words. However, most popular visual descriptors are histograms of image measurements. It has been shown that with histogram features, the Histogram Intersection Kernel (HIK) is more effective than the Euclidean distance in supervised learning tasks. In this paper, we demonstrate that HIK can be used in an unsupervised manner to significantly improve the generation of visual codebooks. We propose a histogram kernel k-means algorithm which is easy to implement and runs almost as fast as the standard k-means. The HIK codebooks have consistently higher recognition accuracy over k-means codebooks by 2–4% in several benchmark object and scene recognition data sets. The algorithm is also generalized to arbitrary additive kernels. Its speed is thousands of times faster than a naive implementation of the kernel k-means algorithm. In addition, we propose a one-class SVM formulation to create more effective visual code words. Finally, we show that the standard k-median clustering method can be used for visual codebook generation and can act as a compromise between the HIK / additive kernel and the k-means approaches.

Keywords: visual codebook, additive kernel, histogram intersection kernel

1. Introduction

Bag of visual words (BOV) is currently a popular approach to object and scene recognition in computer vision. Local features are extracted from an image, and the image is then considered as a *bag of features*, that is, completely ignoring the spatial relationship among features. Probably due to the lack of an efficient and effective mechanism to encode spatial information among features, BOV is widely adopted in vision tasks. A typical BOV-based method consists of the following stages:

- **Extract features.** Visual features and their corresponding descriptors are extracted from local image patches. Two typical visual descriptors are SIFT by Lowe (2004) and HOG by Dalal and Triggs (2005). Usually two ways are used to determine where to extract local features. Some methods extract features at certain detected interest points. Other methods

densely sample local features in a regular grid of pixel locations, for example, in Lazebnik et al. (2006). Visual descriptors extracted from these local patches are considered as feature vectors that describe these local regions.

- **Generate a codebook and map features to visual code words.** A visual codebook is a method that divides the space of visual descriptors into several regions. Features in one region correspond to the same *visual code word*, which is represented by an integer between 1 and the size of the codebook. An image is then encoded as a histogram of visual code words.

- **Learn and test.** Various machine learning methods can be applied to the histogram representation of images. For example, SVM is a frequently used learner in BOV models for object and scene recognition.

The quality of the visual codebook has a significant impact on the success of BOV-based methods. Popular and successful methods for object and scene categorization typically employ unsupervised learning methods (for example, k-means clustering or Gaussian Mixture Model) to obtain a visual codebook. When there is a need to compute the dissimilarity of two feature vectors, the Euclidean distance is the most frequently used metric.

However, in spite of its mathematical simplicity and efficacy in many other applications, we find that the Euclidean distance is not the most suitable similarity (or dissimilarity) measure for creating visual codebooks.

Two observations support our argument. First, most of the popular visual descriptors are based on histograms of various image measurements such as spatial gradients, optical flow, or color. For example, both SIFT and HOG use histograms of pixel intensity gradients in their descriptors. Second, for the case of supervised classification, it has been shown that the ℓ_2 distance is not the most effective metric for comparing two histograms, for example, in Maji et al. (2008). In particular, the Histogram Intersection Kernel (HIK) was demonstrated to give significantly improved results. Other similarity measures designed for comparing histograms, for example, the χ^2 measure, have also exhibited higher accuracy in SVM classification than the dot-product kernel (which corresponds to the Euclidean distance). One common characteristic of HIK and χ^2 is that they are instances in a family of kernels called the additive kernel (Maji and Berg, 2009; Vedaldi and Zisserman, 2010).

In this paper we demonstrate that HIK and other additive kernels can be used to generate better visual codebooks with unsupervised learning, in comparison to the popular k-means clustering method. The proposed methods are simple to implement, and our software implementations are freely available. We show that using roughly twice the computational time of the standard k-means based method (which uses the ℓ_2 distance), we can gain consistent accuracy improvements of 2–4% across a diverse collection of object and scene recognition problems. Specifically, this paper makes four contributions:

First, we show that HIK generates better codebooks and thus improves recognition accuracy. We generalize and speedup the method in Maji et al. (2008), such that the generation and application of HIK codebook has the same theoretical complexity as the standard k-means. Our proposed method achieves consistent performance improvements over k-means codebooks, and has established state-of-the-art performance numbers for four benchmark object and scene recognition data sets. We also show that a one-class SVM formulation can be used to improve the effectiveness of HIK codebooks, by providing well-separated, compact clusters in the histogram feature space.

Second, we show that all additive kernels can be used to efficiently generate visual codebooks. The HIK visual codebook creation method is also generalized to be compatible with any additive

kernel, while the learning cost remains unchanged as $O(nmd)$, where n, m, and d are the number of features to be clustered, the codebook size, and the feature dimension, respectively. In contrast, a naive implementation has complexity $O(n^2d)$. Since $n \gg m$, in practice the speedup is more than three orders of magnitude. Similar recognition accuracies are observed for different additive kernels, including HIK and χ^2. More generally, we suggest that *HIK (or other additive kernel such as χ^2) should be used whenever two histograms are compared*.

Third, we empirically show that k-median is a compromise between k-means and additive kernel codebooks. K-median's performance is consistently lower than the proposed HIK codebook, but better than k-means in most cases. On the other hand, it runs as fast as the proposed method, and also uses less storage.

Finally, we validate our method through experiments on standard data sets, using both the SIFT feature and CENTRIST, a recently proposed feature based on CENsus TRansform hISTogram (Wu and Rehg, 2011), which has been shown to offer performance advantages for scene classification.

The rest of the paper is organized as follows.[1] Related methods are discussed in Section 2. Section 3 introduces the histogram intersection kernel and various other additive kernels, and presents a general kernel k-means based method for visual codebook generation. In Section 4 we propose methods to efficiently generate visual codebooks for HIK and other additive kernels. Experiments are shown in Section 5, and Section 6 concludes this paper.

2. Related Works

In this section we will briefly review two categories of related works: different kernels for comparing histograms, and various visual codebook generation methods.

The main point of this paper is that when histogram features are employed, the histogram intersection kernel or another additive kernel should be used to compare them. HIK was introduced by Swain and Ballard (1991) for color-based object recognition. Odone et al. (2005) demonstrated that HIK forms a positive definite kernel when feature values are non-negative integers, facilitating its use in SVM classifiers. Simultaneously, works such as Lowe (2004) and Dalal and Triggs (2005) demonstrated the value of histogram features for a variety of tasks. However, the high computational cost of HIK at run-time remained a barrier to its use in practice. This barrier was removed for the case of SVM classifiers by various recent research works (Maji et al., 2008; Wu, 2010; Vedaldi and Zisserman, 2010), based on techniques to accelerate the kernel evaluations. Additive kernels, which include HIK as one of its instances, have also shown excellent performance in SVM classification of histograms (Vedaldi and Zisserman, 2010).

In this paper, we extend the results of Maji et al. (2008) in two ways: First, we demonstrate that the speedup of HIK can be extended to codebook generation (and unsupervised learning in general). Second, our Algorithm 2 provides an exact $O(d)$ method, which makes it possible to obtain the maximum efficiency without the loss of accuracy.

On the visual codebook side, k-means is the most widely used method for visual codebook generation (Sivic and Zisserman, 2003). However, several alternative strategies have been explored. K-means usually positions its clusters almost exclusively around the densest regions. A mean-shift type clustering method was used to overcome this drawback in Jurie and Triggs (2005). There are also information theoretic methods that try to capture the "semantic" common visual components by minimizing information loss (Liu and Shah, 2007; Lazebnik and Raginsky, 2009). An extreme

1. A preliminary version of portions of this work has been published in Wu and Rehg (2009).

method was presented in Tuytelaars and Schmid (2007), which divided the space of visual descriptors into regular lattice instead of learning a division of the space from training data. There are also efforts to build hash functions (that is, multiple binary functions / hash bits) in order to accelerate distance computations (Weiss et al., 2009). Recently, sparse coding is also used to vector quantize visual descriptors, for example, in Yang et al. (2009) and Gao et al. (2010). In this work, we propose a new alternative to k-means, based on the histogram intersection kernel and other additive kernels.

In k-means based methods, a visual code word is usually represented by the cluster center (that is, the average of all features that belong to this code word), which is simple and fast to compute. It was discovered that assigning a feature to multiple code words (which is also termed as soft-assignment) may improve the codebook quality (Philbin et al., 2008; van Gemert et al., 2008). Within a probabilistic framework, code words can be represented by the Gaussian Mixture Model (GMM) (Perronnin, 2008; Winn et al., 2005). GMM has better representation power than a single cluster center. However, it requires more computational power. Another interesting representation is the hyperfeature in Agarwal and Triggs (2008), which considers the mapped code word indexes as a type of image feature and repeatedly generates new codebooks and code words into a hierarchy.

Methods have been proposed to accelerate the space division and code word mapping. Nistér and Stewénius (2006) used a tree structure to divide the space of visual descriptors hierarchically and Moosmann et al. (2008) used ensembles of randomly created cluster trees. Both methods map visual features to code words much faster than k-means.

Some methods do not follow the *divide then represent* pattern. For example, Yang et al. (2008) unified the codebook generation step with the classifier learning step seamlessly. In another interesting research work, Vogel and Schiele (2007) manually specified a few code words and used supervised learning to learn these concepts from manually labeled examples.

It is worth noting that all of these previous methods used the ℓ_2 distance metric (except Gao et al., 2010 which followed our previous work Wu and Rehg, 2009, and used HIK). They could therefore in principle be improved through the use of HIK or other additive kernels.

3. Visual Codebook for Additive Kernels

In this section we will first introduce the Histogram Intersection Kernel (HIK), and then its generalization to the additive kernel case. In order to make our presentation clearer, we will use boldface characters (for example, $\mathbf{x}$) to represent vectors. The scalar x_j is the j-th dimension of $\mathbf{x}$.

3.1 Histogram Intersection Kernel

Let $\mathbf{x} = (x_1, \ldots, x_d) \in \mathbb{R}_+^d$ be a histogram of non-negative real values with d histogram bins, where $\mathbb{R}_+$ is the set of non-negative real numbers. $\mathbf{x}$ could represent an image (for example, a histogram of visual code words in the bag of visual words model) or an image patch (for example, a SIFT visual descriptor). The histogram intersection kernel κ_{HI} is defined as follows (Swain and Ballard, 1991):

$$\kappa_{\mathrm{HI}}(\mathbf{x}_1, \mathbf{x}_2) = \sum_{j=1}^{d} \min(x_{1,j}, x_{2,j}) \ . \tag{1}$$

It is proved in Wu (2010) that HIK is a valid positive definite kernel when the data $\mathbf{x} \in \mathbb{R}_+^d$. Thus there exists a mapping ϕ that maps any histogram $\mathbf{x}$ to a corresponding vector $\phi(\mathbf{x})$ in a high dimensional (possibly infinite dimensional) feature space Φ, such that $\kappa_{\mathrm{HI}}(\mathbf{x}_1, \mathbf{x}_2) = \phi(\mathbf{x}_1)^T \phi(\mathbf{x}_2)$.

Through the nonlinear mapping ϕ, histogram similarity is equivalent to a dot product in the feature space Φ. Furthermore, when the feature values are non-negative integers, that is, $\mathbf{x} \in \mathbb{N}^d$, we can explicitly find the mapping $\phi(\cdot)$. If we constrain the feature values to be bounded from above by v, that is, $0 \leq x_j \leq v$ for all $\mathbf{x}$ and $1 \leq j \leq d$, the mapping $\phi(\cdot)$ for HIK is then the following unary representation $B(\cdot)$ of an integer (Odone et al., 2005):

$$B(x) : x \mapsto \left[\underbrace{1 \cdots 1}_{x\ 1's} \underbrace{0 \cdots 0}_{v-x\ 0's} \right] .$$

It is easy to verify that $\kappa_{\mathrm{HI}}(\mathbf{x}_1, \mathbf{x}_2) = B(\mathbf{x}_1)^T B(\mathbf{x}_2)$, in which $B(\mathbf{x})$ is the concatenation of $B(x_1), B(x_2), \ldots, B(x_d)$. Note that $B(x_j) \in \mathbb{R}^v$ and $B(\mathbf{x}) \in \mathbb{R}^{dv}$.

This kernel trick makes it possible to use HIK in creating codebooks, while keeping the simplicity of k-means clustering. That is, we may use a kernel k-means algorithm (Schölkopf et al., 1998) to generate visual codebooks. In Algorithm 1, histograms are compared using HIK instead of the inappropriate Euclidean distance if we set $\phi(\cdot) = B(\cdot)$.[2]

When the data points $\mathbf{x} \in \mathbb{R}^d$, that is, allowing negative feature values, HIK is not a positive definite kernel. And it can not be used in Algorithm 1 to generate visual codebooks.[3]

3.2 Additive Kernels

Algorithm 1 is not restricted to work only with the histogram intersection kernel. It is a general kernel k-means algorithm which can be used together with any positive definite kernel. In particular, we are interested in a family of kernels called the *additive kernels* (Maji and Berg, 2009). Algorithm 1 instantiated with an additive kernel can be greatly accelerated, for which we will present in Section 4.

An additive kernel is a positive definite kernel that can be expressed in the following form

$$\kappa(\mathbf{x}_1, \mathbf{x}_2) = \sum_{j=1}^{d} \hat{\kappa}(x_{1,j}, x_{2,j}) .$$

A positive semidefinite function $\hat{\kappa}(\cdot, \cdot)$ is used to compute the similarity of two scalar values. An additive kernel κ then compares two vectors by comparing and summing up every dimension of these two vectors using $\hat{\kappa}$.

It is obvious that the histogram intersection kernel is an instance of the additive kernels. In fact, a family of additive kernels can be derived from HIK. If $g(\cdot)$ is a non-negative and non-decreasing function, then the generalized histogram intersection kernel,

$$\kappa(\mathbf{x}_1, \mathbf{x}_2) = \sum_{j=1}^{d} g(\min(x_{1,j}, x_{2,j})) ,$$

is a valid additive kernel. HIK corresponds to $g(x) = x, x \geq 0$. The GHI kernel proposed in Boughorbel et al. (2005) is also an instance of this family with $g(x) = x^\beta$ for $\beta > 0$ and $x \geq 0$. In this paper

2. Note that since *k-means++* is used in Algorithm 1 and it is a randomized algorithm, two runs of Algorithm 1 with the same input will possibly generate different results.
3. HIK is a conditionally positive definite kernel when $\mathbf{x} \in \mathbb{R}^d$ (Maji and Berg, 2009). It can still be used in some SVM solvers.

Algorithm 1 Codebook Generation Using Kernel k-means

1: **Input**: $\mathbf{x}_1, \ldots, \mathbf{x}_n \in \mathbb{R}_+^d$ (n input histograms), m (size of the codebook), and ε (tolerance).

2: {The output is a function that maps a histogram to its visual code word index, $w_1(\mathbf{x}_*) : \mathbb{R}_+^d \to \{1, \ldots, m\}$.}

3: $t \leftarrow 0$, {Initialize t, the iteration counter, to 0.}
$\varepsilon^t \leftarrow \infty$. {Initialize the current clustering error to ∞.}

4: Initialize the visual codebook. First, use the *k-means++* method (Arthur and Vassilvitskii, 2007) to choose m distinct examples from the input set $\{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$. We denote these examples as $\mathbf{x}_1, \ldots, \mathbf{x}_m$. Second, use $\mathbf{m}_i = \phi(\mathbf{x}_i)$, $i = 1, 2, \ldots, m$, as the initial visual code words. $\phi(\cdot)$ is the mapping associated with a positive definite kernel.

5: **repeat**

6: For every input histogram $\mathbf{x}_i$, find the visual code word that $\mathbf{x}_i$ belongs to, and denote the index of this visual code word as l_i:

$$l_i \leftarrow \arg\min_{1 \le j \le m} \|\phi(\mathbf{x}_i) - \mathbf{m}_j\|^2, \quad 1 \le i \le n.$$

7: For every visual code word $\mathbf{m}_i$, find the set of the input histograms that belong to this visual code word, and denote this set as π_i:

$$\pi_i = \{j | l_j = i, 1 \le j \le n\}, \quad 1 \le i \le m.$$

8: For every visual code word $\mathbf{m}_i$, update it to be the centroid of input histograms that belong to this visual code word:

$$\mathbf{m}_i \leftarrow \frac{\sum_{j \in \pi_i} \phi(\mathbf{x}_j)}{|\pi_i|}, \quad 1 \le i \le m.$$

9: Update the iteration counter and compute the current clustering error:

$$t \leftarrow t + 1,$$

$$\varepsilon^t = \frac{1}{n} \sum_{i=1}^{n} \|\phi(\mathbf{x}_i) - \mathbf{m}_{l_i}\|^2.$$

10: **until** $\varepsilon^{t-1} - \varepsilon^t \le \varepsilon$.

11: **Output**: For any histogram $\mathbf{x}_* \in \mathbb{R}_+^d$, its corresponding visual code word index is:

$$w_1(\mathbf{x}_*) = \arg\min_{1 \le i \le m} \|\phi(\mathbf{x}_*) - \mathbf{m}_i\|^2. \tag{2}$$

we will explore one specific instance from this family, which we call exponential HIK (or eHIK), defined as

$$\kappa_{\text{eHI}}(\mathbf{x}_1, \mathbf{x}_2) = \sum_{j=1}^{d} \min(e^{\gamma x_{1,j}}, e^{\gamma x_{2,j}}), \quad \gamma > 0 \ .$$

χ^2 is another additive kernel that has been used for comparing histograms. The original χ^2 measure is defined as $\chi^2(x_1, x_2) = \frac{(x_1 - x_2)^2}{x_1 + x_2}$ for $x_1, x_2 \in \mathbb{R}_+$. Alternatively, a variant of χ^2 is explored

in Vedaldi and Zisserman (2010) for SVM classification when the feature values are positive:

$$\kappa_{\chi^2}(\mathbf{x}_1,\mathbf{x}_2) = \sum_{j=1}^{d} \frac{2x_{1,j}x_{2,j}}{x_{1,j}+x_{2,j}} .$$

We will adopt this definition in our experiments.

3.3 K-median Codebook Generation

Although k-means (or equivalently, using the ℓ_2 distance) is the most popular codebook generation method, the histogram intersection kernel has a closer connection to the ℓ_1 distance. For two numbers a and b, it is easy to show that

$$2\min(a,b) + |a-b| = a+b .$$

As a consequence, we have

$$2\kappa_{\mathrm{HI}}(\mathbf{x}_1,\mathbf{x}_2) + \|\mathbf{x}_1-\mathbf{x}_2\|_1 = \|\mathbf{x}_1\|_1 + \|\mathbf{x}_2\|_1 ,$$

in which $\|\mathbf{x}\|_1$ is the ℓ_1 norm of $\mathbf{x}$. In cases when $\|\mathbf{x}\|_1$ is constant for any histogram $\mathbf{x}$, κ_{HI} and the ℓ_1 distance are linearly correlated.

For an array $x_1,\ldots,x_n$, it is well known that the value which minimizes the ℓ_1 error,

$$x_* = \arg\min_x \sum_{i=1}^{n} |x-x_i| ,$$

equals the median value of the array. Thus, k-median is a natural alternative for codebook generation. The only difference between k-median and k-means is that k-median uses ℓ_1 instead of ℓ_2 as the distance metric.

K-median has been less popular than k-means for the creation of visual codebooks. An online k-median algorithm has been used by Larlus and Jurie to create visual codebooks in the Pascal challenge (Everingham et al., 2006). In Section 5, we empirically compare visual codebooks generated by the k-median algorithm to those generated by both the k-means algorithm and the proposed additive kernel k-means method.

4. The Efficient Additive Kernel k-means Clustering Method

As mentioned in Section 3.2, additive kernels are attractive for kernel k-means because very fast clustering is possible for these kernels. In this section, we first propose an efficient kernel k-means algorithm for the histogram intersection kernel, and then generalize the algorithm to all additive kernels.

4.1 Common Computation Bottleneck

Given n examples in $\mathbb{R}^d$, the standard k-means clustering method (that is, $\phi(\mathbf{x}) = \mathbf{x}$ in Algorithm 1) requires $O(nmd)$ steps in one iteration (from line 5 to line 10). Similarly, the k-median algorithm also requires $O(nmd)$ steps in one iteration.

When $\phi(\mathbf{x}) \neq \mathbf{x}$, the centers $\mathbf{m}_i$ are vectors in the unrealized, high dimensional space Φ. $\mathbf{m}_i$ might even be infinite dimensional for some kernels (for example, the RBF kernel). The computations are

then carried out in the following way (using the usual kernel trick $\phi(\mathbf{x}_1)^T\phi(\mathbf{x}_2) = \kappa(\mathbf{x}_1, \mathbf{x}_2)$ such that $\mathbf{m}_i$ does not need to be explicitly generated):

$$\begin{aligned}
&\|\phi(\mathbf{x}_*) - \mathbf{m}_i\|^2 \\
=&\left\|\phi(\mathbf{x}_*) - \frac{\sum_{j\in\pi_i}\phi(\mathbf{x}_j)}{|\pi_i|}\right\|^2 \\
=&\|\phi(\mathbf{x}_*)\|^2 + \frac{1}{|\pi_i|^2}\sum_{j,k\in\pi_i}\kappa(\mathbf{x}_j,\mathbf{x}_k) - \frac{2}{|\pi_i|}\sum_{j\in\pi_i}\kappa(\mathbf{x}_*,\mathbf{x}_j). \qquad (3)
\end{aligned}$$

The first term in Equation 3 does not affect the result in lines 6 and 11 of Algorithm 1. The second term does not change within a specific iteration of Algorithm 1. Thus, we need to compute this term only once for every visual code word in each iteration. Most of the computations are then spent in computing the last term $\sum_{j\in\pi_i}\kappa(\mathbf{x}_*,\mathbf{x}_j)$.

A naive implementation to compute this term will be costly. For example, if we use the histogram intersection kernel and compute this term literally using Equation 1, the complexity is $O(|\pi_i|d)$. The complexity of line 6 in Algorithm 1 will be on the order

$$\sum_{i=1}^{n} m|\pi_{l_i}|d = \sum_{i=1}^{m}\left(\sum_{j:l_j=i} m|\pi_i|d\right) = \sum_{i=1}^{m} m|\pi_i|^2 d\ ,$$

since there are $|\pi_i|$ input histograms $\mathbf{x}_j$ satisfying $l_j = i$. Using the Cauchy-Schwarz inequality, it is clear that

$$\sum_{i=1}^{m}|\pi_i|^2 \geq \frac{1}{m}\left(\sum_{i=1}^{m}|\pi_i|\right)^2 = \frac{n^2}{m}\ ,$$

because $\sum_{i=1}^{m}|\pi_i| = n$. In practice, the sizes of π_i are usually similar for different i, and one iteration of this naive implementation will have complexity $O(n^2 d)$. We generally have $n \gg m$, thus a kernel k-means will be much more expensive than the standard k-means. In summary, the last term in Equation 3 is the bottleneck in the computations.

The form of this term, $\sum_{j\in\pi_i}\kappa(\mathbf{x}_*,\mathbf{x}_j)$, is similar to the binary SVM classifier, which has the following form:

$$\text{sign}\left(\sum_{i\in\pi}\alpha_i y_i \kappa(\mathbf{x}_*,\mathbf{x}_i) + \rho\right)\ , \qquad (4)$$

where $\mathbf{x}_i$, α_i, and y_i are, respectively, the support vectors, and their corresponding weights and labels.

Based on these observations, we propose a more general objective,

$$f(\mathbf{x}_*) = \sum_{i\in\pi} c_i \kappa(\mathbf{x}_*,\mathbf{x}_i)\ , \qquad (5)$$

where π indexes a set of histograms (data points to be clustered, or support vectors) and c_i are constant coefficients. Note that both Equation 4 and the last term in Equation 3 are special forms of Equation 5, with $c_i = \alpha_i y_i$ and $c_i = 1$, respectively.

Our goal is then to reduce the complexity of Equation 5 to $O(d)$ (the same complexity as that of standard k-means when $\phi(\mathbf{x}) = \mathbf{x}$), which will in turn yield efficient kernel k-means clustering and SVM testing methods. We will first present the algorithm for HIK, and then its generalization to arbitrary additive kernels.

4.2 Efficient Computation of HIK

Maji et al. (2008) proposed fast methods to compute Equation 4 for the histogram intersection kernel to improve the testing speed of HIK SVM classifiers, which achieved an exact answer of Equation 4 in $O(d\log_2|\pi|)$ steps and an approximate answer in $O(d)$ steps. In this paper we propose a variant that finds the exact answer for Equation 5 in $O(d)$ steps when the feature values are non-negative integers.

A histogram of visual code word indexes has the property that every histogram component is a non-negative integer, that is, it is a vector in $\mathbb{N}^d$. Similarly, a visual descriptor histogram can usually be transformed into the space $\mathbb{N}^d$. For example, the SIFT descriptors are stored as vectors in $\mathbb{N}^{128}$. In general, a vector in $\mathbb{R}^d_+$ can be transformed into $\mathbb{N}^d$ by a linear transformation followed by quantization.

In the rest of this paper, we assume that any histogram $\mathbf{x} = (x_1, \ldots, x_d)$ satisfies that $x_i \in \mathbb{N}$ and $0 \leq x_i \leq v$ for all i. Then the quantity $f(\mathbf{x}_*)$ can be computed as follows:

$$\begin{aligned} f(\mathbf{x}_*) &= \sum_{i\in\pi} c_i \kappa_{\mathrm{HI}}(\mathbf{x}_*, \mathbf{x}_i) \\ &= \sum_{i\in\pi} \sum_{1\leq j\leq d} c_i \min(x_{*,j}, x_{i,j}) \\ &= \sum_{1\leq j\leq d} \left(\sum_{i\in\pi} c_i \min(x_{*,j}, x_{i,j}) \right) \\ &= \sum_{1\leq j\leq d} \left(\sum_{i:x_{*,j}\geq x_{i,j}} c_i x_{i,j} + x_{*,j} \sum_{i:x_{*,j}<x_{i,j}} c_i \right). \end{aligned} \tag{6}$$

Note that the two summands in Equation 6 can both be pre-computed. It is shown in Maji et al. (2008) that Equation 6 is a piece-wise linear function of $x_{*,j}$. Thus using a binary search for $x_{*,j}$, Equation 6 can be computed in $O(d\log|\pi|)$ steps in Maji et al. (2008).

However, since we assume that $x_{*,j}$ is an integer in the range $[0\ v]$, we have an even faster method. Different dimensions of $\mathbf{x}_*$ make independent contributions to $f(\mathbf{x}_*)$ in Equation 6, because of the additive property. Thus it is sufficient to solve the problem for one single feature dimension at a time. And because there are only $v+1$ possibilities for $x_{*,j}$ given a fixed j, we just need to pre-compute the solutions for these $v+1$ values. Let T be a table of size dv, with

$$T(j,k) \leftarrow \sum_{i:k\geq x_{i,j}} c_i x_{i,j} + k \sum_{i:k<x_{i,j}} c_i$$

for all $1 \leq j \leq d$ and $1 \leq k \leq v$. Then it is clear that

$$f(\mathbf{x}_*) = \sum_{j=1}^{d} T(j, x_{*,j}) \,. \tag{7}$$

This method is summarized in Algorithm 2. Note that since $T(j,0) = 0$ for all j, there is no need to store it.

It is obvious that $f(\mathbf{x}_*)$ can be evaluated in $O(d)$ steps after the table T is pre-computed. And because Algorithm 2 only involves table lookup and summation, it is faster (that is, has less overhead) than the approximation scheme in Maji et al. (2008), which is also $O(d)$. Depending on the

Algorithm 2 Fast Computation of HIK Sums

1: **Input**: n histograms $\mathbf{x}_1,\ldots,\mathbf{x}_n$ in $\mathbb{N}^d$, with $0 \le x_{i,j} \le v$ for $1 \le i \le n$ and $1 \le j \le d$.
2: {The output is a fast method to compute

$$f(\mathbf{x}_*) = \sum_{i=1}^{n} c_i \kappa_{\mathrm{HI}}(\mathbf{x}_*, \mathbf{x}_i) \ ,$$

where $\mathbf{x}_* \in \mathbb{N}^d$ and $0 \le x_{*,j} \le v, \forall\ 1 \le j \le d$.}
3: Create T, a $d \times v$ table.
4: For $1 \le j \le d$, $1 \le k \le v$,

$$T(j,k) \leftarrow \sum_{i:k \ge x_{i,j}} c_i x_{i,j} + k \sum_{i:k < x_{i,j}} c_i \ .$$

5: **Output**:

$$f(\mathbf{x}_*) = \sum_{j=1}^{d} T(j, x_{*j}) \ .$$

relative size of v and the number of approximation bins used in Maji et al. (2008), Algorithm 2's storage requirement, $O(vd)$, could be larger or smaller than that of Maji et al. (2008). It is also worth noting that under our assumptions, Algorithm 2's result is precise rather than approximate.

Both the complexity of the pre-computation and the storage requirement are linear in v, which is a parameter specified by users.[4] Our experiments show that while too small a v usually produces inferior results, a large v does not necessarily improve performance. In this paper, we choose $v = 128$, which seems to give the best results in our experiments.

Our algorithm has the same computational complexity as the standard k-means when generating a visual codebook or mapping histograms to visual code word indexes (that is, Equation 2 or Equation 3). In practice, the proposed method takes about twice the time of k-means. In summary, the proposed method generates a visual codebook that can not only run almost as fast as the k-means method, but also can use the non-linear similarity measure κ_{HI} that is most suitable for comparing histograms.

4.3 Generalization to Additive Kernels

Algorithm 2 can be generalized from HIK to arbitrary additive kernels. The following two conditions are also satisfied by all additive kernels: different dimensions of $\mathbf{x}_*$ make independent contributions to $f(\mathbf{x}_*)$; and there are only $v+1$ possibilities for $x_{*,j}$ when j is fixed. We just need to find an appropriate value for $T(j,k)$, and Equation 7 is then valid for all additive kernels. Of course, we assume that the feature values are natural numbers bounded from above by v.

For an additive kernel

$$\kappa(\mathbf{x}_1, \mathbf{x}_2) = \sum_{j=1}^{d} \hat{\kappa}(x_{1,j}, x_{2,j}) \ ,$$

4. A simple implementation to pre-compute the table T takes $O(ndv)$ steps. We will present a $O(d(n+v))$ implementation of Algorithm 2 in Section 4.3.

we propose Algorithm 3 to efficiently assign values to the table T. Considering a fixed dimension j, by the independence property, we have

$$T(j,k) = \sum_{i=1}^{n} c_i \hat{\kappa}(x_{i,j},k) \; . \tag{8}$$

In general, it takes $O(ndv)$ steps to fill the table T for an additive kernel if we literally translate Equation 8. However, for additive kernels, Algorithm 3 uses a sequential update strategy whose complexity is only $O(d(n+v^2))$.

Algorithm 3 Assign values to T for an arbitrary additive kernel

Input: n histograms $\mathbf{x}_1, \ldots, \mathbf{x}_n$ in $\mathbb{N}^d$, with $0 \le x_{i,j} \le v$, for all $1 \le i \le n$ and $1 \le j \le d$; and an additive kernel $\kappa(\mathbf{x}_1, \mathbf{x}_2) = \sum_{j=1}^{d} \hat{\kappa}(x_{1,j}, x_{2,j})$.
{The output is a table $T \in \mathbb{R}^{dv}$ for fast computation of Equation 5, where $\mathbf{x}_* \in \mathbb{N}^d$ and $0 \le x_{*,j} \le v, \forall\, 1 \le j \le d$.}
for $j = 1, \ldots, d$ **do**
  Create a vector $\mathbf{h} \in \mathbb{R}^v$, and $\mathbf{h} \leftarrow \mathbf{0}$.
  for $i = 1, \ldots, n$ **do**
    $h_{x_{i,j}} \leftarrow h_{x_{i,j}} + c_i$.
  end for
  $T(j,0) = \sum_{i=1}^{n} \hat{\kappa}(x_{i,j}, 0)$.
  for $k = 1, \ldots, v$ **do**
    $T(j,k) \leftarrow T(j,k-1) + \sum_{v=0}^{v} h_v \left(\hat{\kappa}(v,k) - \hat{\kappa}(v,k-1) \right)$.
  end for
end for
Output: A table T such that

$$f(\mathbf{x}_*) = \sum_{j=1}^{d} T(j, x_{*j}) \; .$$

For a fixed feature dimension j,

$$\begin{aligned}
& T(j,k) - T(j,k-1) \\
&= \sum_{i=1}^{n} c_i \left(\hat{\kappa}(x_{i,j},k) - \hat{\kappa}(x_{i,j},k-1) \right) \\
&= \sum_{v=0}^{v} \left(\sum_{i:x_{i,j}=v} c_i \left(\hat{\kappa}(v,k) - \hat{\kappa}(v,k-1) \right) \right) \\
&= \sum_{v=0}^{v} \left(\left(\sum_{i:x_{i,j}=v} c_i \right) \left(\hat{\kappa}(v,k) - \hat{\kappa}(v,k-1) \right) \right) .
\end{aligned}$$

In Algorithm 3, we make a weighted histogram $\mathbf{h}$ for the j-th dimension such that $h_v = \sum_{i:x_{i,j}=v} c_i$. This is the first inner-loop, and its complexity is $O(n)$. It then takes $O(v^2)$ steps to sequentially update the j-th row of the table T. In total, Algorithm 3 takes $O(d(n+v^2))$ steps. Since in general $n \gg v$, Algorithm 3 is more efficient than a $O(ndv)$ method.

	k-means	Kernel k-means (naive)	Kernel k-means (proposed)
Storage	md	nd	mdv
Running time	$O(nmd)$	$O(n^2d)$	$O(nmd)$

Table 1: Space and running time requirements of the standard k-means, a naive implementation of kernel k-means, and the proposed method. The space requirement does not include the memory required to store the input histograms. The running time requirement shows the complexity of one kernel k-means iteration.

One difference between Algorithms 2 and 3 is that $T(j,0)$ may not be equal to 0 in Algorithm 3. A more important difference is that Algorithm 2 can be further improved to $O(d(n+v))$. Note that in Algorithm 2, $\hat{\kappa}(x,y) = \min(x,y)$. We then have $\hat{\kappa}(v,k) - \hat{\kappa}(v,k-1)$ equals 1 if $v > k-1$ and 0 if otherwise. In consequence, $T(j,k) - T(j,k-1) = \sum_{v=k}^{v} h_v$, which can in turn be sequentially updated and takes only $O(1)$ steps to compute for every k value. The complexity of Algorithm 2 is then $O(d(n+v))$.

In practice, we generally have $n \gg m$, $n \gg d$, and $n \gg v$. Typical values in our experiments are $n = 300,000$, $d = 128$ or $d = 256$, $m = 200$, and $v = 128$. The complexity of kernel k-means is then dominated by the line 6 of Algorithm 1. Space and running time requirements of various algorithms are summarized in Table 1. The naive implementation does not need additional storage during the visual codebook generation step. However, it needs to keep all input histograms (nd numbers) for the quantization step. The other two methods do not need to keep input histograms for quantization.

The theoretical complexities in Table 1 match the empirical running time in our experiments. For example, in one experiment using the Caltech 101 data set (refer to Section 5), the naive implementation and the proposed method took 2403 and 1.2 seconds, respectively. The empirical speedup ratio is 2000. In this experiment, the theoretical speedup ratio is $O(n/m)$, and $n/m \approx 1200$. Since the naive implementation is impractical for large-scale problems, we will not provide empirical results of this method in our experiments.[5]

4.4 One Class SVM Codebook Representation

A codebook generated by the k-means algorithm first divides the space $\mathbb{R}^d$ into m regions, and then represents each code word (or, region) by the centroid of the examples (histogram, feature vectors, etc.) that fall into this region. This approach is optimal if we assume that vectors in all regions follow Gaussian distributions with the same spherical covariance matrix (that is, only differ in their means).

This assumption rarely holds. Different regions usually have very different densities and covariance structures. Simply dividing the space $\mathbb{R}^d$ into a Voronoi diagram from the set of region centers is, in many cases, misleading. However, further refinements are usually computationally prohibitive. For example, if we model regions as Gaussian distributions with distinct covariance matrices, the generation of codebooks and mapping from visual features to code words will require much more storage and computational resources than we can afford.

5. Since one kernel k-means iteration takes more than 2400 seconds, it will take months to finish running all the experiments in Section 5.

We propose to use one-class SVM (Schölkopf et al., 2001) to represent the divided regions in an effective and computationally-efficient way. Given a set of histograms in a region $\mathbf{x}_\pi = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$, we construct a one-class SVM with parameter $\nu \in (0,1]$,

$$\text{sign}\left(\sum_{i\in\pi} \alpha_i \kappa(\mathbf{x}, \mathbf{x}_i) + \rho\right), \tag{9}$$

where α_i's are non-negative, sparse, and $\sum_i \alpha_i = 1$. Intuitively, a one-class SVM classifier seeks a "simple" (compact) subset of $\mathbf{x}_\pi$ (or the divided region) that retains a large portion of the histograms (or densities). It is proved that ν is the upper bound on the fraction of outliers (that is, on which Equation 9 are less than 0), and at the same time a lower bound on the fraction of support vectors (that is, $\alpha_i \neq 0$) (Schölkopf et al., 2001).

The one-class SVM summarizes the distribution of histograms inside a visual code word. It takes into consideration the shape and density of the histogram distribution. It seeks to include most of the histograms (at least $(1-\nu)|\pi|$) in a compact hypersphere in the feature space, while paying less attention to those borderline cases (at most $\nu|\pi|$ examples). We believe that this compact hypersphere better summarizes a visual code word.

At the same time, these new code words can be computed very efficiently. Equation 9 is evaluated in $O(d)$ steps because it is again a special case of Algorithm 2. We propose Algorithm 4 to use one-class SVM to generate visual code words. Note that we use the space $\mathbb{R}^d$ because Algorithm 4 is not restricted to $\mathbb{N}^d$. In this paper, we set the parameter $\nu = 0.2$.

Algorithm 4 One-class SVM Code Word Generation

1: **Input**: Same as that of Algorithm 1.

2: Use Algorithm 1 to generate the divisions π_i ($i = 1, \ldots, m$) from the input histograms $\mathbf{x}_1, \ldots, \mathbf{x}_n$ in $\mathbb{R}^d_+$.

3: For each division $1 \leq i \leq m$, train a one-class SVM from its data $\mathbf{x}_{\pi_i}$ with a parameter ν,

$$w_2^i(\mathbf{x}_*) = \sum_{j\in\pi_i} \alpha_j \kappa(\mathbf{x}_*, \mathbf{x}_j) + \rho_i \,. \tag{10}$$

4: **Output**: For any histogram $\mathbf{x}_* \in \mathbb{R}^d_+$,

$$w_2(\mathbf{x}_*) = \arg\max_{1\leq i\leq m} w_2^i(\mathbf{x}_*) \,.$$

In many applications, a histogram $\mathbf{x} = (x_1, \ldots, x_d)$ satisfies the condition that $\|\mathbf{x}\|_1 = \sum_{j=1}^d x_j = N$ is a constant. Under this condition, Equation 10 is equivalent to

$$w_2^i(\mathbf{x}_*) = r_i^2 - \|\phi(\mathbf{x}_*) - \mathbf{m}_i\|^2 \,,$$

where $\mathbf{m}_i = \sum_{j\in\pi_i} \alpha_j \mathbf{x}_j$ and $r_i^2 = N + \|\mathbf{m}_i\|^2 - 2\rho_i$. In other words, a histogram is considered as belonging to the i-th visual word if it is inside the sphere (in the feature space Φ) centered at $\mathbf{m}_i$ with radius r_i. A sphere in Φ is different from a usual k-means sphere because it respects the similarity measure κ, and its radius r_i automatically adapts to the distribution of histograms in a visual word. Note that different kernels such as the dot-product kernel or κ_{HI} can be used in Algorithm 4.

5. Experiments

We validate the proposed methods using four benchmark data sets in computer vision: the Caltech 101 object recognition data set (Fei-Fei et al., 2004), the 15 class scene recognition data set (Lazebnik et al., 2006), the 8 class sports events data set (Li and Fei-Fei, 2007), and the 67 class indoor data set (Quattoni and Torralba, 2009).

5.1 Setup

In each data set, the available data is randomly split into a training set and a testing set based on published protocols on these data sets. The random splitting is repeated 5 times, and the average accuracy is reported. In each train/test splitting, a visual codebook is generated using the training images, and both training and testing images are transformed into histograms of code words. Accuracy is computed as the mean accuracy of all categories (that is, the average of diagonal entries in the confusion matrix).

The proposed algorithms can efficiently process a huge number of histogram features, for example, approximately 200k to 320k histograms are clustered across the first three data sets in less than 6 minute. In the 67 class indoor data set, more than 1 million histograms are clustered.

In the BOV model, we use 16×16 image patches and densely sample features over a grid with a spacing of 2, 4, or 8 pixels. We use two types of visual descriptors: SIFT for Caltech 101, CENTRIST (CENsus TRansform hISTogram, refer to Wu and Rehg (2011) for more details) for the scene, event, and indoor data sets.[6] All feature vectors are scaled and rounded such that a histogram only contains non-negative integers that approximately sum to 128 (thus $v = 128$ is always valid.)

The first step is to use visual descriptors from the training images to form a visual codebook, in which we use $m = 200$ to generate 200 visual code words. Next, every feature is mapped to an integer (code word index) between 1 and m. Thus an image or image sub-window is represented by a histogram of code words in the specified image region. In order to incorporate spatial information, we use the spatial hierarchy in Wu and Rehg (2008). An image is represented by the concatenation of histograms from all the 31 sub-windows, which is a 6200 dimensional histogram. To capture the edge information, we sometimes use Sobel gradients of an input image as an additional input, and concatenate histograms from the original input and the Sobel gradient image (which is 12400 dimensional). Following Boiman et al. (2008), we also sample features at 5 scales.

SVM is used for classification. LIBSVM (Chang and Lin, 2001) is used for the scene and sports data set. Since LIBSVM uses the 1-vs-1 strategy, it will produce too many classifiers for the Caltech 101 and indoor data set (more than 5000 and 2200 respectively). Therefore we instead use the Crammer & Singer formulation in BSVM (Hsu and Lin, 2006) for these two data sets. Since we are classifying histograms, we modified both LIBSVM and BSVM so that they are able to use the histogram intersection kernel.[7] It is observed that HIK is robust to the C parameter in SVM. For example, using the LIBSVM solver, classification accuracy remains almost unchanged after $C > 0.001$, as empirically showed in Wu (2010). Thus we do not use cross-validation to choose a different C value for every different training set. Instead, we use cross-validation to find $C = 2$ and

6. We will also evaluate the effect when these two feature types are switched in these data sets.

7. The methods proposed in this paper are publicly available in the libHIK package, which can be downloaded from `http://c2inet.sce.ntu.edu.sg/Jianxin/projects/libHIK/libHIK.htm`. The modified version of LIBSVM and BSVM are also included in libHIK.

$C = 0.03125$ for LIBSVM and BSVM respectively on a sample training set. These C values are then used on all data sets for LIBSVM and BSVM, respectively.

5.2 Main Results

We conducted several sets of experiments to validate the proposed algorithms. Experimental results are organized using the following rule: texts in the italic type summarize findings from one set of experiments and details are described after the italic texts. Mean / standard deviation values and paired t-tests are used to show the benefit of histogram kernel codebooks (Algorithm 1), while the Wilcoxon test is used for evaluating the one-class SVM code word generation method (Algorithm 4). We first present the main results, which are based on the experimental results summarized in Table 2.

In Table 2, sub-tables (a), (b), (c), and (d) are results for the Caltech 101, 15 class scene, 8 class sports, and the 67 class indoor data sets, respectively. κ_{HI} and κ_{LIN} means that a histogram intersection or a linear kernel visual codebook is used, respectively. oc_{svm} and $\neg oc_{\mathrm{svm}}$ indicate whether one-class SVM is used in generating code words. B and $\neg B$ indicate whether Sobel images are concatenated or not. And $s = 4$ or $s = 8$ is the grid step size when densely sampling features. The number of training/testing images in each category are indicated in the sub-table captions, which follows the protocol of previously published results on these data sets.

Histogram Intersection Kernel Visual Codebook (Algorithm 1) greatly improves classification accuracy. We compare the classification accuracies of systems that use Algorithm 1 with κ_{HI}, the standard k-means algorithm (that is, using κ_{LIN}), and k-median. From the experimental results in Table 2, it is obvious that in all four data sets, the classification accuracy with a κ_{HI}-based codebook is consistently higher than that with a k-means codebook. Using a paired t-test with significance level 0.05, the differences are statistically significant in 21 out of the 24 cases in Table 2, when comparing κ_{HI} and κ_{LIN} based codebooks. The three exceptions all come from the 8 class sports event data set, when one-class SVM is not used (that is, comparing the second row to the fifth row in Table 2c). HIK codebooks also have advantages over k-median codebooks in most cases.

HIK codebook can be computed efficiently (Algorithm 2). We have shown that Algorithm 2 evaluates in $O(d)$ steps, in the same order as k-means. Empirically, the κ_{HI}-based method spent less than 2 times CPU cycles than that of k-means. For example, the proposed method took 105 seconds to generate a codebook for the Caltech 101 data set, while k-means used 56 seconds in our experiments.

One-class SVM improves histogram intersection kernel code words (Algorithm 4). The t-test is not powerful enough here, because we have only 5 paired samples and they are not necessarily normally distributed. The Wilcoxon signed-rank test is more appropriate (Demšar, 2006) to show the effect of Algorithm 4. Algorithm 4 improved the classification accuracy of the κ_{HI}-based method in 11 out of 12 cases in Table 2. The Wilcoxon test shows that the difference is significant at significance level 0.01.

In summary, using HIK codebooks and one-class SVM together generated the best results in almost all cases (best results are shown in boldface within each column of Table 2).

One-Class SVM degrades the standard k-means code words. It is interesting to observe a completely reversed trend when κ_{LIN} is used with one-class SVM. Applying Algorithm 4 in the standard k-means method reduced accuracy in all cases. Since a vector in $\mathbb{R}^d$ is not an appropriate understanding of a histogram with d bins, we conjecture that Algorithm 4 with κ_{LIN} produced a better division of the space $\mathbb{R}^d$, but probably a worse one in the space of histograms.

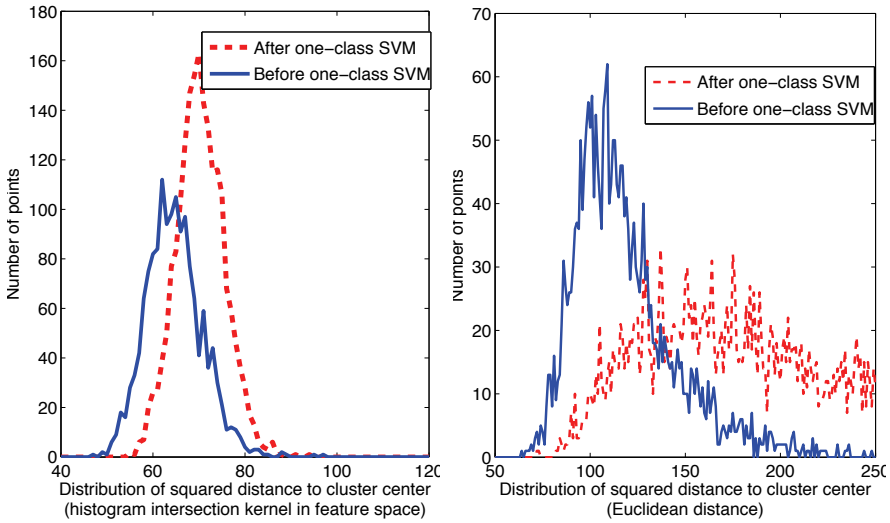


Figure 1: Effects of one-class SVM.

Figure 1 shows the effect of applying Algorithm 4 to example code words. The distribution of squared distance to cluster center becomes more compact in case of κ_{HI} with a minor increase in the average error. However, in the k-means case, the distances spread to larger values.

K-median is a compromise between k-means and HIK codebooks. As shown in Table 2, HIK codebooks outperformed k-median codebooks in most cases.[8] However, k-median generally outperformed the popular k-means codebooks. Furthermore, k-median requires less memory than the proposed method. Qualitative comparisons of these methods are summarized in Table 3.

5.3 Experimental Results for Additive Kernels

Experiments with codebooks generated using the other two additive kernels (χ^2 and exponential HIK) are shown in Table 4. For ease of comparison, results of HIK (without one-class SVM) codebooks are also shown in Table 4.

HIK and χ^2 based codebooks have very similar accuracies, and both outperform the exponential HIK codebooks. However, all three additive kernel based codebooks generally have higher accuracies than the standard k-means codebook generation method. Since the time complexity of additive kernel based codebooks is the same as that of the k-means method, it is advantageous to apply such kernels in generating visual codebooks. For example, the χ^2 kernel in some cases leads to higher accuracies than the histogram intersection kernel.

5.4 Effects of Information Content

Next we study the effects of using different types and amounts of information, for example, different types of base features and step sizes in dense feature sampling.

8. There is not an obvious kernel for the ℓ_1 distance, so we did not use one-class SVM for codebooks generated by k-median.

	$B, s = 4$	$B, s = 8$	$\neg B, s = 8$
$\kappa_{\mathrm{HI}}, oc_{\mathrm{svm}}$	**67.44±0.95%**	**65.20±0.91%**	**61.00±0.90%**
$\kappa_{\mathrm{HI}}, \neg oc_{\mathrm{svm}}$	66.54±0.58%	64.11±0.84%	60.33±0.95%
k-median	66.38±0.79%	63.65±0.94%	59.64±1.03%
$\kappa_{\mathrm{LIN}}, oc_{\mathrm{svm}}$	62.69±0.80%	60.09±0.92%	56.31±1.13%
$\kappa_{\mathrm{LIN}}, \neg oc_{\mathrm{svm}}$	64.39±0.92%	61.20±0.95%	57.74±0.70%

(a) Caltech 101, 15 train, 20 test

	$B, s = 4$	$B, s = 8$	$\neg B, s = 8$
$\kappa_{\mathrm{HI}}, oc_{\mathrm{svm}}$	**84.12±0.52%**	**84.00±0.46%**	**82.02±0.54%**
$\kappa_{\mathrm{HI}}, \neg oc_{\mathrm{svm}}$	83.59±0.45%	83.74±0.42%	81.77±0.49%
k-median	83.04±0.61%	82.70±0.42%	80.98±0.50%
$\kappa_{\mathrm{LIN}}, oc_{\mathrm{svm}}$	79.84±0.78%	79.88±0.41%	77.00±0.80%
$\kappa_{\mathrm{LIN}}, \neg oc_{\mathrm{svm}}$	82.41±0.59%	82.31±0.60%	80.02±0.58%

(b) 15 class scene, 100 train, rest test

	$B, s = 4$	$B, s = 8$	$\neg B, s = 8$
$\kappa_{\mathrm{HI}}, oc_{\mathrm{svm}}$	**84.21±0.99%**	**83.54±1.13%**	81.33±1.56%
$\kappa_{\mathrm{HI}}, \neg oc_{\mathrm{svm}}$	83.17±1.01%	83.13±0.85%	**81.87±1.14%**
k-median	82.13±1.30%	81.71±1.30%	80.25±1.12%
$\kappa_{\mathrm{LIN}}, oc_{\mathrm{svm}}$	80.42±1.44%	79.42±1.51%	77.46±0.83%
$\kappa_{\mathrm{LIN}}, \neg oc_{\mathrm{svm}}$	82.54±0.86%	82.29±1.38%	81.42±0.76%

(c) 8 class sports, 70 train, 60 test

	$B, s = 4$	$B, s = 8$	$\neg B, s = 8$
$\kappa_{\mathrm{HI}}, oc_{\mathrm{svm}}$	**43.01±0.81%**	**41.75±0.94%**	**35.09±1.04%**
$\kappa_{\mathrm{HI}}, \neg oc_{\mathrm{svm}}$	41.73±0.80%	40.07±0.27%	33.55±0.26%
k-median	41.81±1.11%	40.22±1.07%	34.04±1.56%
$\kappa_{\mathrm{LIN}}, oc_{\mathrm{svm}}$	35.94±1.14%	34.63±1.24%	28.69±1.04%
$\kappa_{\mathrm{LIN}}, \neg oc_{\mathrm{svm}}$	39.79±0.47%	38.28±0.39%	32.49±0.72%

(d) 67 class indoor, 80 train, 20 test

Table 2: Results of HIK, k-median and k-means codebooks and one-class SVM code words. The best result in each column is shown in **boldface**.

	HIK	k-median	k-means
Computation time	2	2	1
Codebook storage size	v	1	1

Table 3: Comparison of three codebook generation methods. k-means is used as a baseline, that is, a value '2' means approximately 200% of that of k-means.

	$B, s = 4$	$B, s = 8$	$\neg B, s = 8$
κ_{HI}	66.54±0.58%	64.11±0.84%	60.33±0.95%
κ_{χ^2}	**67.35±0.77%**	**64.31±1.28%**	**60.63±0.85%**
κ_{eHI}	66.18±0.72%	63.71±0.58%	57.13±0.89%

(a) Caltech 101, 15 train, 20 test

	$B, s = 4$	$B, s = 8$	$\neg B, s = 8$
κ_{HI}	83.59±0.45%	**83.74±0.42%**	**81.77±0.49%**
κ_{χ^2}	**83.67±0.42%**	83.56±0.51%	81.60±0.46%
κ_{eHI}	83.17±0.52%	82.78±0.51%	80.79±0.71%

(b) 15 class scene, 100 train, rest test

	$B, s = 4$	$B, s = 8$	$\neg B, s = 8$
κ_{HI}	83.17±1.01%	83.13±0.85%	**81.87±1.14%**
κ_{χ^2}	**83.54±1.01%**	**83.21±1.31%**	81.75±0.65%
κ_{eHI}	80.71±1.60%	80.67±1.81%	78.46±1.05%

(c) 8 class sports, 70 train, 60 test

	$B, s = 4$	$B, s = 8$	$\neg B, s = 8$
κ_{HI}	41.73±0.80%	40.07±0.27%	**33.55±0.26%**
κ_{χ^2}	4**1.85±1.03%**	**40.22±1.01%**	33.51±0.99%
κ_{eHI}	38.97±0.93%	37.04±1.12%	31.72±0.84%

(d) 67 class indoor, 80 train, 20 test

Table 4: Results of HIK, χ^2 and exponential HIK codebooks. One-class SVM code word generation is *not* used. The best result in each column is shown in **boldface**.

Caltech 101	15 scene	8 sports	67 indoor
53.25±0.80%	78.54±0.22%	81.17±0.65%	33.48±0.59%
61.00±0.90%	82.02±0.54%	81.33±1.56%	35.09±1.04%

Table 5: Results when features are sampled in only 1 image scale and 5 scales, respectively. HIK codebooks are used, with oc_{svm}, $\neg B$ and $s = 8$.

Sampling features at 5 scales improves accuracy. It is advantageous to sample features from multiple scaled versions of the input image. Also, Table 5 reinforces the conclusions from Section 5.2.

Smaller step size is better. Similarly, a smaller step size means that more features are sampled. Table 2 shows that when other conditions were the same, $s = 4$ outperformed $s = 8$ in general. We observed differences between object and scene recognition. The accuracy difference in Caltech 101 is significant. In the sports and indoor data set, $s = 4$ slightly outperformed $s = 8$ and they are indistinguishable in the 15 class scene data set. Thus it is not necessary to compute $s = 2$ results for

Caltech 101	15 scene	8 sports	67 indoor
60.99±0.67%	79.86±0.30%	82.33±0.74%	38.04±1.24%
65.20±0.91%	84.00±0.46%	83.54±1.13%	41.75±0.94%

Table 6: Results when feature type is switched. We use B, $s = 8$, and oc_{SVM}. The second row contains numbers extracted from Table 2, and the first row are results when feature type is switched.

the two scene recognition data sets. In Caltech 101, however, $s = 2$ further improved recognition accuracy to $67.82 \pm 0.59\%$ (using κ_{HI}, oc_{SVM}, and B.)

Use the right feature for different tasks. SIFT is widely used in object recognition for its performance. And CENTRIST has been shown as a suitable feature for place and scene recognition in Wu and Rehg (2011). As shown in Table 6, if we use SIFT for scene recognition and CENTRIST for object recognition, the recognition accuracies are reduced.

More code words are (sometimes) better. We also experimented with different numbers of code words. In the scene recognition tasks, we did not observe significant changes in recognition accuracies. In the Caltech 101 data set, however, a higher accuracy $70.74 \pm 0.69\%$ was achieved using 1000 code words (with κ_{HI}, oc_{SVM}, B, and $s = 2$). In comparison, using standard k-means with 1000 code words (together with B, $s = 2$, and $\neg oc_{\text{SVM}}$ which is the better choice for κ_{LIN}), the accuracy is $67.89 \pm 1.11\%$. The proposed method is significantly better than standard k-means codebooks with more visual code words.

In summary, we need to choose the appropriate feature for a specific task (CENTRIST for scene recognition and SIFT for object recognition), and to incorporate as much information as possible.

What's more interesting is the different behaviors of object and scene recognition problems exhibited in our experiments. Scene recognition requires different type of features (CENTRIST instead of SIFT) and less information (performance almost stabilized when step size is 8 and codebook size is 200). We strongly recommend the CENTRIST descriptor, or its variant like `PACT` (Wu and Rehg, 2008), and the proposed algorithms for recognizing place and scene categories.

6. Conclusion

In this article, we show that when the histogram intersection kernel is used as the similarity measure in clustering visual descriptors that are histograms, the generated visual codebooks produce better code words and as a consequence, improve the bag of visual words model. We propose a HIK based codebook generation method which runs almost as fast as k-means and has consistently higher accuracies than k-means codebooks by 2–4% in several benchmark object and scene recognition data sets. As an alternative to k-means, in which cluster centroids are used to represent code words, we proposed a one-class SVM formulation to generate better visual code words. We also generalize the proposed visual codebook generation method to arbitrary additive kernels. In particular, this extends our speedup results to the popular χ^2 kernel. The proposed algorithms achieve state-of-the-art accuracies on four benchmark object and scene recognition data sets.

Although k-median is rarely used to generate codebooks, we empirically evaluated k-median codebooks and recommend it as a compromise between the proposed method and k-means. K-

median codebooks have lower accuracies than HIK codebooks but usually have higher accuracy than k-means codebooks. They also require less memory than HIK codebooks.

We provide a software package, named libHIK, which contains implementation of the methods proposed in this paper. The software is available at `http://c2inet.sce.ntu.edu.sg/Jianxin/projects/libHIK/libHIK.htm`.

Acknowledgments

J. Wu is supported by the NTU startup grant and the Singapore MoE AcRF Tier 1 project RG 34/09. This research was supported in part by a grant from the Google Research Awards Program. We thank the anonymous reviewers, whose comments have helped improving this paper.

References

Ankur Agarwal and Bill Triggs. Multilevel image coding with hyperfeatures. *International Journal of Computer Vision*, 78(1):15–27, 2008.

David Arthur and Sergei Vassilvitskii. **k-means++**: the advantage of careful seeding. In *18th Symposium on Discrete Algorithms*, pages 1027–1035, 2007.

Oren Boiman, Eli Shechtman, and Michal Irani. In defense of nearest-neighbor based image classification. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, 2008.

Sabri Boughorbel, Jean-Philippe Tarel, and Nozha Boujemaa. Generalized histogram intersection kernel for image recognition. In *Proc. Int'l Conf. on Image Processing*, 2005.

Chih-Chung Chang and Chih-Jen Lin. LIBSVM: a library for support vector machines, 2001. Software available at `http://www.csie.ntu.edu.tw/~cjlin/libsvm`.

Navneet Dalal and Bill Triggs. Histograms of oriented gradients for human detection. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, volume 1, pages 886–893, 2005.

Janez Demšar. Statistical comparisons of classifiers over multiple data sets. *Journal of Machine Learning Research*, 7:1–30, 2006.

Mark Everingham, Andrew Zisserman, Christopher Williams, and Luc Van Gool. The PASCAL visual object classes challenge 2006 (VOC 2006) results, 2006.

Li Fei-Fei, Rob Fergus, and Pietro Perona. Learning generative visual models from few training example: an incremental Bayesian approach tested on 101 object categories. In *CVPR 2004, Workshop on Generative-Model Based Vision*, 2004.

Shenghua Gao, Ivor Wai-Hung Tsang, Liang-Tien Chia, and Peilin Zhao. Local features are not lonely – Laplacian sparse coding for image classification. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, 2010.

Chih-Wei Hsu and Chih-Jen Lin. BSVM, 2006. Software available at `http://www.csie.ntu.edu.tw/~cjlin/bsvm`.

Frédéric Jurie and Bill Triggs. Creating efficient codebooks for visual recognition. In *The IEEE Conf. on Computer Vision*, volume 1, pages 604–610, 2005.

Svetlana Lazebnik and Maxim Raginsky. Supervised learning of quantizer codebooks by information loss minimization. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 31(7): 1294–1309, 2009.

Svetlana Lazebnik, Cordelia Schmid, and Jean Ponce. Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, volume II, pages 2169–2178, 2006.

Li-Jia Li and Li Fei-Fei. What, where and who? Classifying events by scene and object recognition. In *The IEEE Conf. on Computer Vision*, 2007.

Jingen Liu and Mubarak Shah. Scene modeling using Co-Clustering. In *The IEEE Conf. on Computer Vision*, 2007.

David G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):91–110, 2004.

Subhransu Maji and Alexander C. Berg. Max-margin additive classifiers for detection. In *The IEEE Conf. on Computer Vision*, 2009.

Subhransu Maji, Alexander C. Berg, and Jitendra Malik. Classification using intersection kernel support vector machines is efficient. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, 2008.

Frank Moosmann, Eric Nowak, and Frederic Jurie. Randomized clustering forests for image classification. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 30(9):1632–1646, 2008.

David Nistér and Henrik Stewénius. Scalable recognition with a vocabulary tree. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, volume 2, pages 2161–2168, 2006.

Francesca Odone, Annalisa Barla, and Alessandro Verri. Building kernels from binary strings for image matching. *IEEE Trans. Image Processing*, 14(2):169–180, 2005.

Florent Perronnin. Universal and adapted vocabularies for generic visual categorization. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 30(7):1243–1256, 2008.

James Philbin, Ondřej Chum, Michael Isard, Josef Sivic, and Andrew Zisserman. Lost in quantization: Improving particular object retrieval in large scale image databases. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, 2008.

Ariadna Quattoni and Antonio Torralba. Recognizing indoor scenes. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, 2009.

Bernhard Schölkopf, Alexander Smola, and Klaus-Robert Müller. Nonlinear component analysis as a kernel eigenvalue problem. *Neural Computation*, 10(5):1299–1319, 1998.

Bernhard Schölkopf, John C. Platt, John Shawe-Taylor, Alex J. Smola, and Robert C. Williamson. Estimating the support of a high-dimensional distribution. *Neural Computation*, 13(7):1443–1471, 2001.

Josef Sivic and Andrew Zisserman. Video Google: A text retrieval approach to object matching in videos. In *The IEEE Conf. on Computer Vision*, volume 2, pages 1470–1477, 2003.

Michael J. Swain and Dana H. Ballard. Color indexing. *International Journal of Computer Vision*, 7(1):11–32, 1991.

Tinne Tuytelaars and Cordelia Schmid. Vector quantizing feature space with a regular lattice. In *The IEEE Conf. on Computer Vision*, 2007.

Jan C. van Gemert, Jan-Mark Geusebroek, Cor J. Veenman, and Arnold W.M. Smeulders. Kernel codebooks for scene categorization. In *European Conf. Computer Vision*, 2008.

Andrea Vedaldi and Andrew Zisserman. Efficient additive kernels via explicit feature maps. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, 2010.

Julia Vogel and Bernt Schiele. Semantic modeling of natural scenes for content-based image retrieval. *International Journal of Computer Vision*, 72(2):133–157, 2007.

Yair Weiss, Antonio Torralba, and Rob Fergus. Spectral hashing. In *Advances in Neural Information Processing Systems 21*, pages 1753–1760, 2009.

John Winn, Antonio Criminisi, and Thomas Minka. Object categorization by learned universal visual dictionary. In *The IEEE Conf. on Computer Vision*, volume 2, pages 1800–1807, 2005.

Jianxin Wu. A fast dual method for HIK SVM learning. In *European Conf. Computer Vision*, LNCS 6312, pages 552–565, 2010.

Jianxin Wu and James M. Rehg. Where am I: Place instance and category recognition using spatial PACT. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, pages 1–8, 2008.

Jianxin Wu and James M. Rehg. Beyond the Euclidean distance: Creating effective visual codebooks using the histogram intersection kernel. In *The IEEE Conf. on Computer Vision*, pages 630–637, 2009.

Jianxin Wu and James M. Rehg. CENTRIST: A visual descriptor for scene categorization. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 33(8):1489–1501, 2011.

Jianchao Yang, Kai Yu, Yihong Gong, and Thomas Huang. Linear spatial pyramid matching using sparse coding for image classification. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, 2009.

Liu Yang, Rong Jin, Rahul Sukthankar, and Frederic Jurie. Unifying discriminative visual codebook generation with classifier training for object category recognition. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, 2008.

Journal of Machine Learning Research 12 (2011) 3119-3145 Submitted 7/10; Revised 04/11; Published

Unsupervised Supervised Learning II: Margin-Based Classification Without Labels

Krishnakumar Balasubramanian KRISHNAKUMAR3@GATECH.EDU
College of Computing
Georgia Institute of Technology
266 Ferst Dr.
Atlanta, GA 30332, USA

Pinar Donmez PINARD@YAHOO-INC.COM
Yahoo! Labs
701 First Ave.
Sunnyale CA 94089, USA

Guy Lebanon LEBANON@CC.GATECH.EDU
College of Computing
Georgia Institute of Technology
266 Ferst Dr.
Atlanta, GA 30332, USA

Editor: Ingo Steinwart

Abstract

Many popular linear classifiers, such as logistic regression, boosting, or SVM, are trained by optimizing a margin-based risk function. Traditionally, these risk functions are computed based on a labeled data set. We develop a novel technique for estimating such risks using only unlabeled data and the marginal label distribution. We prove that the proposed risk estimator is consistent on high-dimensional data sets and demonstrate it on synthetic and real-world data. In particular, we show how the estimate is used for evaluating classifiers in transfer learning, and for training classifiers with no labeled data whatsoever.

Keywords: classification, large margin, maximum likelihood

1. Introduction

Many popular linear classifiers, such as logistic regression, boosting, or SVM, are trained by optimizing a margin-based risk function. For standard linear classifiers $\hat{Y} = \operatorname{sign} \sum \theta_j X_j$ with $Y \in \{-1,+1\}$, and $X, \theta \in \mathbb{R}^d$ the margin is defined as the product

$$Y f_\theta(X) \quad \text{where} \quad f_\theta(X) \stackrel{\text{def}}{=} \sum_{j=1}^{d} \theta_j X_j.$$

Training such classifiers involves choosing a particular value of θ. This is done by minimizing the risk or expected loss

$$R(\theta) = \mathsf{E}_{p(X,Y)} \mathcal{L}(Y, f_\theta(X)) \tag{1}$$

with the three most popular loss functions

$$\mathcal{L}_1(Y, f_\theta(X)) = \exp(-Y f_\theta(X)), \tag{2}$$
$$\mathcal{L}_2(Y, f_\theta(X)) = \log(1+\exp(-Y f_\theta(X))) \text{ and} \tag{3}$$
$$\mathcal{L}_3(Y, f_\theta(X)) = (1-Y f_\theta(X))_+ \tag{4}$$

being exponential loss $\mathcal{L}_1$ (boosting), logloss $\mathcal{L}_2$ (logistic regression) and hinge loss $\mathcal{L}_3$ (SVM) respectively (A_+ above corresponds to A if $A > 0$ and 0 otherwise).

Since the risk $R(\theta)$ depends on the unknown distribution p, it is usually replaced during training with its empirical counterpart

$$R_n(\theta) = \frac{1}{n}\sum_{i=1}^{n} \mathcal{L}(Y^{(i)}, f_\theta(X^{(i)})) \tag{5}$$

based on a labeled training set

$$(X^{(1)}, Y^{(1)}), \ldots, (X^{(n)}, Y^{(n)}) \overset{\text{iid}}{\sim} p \tag{6}$$

leading to the following estimator

$$\hat{\theta}_n = \arg\min_\theta R_n(\theta).$$

Note, however, that evaluating and minimizing R_n requires labeled data (6). While suitable in some cases, there are certainly situations in which labeled data is difficult or impossible to obtain.

In this paper we construct an estimator for $R(\theta)$ using only unlabeled data, that is using

$$X^{(1)}, \ldots, X^{(n)} \overset{\text{iid}}{\sim} p \tag{7}$$

instead of (6). Our estimator is based on the assumption that when the data is high dimensional ($d \to \infty$) the quantities

$$f_\theta(X)|\{Y = y\}, \quad y \in \{-1, +1\} \tag{8}$$

are normally distributed. This phenomenon is supported by empirical evidence and may also be derived using non-iid central limit theorems. We then observe that the limit distributions of (8) may be estimated from unlabeled data (7) and that these distributions may be used to measure margin-based losses such as (2)-(4). We examine two novel unsupervised applications: (i) estimating margin-based losses in transfer learning and (ii) training margin-based classifiers. We investigate these applications theoretically and also provide empirical results on synthetic and real-world data. Our empirical evaluation shows the effectiveness of the proposed framework in risk estimation and classifier training without any labeled data.

The consequences of estimating $R(\theta)$ without labels are indeed profound. Label scarcity is a well known problem which has lead to the emergence of semisupervised learning: learning using a few labeled examples and many unlabeled ones. The techniques we develop lead to a new paradigm that goes beyond semisupervised learning in requiring no labels whatsoever.

2. Unsupervised Risk Estimation

In this section we describe in detail the proposed estimation framework and discuss its theoretical properties. Specifically, we construct an estimator for $R(\theta)$ defined in (1) using the unlabeled data (7) which we denote $\hat{R}_n(\theta; X^{(1)}, \ldots, X^{(n)})$ or simply $\hat{R}_n(\theta)$ (to distinguish it from R_n in (5)).

Our estimation is based on two assumptions. The first assumption is that the label marginals $p(Y)$ are known and that $p(Y=1) \neq p(Y=-1)$. While this assumption may seem restrictive at first, there are many cases where it holds. Examples include medical diagnosis ($p(Y)$ is the well known marginal disease frequency), handwriting recognition or OCR ($p(Y)$ is the easily computable marginal frequencies of different letters in the English language), life expectancy prediction ($p(Y)$ is based on marginal life expectancy tables). In these and other examples $p(Y)$ is known with great accuracy even if labeled data is unavailable. Our experiments show that assuming a wrong marginal $p'(Y)$ causes a graceful performance degradation in $|p(Y) - p'(Y)|$. Furthermore, the assumption of a known $p(Y)$ may be replaced with a weaker form in which we know the ordering of the marginal distributions, for example, $p(Y=1) > p(Y=-1)$, but without knowing the specific values of the marginal distributions.

The second assumption is that the quantity $f_\theta(X)|Y$ follows a normal distribution. As $f_\theta(X)|Y$ is a linear combination of random variables, it is frequently normal when X is high dimensional. From a theoretical perspective this assumption is motivated by the central limit theorem (CLT). The classical CLT states that $f_\theta(X) = \sum_{i=1}^{d} \theta_i X_i | Y$ is approximately normal for large d if the data components $X_1, \ldots, X_d$ are iid given Y. A more general CLT states that $f_\theta(X)|Y$ is asymptotically normal if $X_1, \ldots, X_d | Y$ are independent (but not necessary identically distributed). Even more general CLTs state that $f_\theta(X)|Y$ is asymptotically normal if $X_1, \ldots, X_d | Y$ are not independent but their dependency is limited in some way. We examine this issue in Section 2.1 and also show that the normality assumption holds empirically for several standard data sets.

To derive the estimator we rewrite (1) by taking expectation with respect to Y and $\alpha = f_\theta(X)$

$$R(\theta) = \mathsf{E}_{p(f_\theta(X),Y)} \mathcal{L}(Y, f_\theta(X)) = \sum_{y \in \{-1,+1\}} p(y) \int_{\mathbb{R}} p(f_\theta(X) = \alpha | y) \mathcal{L}(y, \alpha) \, d\alpha. \tag{9}$$

Equation (9) involves three terms $\mathcal{L}(y,\alpha)$, $p(y)$ and $p(f_\theta(X) = \alpha | y)$. The loss function $\mathcal{L}$ is known and poses no difficulty. The second term $p(y)$ is assumed to be known (see discussion above). The third term is assumed to be normal $f_\theta(X) \,|\, \{Y = y\} = \sum_i \theta_i X_i \,|\, \{Y = y\} \sim N(\mu_y, \sigma_y)$ with parameters μ_y, σ_y, $y \in \{-1, 1\}$ that are estimated by maximizing the likelihood of a Gaussian mixture model (we denote $\mu = (\mu_1, \mu_{-1})$ and $\sigma^2 = (\sigma_1^2, \sigma_{-1}^2)$. These estimated parameters are used to construct the plug-in estimator $\hat{R}_n(\theta)$ as follows:

$$\ell_n(\mu, \sigma) = \sum_{i=1}^{n} \log \sum_{y^{(i)} \in \{-1,+1\}} p(y^{(i)}) p_{\mu_y, \sigma_y}(f_\theta(X^{(i)}) | y^{(i)}). \tag{10}$$

$$(\hat{\mu}^{(n)}, \hat{\sigma}^{(n)}) = \arg\max_{\mu, \sigma} \ell_n(\mu, \sigma). \tag{11}$$

$$\hat{R}_n(\theta) = \sum_{y \in \{-1,+1\}} p(y) \int_{\mathbb{R}} p_{\hat{\mu}_y^{(n)}, \hat{\sigma}_y^{(n)}}(f_\theta(X) = \alpha | y) \mathcal{L}(y, \alpha) \, d\alpha. \tag{12}$$

We make the following observations.

1. Although we do not denote it explicitly, μ_y and σ_y are functions of θ.

2. The loglikelihood (10) does not use labeled data (it marginalizes over the label $y^{(i)}$).

3. The parameters of the loglikelihood (10) are $\mu = (\mu_1, \mu_{-1})$ and $\sigma = (\sigma_1, \sigma_{-1})$ rather than the parameter θ associated with the margin-based classifier. We consider the latter one as a fixed constant at this point.

4. The estimation problem (11) is equivalent to the problem of maximum likelihood for means and variances of a Gaussian mixture model where the label marginals are assumed to be known. It is well known that in this case (barring the symmetric case of a uniform $p(y)$) the MLE converges to the true parameter values (Teicher, 1963).

5. The estimator $\hat{R}_n$ (12) is consistent in the limit of infinite unlabeled data

$$P\left(\lim_{n\to\infty} \hat{R}_n(\theta) = R(\theta)\right) = 1.$$

6. The two risk estimators $\hat{R}_n(\theta)$ (12) and $R_n(\theta)$ (5) approximate the expected loss $R(\theta)$. The latter uses labeled samples and is typically more accurate than the former for a fixed n.

7. Under suitable conditions $\arg\min_\theta \hat{R}_n(\theta)$ converges to the expected risk minimizer

$$P\left(\lim_{n\to\infty} \arg\min_{\theta\in\Theta} \hat{R}_n(\theta) = \arg\min_{\theta\in\Theta} R(\theta)\right) = 1.$$

   This far reaching conclusion implies that in cases where $\arg\min_\theta R(\theta)$ is the Bayes classifier (as is the case with exponential loss, log loss, and hinge loss) we can retrieve the optimal classifier without a single labeled data point.

2.1 Asymptotic Normality of $f_\theta(X)|Y$

The quantity $f_\theta(X)|Y$ is essentially a sum of d random variables which under some conditions for large d is likely to be normally distributed. One way to verify this is empirically, as we show in Figures 1-3 which contrast the histogram with a fitted normal pdf for text, digit images, and face images data. For these data sets the dimensionality d is sufficiently high to provide a nearly normal $f_\theta(X)|Y$. For example, in the case of text documents (X_i is the relative number of times word i appeared in the document) d corresponds to the vocabulary size which is typically a large number in the range $10^3 - 10^5$. Similarly, in the case of image classification (X_i denotes the brightness of the i-pixel) the dimensionality is on the order of $10^2 - 10^4$.

Figures 1-3 show that in these cases of text and image data $f_\theta(X)|Y$ is approximately normal for both randomly drawn θ vectors (Figure 1) and for θ representing estimated classifiers (Figures 2 and 3). A caveat in this case is that normality may not hold when θ is sparse, as may happen for example for L_1 regularized models (last row of Figure 2).

From a theoretical standpoint normality may be argued using a central limit theorem. We examine below several progressively more general central limit theorems and discuss whether these theorems are likely to hold in practice for high dimensional data. The original central limit theorem states that $\sum_{i=1}^d Z_i$ is approximately normal for large d if Z_i are iid.

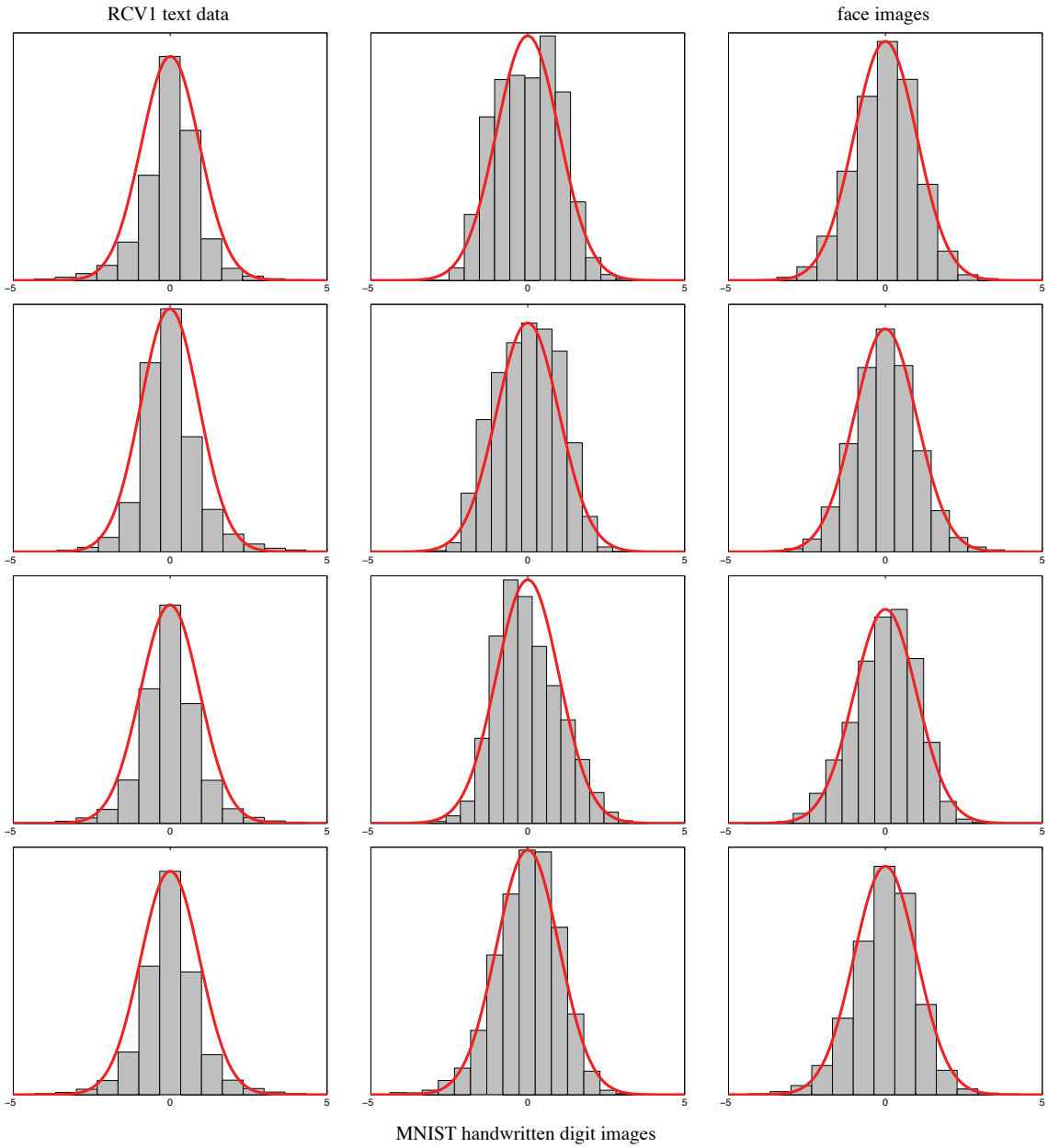


Figure 1: Centered histograms of $f_\theta(X)|\{Y=1\}$ overlayed with the pdf of a fitted Gaussian for randomly drawn θ vectors ($\theta_i \sim U(-1/2, 1/2)$). The columns represent data sets (RCV1 text data, Lewis et al., 2004, MNIST digit images, and face images, Pham et al., 2002) and the rows represent multiple random draws. For uniformity we subtracted the empirical mean and divided by the empirical standard deviation. The twelve panels show that even in moderate dimensionality (RCV1: 1000 top words, MNIST digits: 784 pixels, face images: 400 pixels) the assumption that $f_\theta(X)|Y$ is normal holds often for randomly drawn θ.

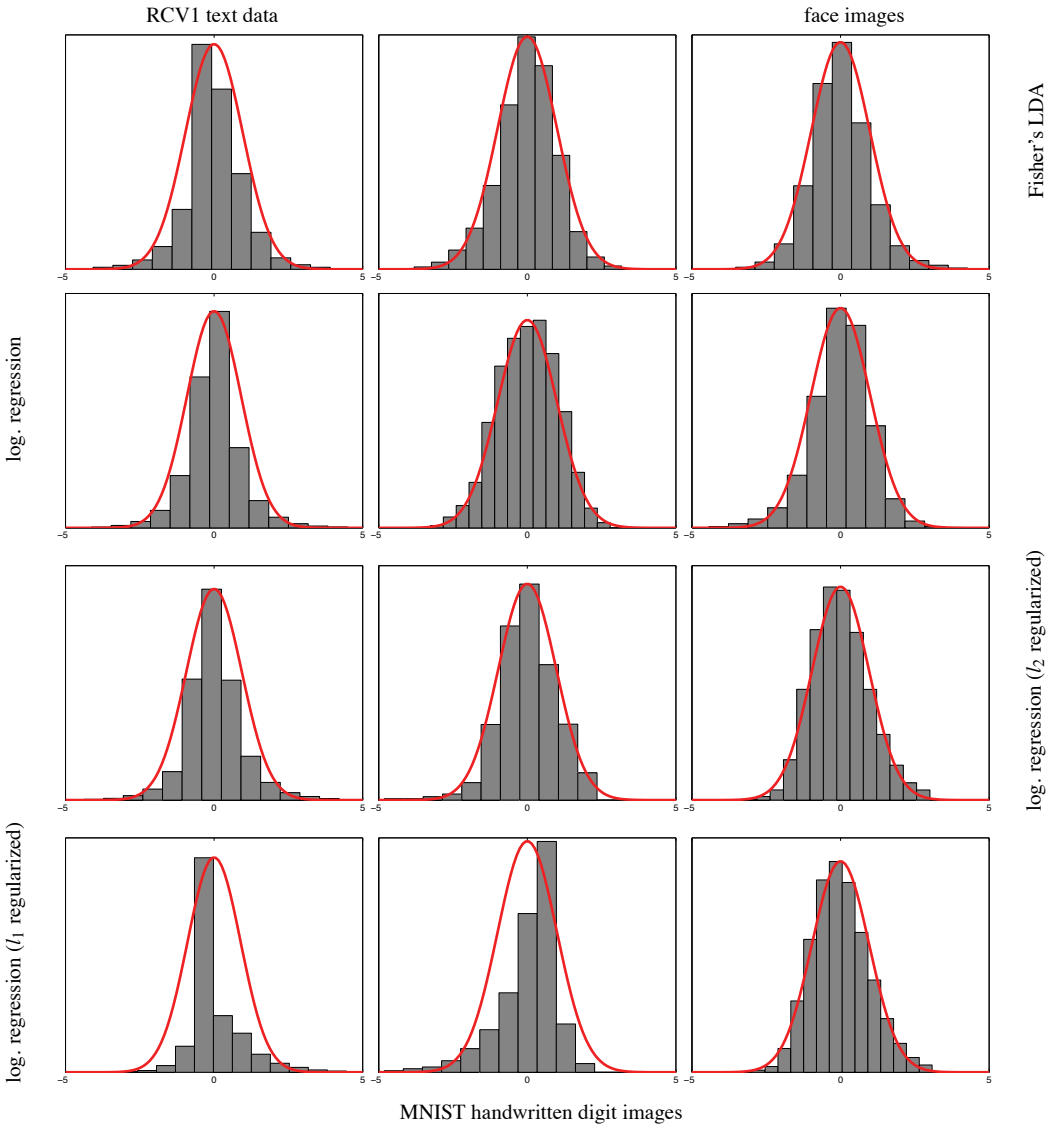


Figure 2: Centered histograms of $f_\theta(X)|\{Y=1\}$ overlayed with the pdf of a fitted Gaussian for multiple θ vectors (four rows: Fisher's LDA, logistic regression, l_2 regularized logistic regression, and l_1 regularized logistic regression-all regularization parameters were selected by cross validation) and data sets (columns: RCV1 text data, Lewis et al., 2004, MNIST digit images, and face images, Pham et al., 2002). For uniformity we subtracted the empirical mean and divided by the empirical standard deviation. The twelve panels show that even in moderate dimensionality (RCV1: 1000 top words, MNIST digits: 784 pixels, face images: 400 pixels) the assumption that $f_\theta(X)|Y$ is normal holds well for fitted θ values (except perhaps for L_1 regularization in the last row which promotes sparse θ).

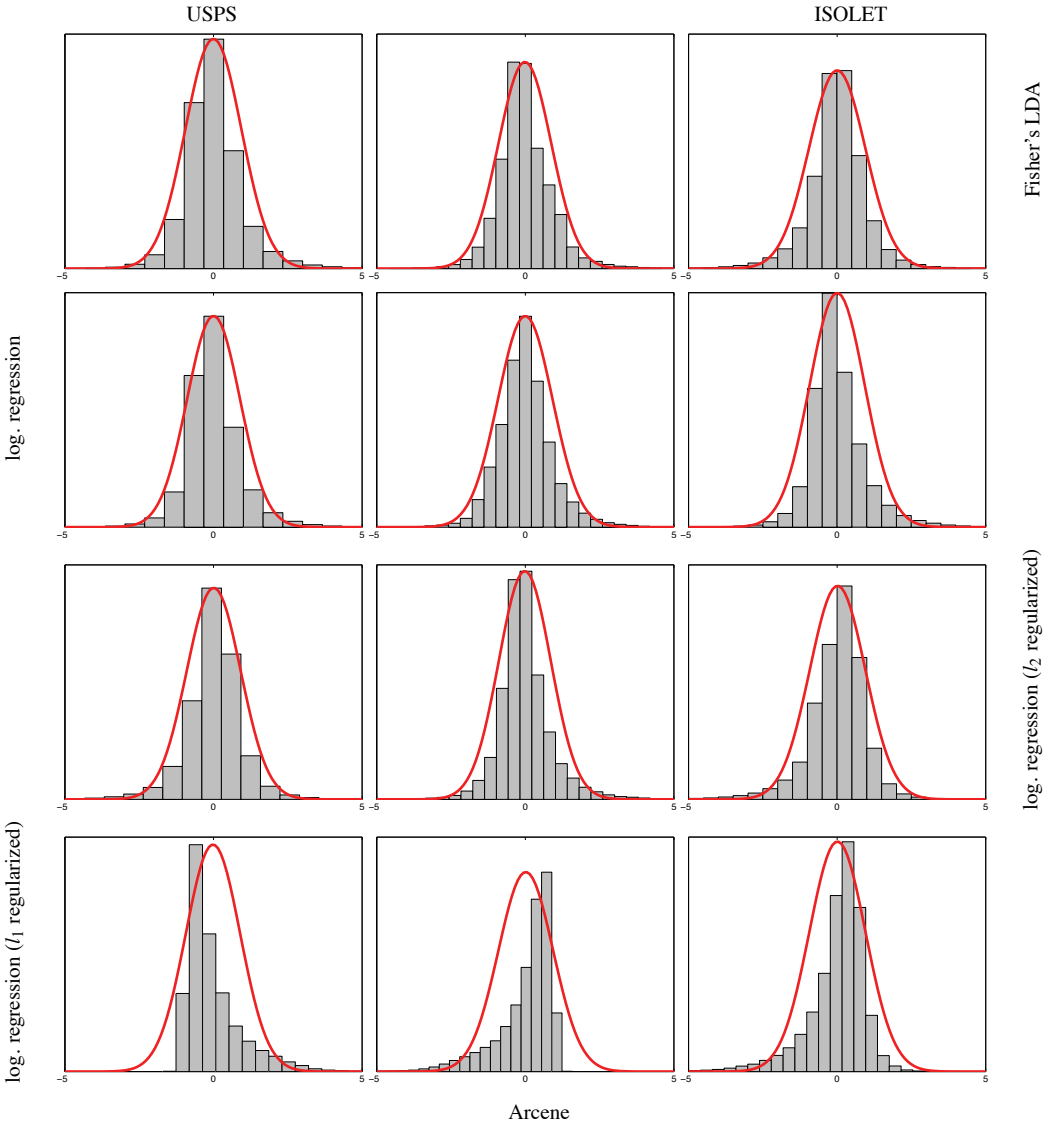


Figure 3: Centered histograms of $f_\theta(X)|\{Y=1\}$ overlayed with the pdf of a fitted Gaussian for multiple θ vectors (four rows: Fisher's LDA, logistic regression, l_2 regularized logistic regression, and l_1 regularized logistic regression-all regularization parameters were selected by cross validation) and data sets (columns: USPS Handwritten Digits, Arcene data set, and ISOLET). For uniformity we subtracted the empirical mean and divided by the empirical standard deviation. The twelve panels further confirm that the assumption that $f_\theta(X)|Y$ is normal holds well for fitted θ values (except perhaps for L_1 regularization in the last row which promotes sparse θ) for various data sets.

Proposition 1 (de-Moivre) *If $Z_i, i \in \mathbb{N}$ are iid with expectation μ and variance σ^2 and $Z_d = d^{-1}\sum_{i=1}^{d} Z_i$ then we have the following convergence in distribution*

$$\sqrt{d}(Z_d - \mu)/\sigma \rightsquigarrow N(0,1) \quad \text{as } d \to \infty.$$

As a result, the quantity $\sum_{i=1}^{d} Z_i$ (which is a linear transformation of $\sqrt{d}(Z_d - \mu)/\sigma$) is approximately normal for large d. This relatively restricted theorem is unlikely to hold in most practical cases as the data dimensions are often not iid.

A more general CLT does not require the summands Z_i to be identically distributed.

Proposition 2 (Lindberg) *For $Z_i, i \in \mathbb{N}$ independent with expectation μ_i and variance σ_i^2, and denoting $s_d^2 = \sum_{i=1}^{d} \sigma_i^2$, we have the following convergence in distribution as $d \to \infty$*

$$s_d^{-1} \sum_{i=1}^{d} (Z_i - \mu_i) \rightsquigarrow N(0,1)$$

if the following condition holds for every $\varepsilon > 0$

$$\lim_{d\to\infty} s_d^{-2} \sum_{i=1}^{d} E(Z_i - \mu_i)^2 1_{\{|X_i - \mu_i| > \varepsilon s_d\}} = 0. \tag{13}$$

This CLT is more general as it only requires that the data dimensions be independent. The condition (13) is relatively mild and specifies that contributions of each of the Z_i to the variance s_d should not dominate it. Nevertheless, the Lindberg CLT is still inapplicable for dependent data dimensions.

More general CLTs replace the condition that $Z_i, i \in \mathbb{N}$ be independent with the notion of $m(k)$-dependence.

Definition 3 *The random variables $Z_i, i \in \mathbb{N}$ are said to be $m(k)$-dependent if whenever $s - r > m(k)$ the two sets $\{Z_1, \ldots, Z_r\}$, $\{Z_s, \ldots, Z_k\}$ are independent.*

An early CLT for $m(k)$-dependent RVs was provided by Hoeffding and Robbins (1948). Below is a slightly weakened version of the CLT, as proved in Berk (1973).

Proposition 4 (Berk) *For each $k \in \mathbb{N}$ let $d(k)$ and $m(k)$ be increasing sequences and suppose that $Z_1^{(k)}, \ldots, Z_{d(k)}^{(k)}$ is an $m(k)$-dependent sequence of random variables. If*

1. $E|Z_i^{(k)}|^2 \leq M$ *for all i and k,*
2. $\mathrm{Var}(Z_{i+1}^{(k)} + \ldots + Z_j^{(k)}) \leq (j-i)K$ *for all i, j, k,*
3. $\lim_{k\to\infty} \mathrm{Var}(Z_1^{(k)} + \ldots + Z_{d(k)}^{(k)})/d(k)$ *exists and is non-zero, and*
4. $\lim_{k\to\infty} m^2(k)/d(k) = 0$

then $\frac{\sum_{i=1}^{d(k)} Z_i^{(k)}}{\sqrt{d(k)}}$ is asymptotically normal as $k \to \infty$.

Proposition 4 states that under mild conditions the sum of $m(k)$-dependent RVs is asymptotically normal. If $m(k)$ is a constant, that is, $m(k) = m$, $m(k)$-dependence implies that a Z_i may only depend on its neighboring dimensions (in the sense of Definition 3). Intuitively, dimensions whose indices are far removed from each other are independent. The full power of Proposition 4 is invoked when $m(k)$ grows with k relaxing the independence restriction as the dimensionality grows. Intuitively, the dependency of the summands is not fixed to a certain order, but it cannot grow too rapidly.

A more realistic variation of $m(k)$ dependence where the dependency of each variable is specified using a dependency graph (rather than each dimension depends on neighboring dimensions) is advocated in a number of papers, including the following recent result by Rinott (1994).

Definition 5 *A graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ indexing random variables is called a dependency graph if for any pair of disjoint subsets of $\mathcal{V}$, A_1 and A_2 such that no edge in $\mathcal{E}$ has one endpoint in A_1 and the other in A_2, we have independence between $\{Z_i : i \in A_1\}$ and $\{Z_i : i \in A_2\}$. The degree $d(v)$ of a vertex is the number of edges connected to it and the maximal degree is* $\max_{v \in \mathcal{V}} d(v)$.

Proposition 6 (Rinott) *Let $Z_1, \ldots, Z_n$ be random variables having a dependency graph whose maximal degree is strictly less than D, satisfying $|Z_i - \mathsf{E}Z_i| \leq B$ a.s., $\forall i$, $\mathsf{E}(\sum_{i=1}^n Z_i) = \lambda$ and $\mathsf{Var}(\sum_{i=1}^n Z_i) = \sigma^2 > 0$, Then for any $w \in \mathbb{R}$,*

$$\left| P\left(\frac{\sum_{i=1}^n Z_i - \lambda}{\sigma} \leq w \right) - \Phi(w) \right| \leq \frac{1}{\sigma} \left(\frac{1}{\sqrt{2\pi}} DB + 16 \left(\frac{n}{\sigma^2} \right)^{1/2} D^{3/2} B^2 + 10 \left(\frac{n}{\sigma^2} \right) D^2 B^3 \right)$$

where $\Phi(w)$ is the CDF corresponding to a N(0,1) distribution.

The above theorem states a stronger result than convergence in distribution to a Gaussian in that it states a uniform rate of convergence of the CDF. Such results are known in the literature as Berry Essen bounds (Davidson, 1994). When D and B are bounded and $\mathsf{Var}(\sum_{i=1}^n Z_i) = O(n)$ it yields a CLT with an optimal convergence rate of $n^{-1/2}$.

The question of whether the above CLTs apply in practice is a delicate one. For text one can argue that the appearance of a word depends on some words but is independent of other words. Similarly for images it is plausible to say that the brightness of a pixel is independent of pixels that are spatially far removed from it. In practice one needs to verify the normality assumption empirically, which is simple to do by comparing the empirical histogram of $f_\theta(X)$ with that of a fitted mixture of Gaussians. As the figures above indicate this holds for text and image data for some values of θ, assuming it is not sparse. Also, it is worth mentioning that one dimensional CLTs kick in relatively early perhaps at 50 or 100 dimensions. Even when the high dimensional data lie on a lower dimensional manifold whose dimensionality is on the order of 100 dimensions, the CLT still applies to some extent (see histogram plots).

2.2 Unsupervised Consistency of $\hat{R}_n(\theta)$

We start with proving identifiability of the maximum likelihood estimator (MLE) for a mixture of two Gaussians with known ordering of mixture proportions. Invoking classical consistency results in conjunction with identifiability we show consistency of the MLE estimator for (μ, σ) parameterizing the distribution of $f_\theta(X)|Y$. As a result consistency of the estimator $\hat{R}_n(\theta)$ follows.

Definition 7 *A parametric family $\{p_\alpha : \alpha \in A\}$ is identifiable when $p_\alpha(x) = p_{\alpha'}(x), \forall x$ implies $\alpha = \alpha'$.*

Proposition 8 *Assuming known label marginals with $p(Y = 1) \neq p(Y = -1)$ the Gaussian mixture family*

$$p_{\mu,\sigma}(x) = p(y = 1) N(x; \mu_1, \sigma_1^2) + p(y = -1) N(x; \mu_{-1}, \sigma_{-1}^2)$$

is identifiable.

Proof It can be shown that the family of Gaussian mixture model with no apriori information about label marginals is identifiable up to a permutation of the labels y (Teicher, 1963). We proceed by

assuming with no loss of generality that $p(y=1) > p(y=-1)$. The alternative case $p(y=1) < p(y=-1)$ may be handled in the same manner. Using the result of Teicher (1963) we have that if $p_{\mu,\sigma}(x) = p_{\mu',\sigma'}(x)$ for all x, then $(p(y),\mu,\sigma) = (p(y),\mu',\sigma')$ up to a permutation of the labels. Since permuting the labels violates our assumption $p(y=1) > p(y=-1)$ we establish $(\mu,\sigma) = (\mu',\sigma')$ proving identifiability. ■

The assumption that $p(y)$ is known is not entirely crucial. It may be relaxed by assuming that it is known whether $p(Y=1) > p(Y=-1)$ or $p(Y=1) < p(Y=-1)$. Proving Proposition 8 under this much weaker assumption follows identical lines.

Proposition 9 *Under the assumptions of Proposition 8 the MLE estimates for* $(\mu,\sigma) = (\mu_1,\mu_{-1},\sigma_1,\sigma_{-1})$

$$(\hat{\mu}^{(n)},\hat{\sigma}^{(n)}) = \arg\max_{\mu,\sigma} \ell_n(\mu,\sigma),$$

$$\ell_n(\mu,\sigma) = \sum_{i=1}^{n} \log \sum_{y^{(i)}\in\{-1,+1\}} p(y^{(i)}) p_{\mu_y,\sigma_y}(f_\theta(X^{(i)})|y^{(i)}).$$

are consistent, that is, $(\hat{\mu}_1^{(n)},\hat{\mu}_{-1}^{(n)},\hat{\sigma}_1^{(n)},\hat{\sigma}_{-1}^{(n)})$ *converge as* $n\to\infty$ *to the true parameter values with probability 1.*

Proof Denoting $p_\eta(z) = \sum_y p(y) p_{\mu_y,\sigma_y}(z|y)$ with $\eta = (\mu,\sigma)$ we note that p_η is identifiable (see Proposition 8) in η and the available samples $z^{(i)} = f_\theta(X^{(i)})$ are iid samples from $p_\eta(z)$. We therefore use standard statistics theory which indicates that the MLE for identifiable parametric model is strongly consistent (Ferguson, 1996, Chapter 17). ■

Proposition 10 *Under the assumptions of Proposition 8 and assuming the loss* $\mathcal{L}$ *is given by one of* (2)-(4) *with a normal* $f_\theta(X)|Y \sim N(\mu_y,\sigma_y^2)$*, the plug-in risk estimate*

$$\hat{R}_n(\theta) = \sum_{y\in\{-1,+1\}} p(y) \int_{\mathbb{R}} p_{\hat{\mu}_y^{(n)},\hat{\sigma}_y^{(n)}}(f_\theta(X)=\alpha|y)\mathcal{L}(y,\alpha)\,d\alpha. \tag{14}$$

is consistent, that is, for all θ*,*

$$P\left(\lim_n \hat{R}_n(\theta) = R(\theta)\right) = 1.$$

Proof The plug-in risk estimate $\hat{R}_n$ in (14) is a continuous function (when L is given by (2), (3) or (4)) of $\hat{\mu}_1^{(n)},\hat{\mu}_{-1}^{(n)},\hat{\sigma}_1^{(n)},\hat{\sigma}_{-1}^{(n)}$ (note that μ_y and σ_y are functions of θ), which we denote $\hat{R}_n(\theta) = h(\hat{\mu}_1^{(n)},\hat{\mu}_{-1}^{(n)},\hat{\sigma}_1^{(n)},\hat{\sigma}_{-1}^{(n)})$.

Using Proposition 9 we have that

$$\lim_{n\to\infty}(\hat{\mu}_1^{(n)},\hat{\mu}_{-1}^{(n)},\hat{\sigma}_1^{(n)},\hat{\sigma}_{-1}^{(n)}) = (\mu_1^{\text{true}},\mu_{-1}^{\text{true}},\sigma_1^{\text{true}},\sigma_{-1}^{\text{true}})$$

with probability 1. Since continuous functions preserve limits we have

$$\lim_{n\to\infty} h(\hat{\mu}_1^{(n)},\hat{\mu}_{-1}^{(n)},\hat{\sigma}_1^{(n)},\hat{\sigma}_{-1}^{(n)}) = h(\mu_1^{\text{true}},\mu_{-1}^{\text{true}},\sigma_1^{\text{true}},\sigma_{-1}^{\text{true}})$$

with probability 1 which implies convergence $\lim_{n\to\infty}\hat{R}_n(\theta) = R(\theta)$ with probability 1. ■

2.3 Unsupervised Consistency of $\arg\min \hat{R}_n(\theta)$

The convergence above $\hat{R}_n(\theta) \to R(\theta)$ is pointwise in θ. If the stronger concept of uniform convergence is assumed over $\theta \in \Theta$ we obtain consistency of $\arg\min_\theta \hat{R}_n(\theta)$. This surprising result indicates that in some cases it is possible to retrieve the expected risk minimizer (and therefore the Bayes classifier in the case of the hinge loss, log-loss and exp-loss) using only unlabeled data. We show this uniform convergence using a modification of Wald's classical MLE consistency result (Ferguson, 1996, Chapter 17).

Denoting

$$p_\eta(z) = \sum_{y\in\{-1,+1\}} p(y) p_{\mu_y,\sigma_y}(f(X) = z|y), \quad \eta = (\mu_1, \mu_{-1}, \sigma_1, \sigma_{-1})$$

we first show that the MLE converges to the true parameter value $\hat{\eta}_n \to \eta_0$ uniformly. Uniform convergence of the risk estimator $\hat{R}_n(\theta)$ follows. Since changing $\theta \in \Theta$ results in a different $\eta \in E$ we can state the uniform convergence in $\theta \in \Theta$ or alternatively in $\eta \in E$.

Proposition 11 *Let θ take values in Θ for which $\eta \in E$ for some compact set E. Then assuming the conditions in Proposition 10 the convergence of the MLE to the true value $\hat{\eta}_n \to \eta_0$ is uniform in $\eta_0 \in E$ (or alternatively $\theta \in \Theta$).*

Proof We start by making the following notation

$$\begin{aligned} U(z,\eta,\eta_0) &= \log p_\eta(z) - \log p_{\eta_0}(z), \\ \alpha(\eta,\eta_0) &= E_{p_{\eta_0}} U(z,\eta,\eta_0) = -D(p_{\eta_0}, p_\eta) \le 0 \end{aligned}$$

with the latter quantity being non-positive and 0 iff $\eta = \eta_0$ (due to Shannon's inequality and identifiability of p_η).

For $\rho > 0$ we define the compact set $S_{\eta_0,\rho} = \{\eta \in E : \|\eta - \eta_0\| \ge \rho\}$. Since $\alpha(\eta,\eta_0)$ is continuous it achieves its maximum (with respect to η) on $S_{\eta_0,\rho}$ denoted by $\delta_\rho(\eta_0) = \max_{\eta\in S_{\eta_0,\rho}} \alpha(\eta,\eta_0) < 0$ which is negative since $\alpha(\eta,\eta_0) = 0$ iff $\eta = \eta_0$. Furthermore, note that $\delta_\rho(\eta_0)$ is itself continuous in $\eta_0 \in E$ and since E is compact it achieves its maximum

$$\delta = \max_{\eta_0\in E} \delta_\rho(\eta_0) = \max_{\eta_0\in E} \max_{\eta\in S_{\eta_0,\rho}} \alpha(\eta,\eta_0) < 0$$

which is negative for the same reason.

Invoking the uniform strong law of large numbers (Ferguson, 1996, Chapter 16) we have $n^{-1}\sum_{i=1}^n U(z^{(i)},\eta,\eta_0) \to \alpha(\eta,\eta_0)$ uniformly over $(\eta,\eta_0) \in E^2$. Consequentially, there exists N such that for $n > N$ (with probability 1)

$$\sup_{\eta_0\in E} \sup_{\eta\in S_{\eta_0,\rho}} \frac{1}{n}\sum_{i=1}^n U(z^{(i)},\eta,\eta_0) < \delta/2 < 0.$$

But since $n^{-1}\sum_{i=1}^n U(z^{(i)},\eta,\eta_0) \to 0$ for $\eta = \eta_0$ it follows that the MLE

$$\hat{\eta}_n = \max_{\eta\in E} \frac{1}{n}\sum_{i=1}^n U(z^{(i)},\eta,\eta_0)$$

is outside $S_{\eta_0,\rho}$ (for $n > N$ uniformly in $\eta_0 \in E$) which implies $\|\hat{\eta}_n - \eta_0\| \le \rho$. Since $\rho > 0$ is arbitrarily and N does not depend on η_0 we have $\hat{\eta}_n \to \eta_0$ uniformly over $\eta_0 \in E$. ■

Proposition 12 *Assuming that X, Θ are bounded in addition to the assumptions of Proposition 11 the convergence $\hat{R}_n(\theta) \to R(\theta)$ is uniform in $\theta \in \Theta$.*

Proof Since X, Θ are bounded the margin value $f_\theta(X)$ is bounded with probability 1. As a result the loss function is bounded in absolute value by a constant C. We also note that a mixture of two Gaussian model (with known mixing proportions) is Lipschitz continuous in its parameters

$$\left| \sum_{y\in\{-1,+1\}} p(y) p_{\hat{\mu}_y^{(n)},\hat{\sigma}_y^{(n)}}(z) - \sum_{y\in\{-1,+1\}} p(y) p_{\mu_y^{true},\sigma_y^{true}}(z) \right| \leq t(z) \cdot \left|\left| (\hat{\mu}_1^{(n)}, \hat{\mu}_{-1}^{(n)}, \hat{\sigma}_1^{(n)}, \hat{\sigma}_{-1}^{(n)}) - (\mu_1^{\text{true}}, \mu_{-1}^{\text{true}}, \sigma_1^{\text{true}}, \sigma_{-1}^{\text{true}}) \right|\right|$$

which may be verified by noting that the partial derivatives of $p_\eta(z) = \sum_y p(y) p_{\mu_y,\sigma_y}(z|y)$

$$\frac{\partial p_\eta(z)}{\partial \hat{\mu}_1^{(n)}} = \frac{p(y=1)(z-\hat{\mu}_1^{(n)})}{(2\pi)^{1/2}\hat{\sigma}_1^{(n)^3}} e^{-\frac{(z-\hat{\mu}_1^{(n)})^2}{2\hat{\sigma}_1^{(n)^3}}},$$

$$\frac{\partial p_\eta(z)}{\partial \hat{\mu}_{-1}^{(n)}} = \frac{p(y=-1)(z-\hat{\mu}_{-1}^{(n)})}{(2\pi)^{1/2}\hat{\sigma}_{-1}^{(n)^3}} e^{-\frac{(z-\hat{\mu}_{-1}^{(n)})^2}{2\hat{\sigma}_{-1}^{(n)^3}}},$$

$$\frac{\partial p_\eta(z)}{\partial \hat{\sigma}_1^{(n)}} = -\frac{p(y=1)(z-\hat{\mu}_1^{(n)})^2}{(2\pi)^{3/2}\hat{\sigma}_1^{(n)^6}} e^{-\frac{(z-\hat{\mu}_1^{(n)})^2}{2\hat{\sigma}_1^{(n)^2}}},$$

$$\frac{\partial p_\eta(z)}{\partial \hat{\sigma}_{-1}^{(n)}} = -\frac{p(y=-1)(z-\hat{\mu}_{-1}^{(n)})^2}{(2\pi)^{3/2}\hat{\sigma}_{-1}^{(n)^6}} e^{-\frac{(z-\hat{\mu}_{-1}^{(n)})^2}{2\hat{\sigma}_{-1}^{(n)^2}}}$$

are bounded for a compact E. These observations, together with Proposition 11 lead to

$$\begin{aligned}
|\hat{R}_n(\theta) - R(\theta)| &\leq \sum_{y\in\{-1,+1\}} p(y) \int \left| p_{\hat{\mu}_y^{(n)},\hat{\sigma}_y^{(n)}}(f_\theta(X)=\alpha) - p_{\mu_y^{\text{true}},\sigma_y^{\text{true}}}(f_\theta(X)=\alpha) \right| |\mathcal{L}(y,\alpha)| d\alpha \\
&\leq C \int \left| \sum_{y\in\{-1,+1\}} p(y) p_{\hat{\mu}_y^{(n)},\hat{\sigma}_y^{(n)}}(\alpha) - \sum_{y\in\{-1,+1\}} p(y) p_{\mu_y^{\text{true}},\sigma_y^{\text{true}}}(\alpha) \right| d\alpha \\
&\leq C \left\| (\hat{\mu}_1^{(n)}, \hat{\mu}_{-1}^{(n)}, \hat{\sigma}_1^{(n)}, \hat{\sigma}_{-1}^{(n)}) - (\mu_1^{\text{true}}, \mu_{-1}^{\text{true}}, \sigma_1^{\text{true}}, \sigma_{-1}^{\text{true}}) \right\| \int_a^b t(z) dz \\
&\leq C' \left\| (\hat{\mu}_1^{(n)}, \hat{\mu}_{-1}^{(n)}, \hat{\sigma}_1^{(n)}, \hat{\sigma}_{-1}^{(n)}) - (\mu_1^{\text{true}}, \mu_{-1}^{\text{true}}, \sigma_1^{\text{true}}, \sigma_{-1}^{\text{true}}) \right\| \to 0
\end{aligned}$$

uniformly over $\theta \in \Theta$. ■

Proposition 13 *Under the assumptions of Proposition 12*

$$P\left(\lim_{n\to\infty} \arg\min_{\theta\in\Theta} \hat{R}_n(\theta) = \arg\min_{\theta\in\Theta} R(\theta) \right) = 1.$$

Proof We denote $t^* = \arg\min R(\theta)$, $t_n = \arg\min \hat{R}_n(\theta)$. Since $\hat{R}_n(\theta) \to R(\theta)$ uniformly, for each $\varepsilon > 0$ there exists N such that for all $n > N$, $|\hat{R}_n(\theta) - R(\theta)| < \varepsilon$.

Let $S = \{\theta : \|\theta - t^*\| \geq \varepsilon\}$ and $\min_{\theta \in S} R(\theta) > R(t^*)$ (S is compact and thus R achieves its minimum on it). There exists N' such that for all $n > N'$ and $\theta \in S$, $\hat{R}_n(\theta) \geq R(t^*) + \varepsilon$. On the other hand, $\hat{R}_n(t^*) \to R(t^*)$ which together with the previous statement implies that there exists N'' such that for $n > N''$, $\hat{R}_n(t^*) < \hat{R}_n(\theta)$ for all $\theta \in S$. We thus conclude that for $n > N''$, $t_n \notin S$. Since we showed that for each $\varepsilon > 0$ there exists N such that for all $n > N$ we have $\|t_n - t^*\| \leq \varepsilon$, $t_n \to t^*$ which concludes the proof. ■

2.4 Asymptotic Variance

In addition to consistency, it is useful to characterize the accuracy of our estimator $\hat{R}_n(\theta)$ as a function of $p(y), \mu, \sigma$. We do so by computing the asymptotic variance of the estimator which equals the inverse Fisher information

$$\sqrt{n}(\hat{\eta}_n^{\text{mle}} - \eta_0) \rightsquigarrow N(0, I^{-1}(\eta^{\text{true}}))$$

and analyzing its dependency on the model parameters. We first derive the asymptotic variance of MLE for mixture of Gaussians (we denote below $\eta = (\eta_1, \eta_2), \eta_i = (\mu_i, \sigma_i)$)

$$\begin{aligned} p_\eta(z) &= p(Y=1)N(z;\mu_1,\sigma_1^2) + p(Y=-1)N(z;\mu_{-1},\sigma_{-1}^2) \\ &= p_1 p_{\eta_1}(z) + p_{-1} p_{\eta_{-1}}(z). \end{aligned}$$

The elements of 4×4 information matrix $I(\eta)$

$$I(\eta_i, \eta_j) = \mathsf{E}\left(\frac{\partial \log p_\eta(z)}{\partial \eta_i} \frac{\partial \log p_\eta(z)}{\partial \eta_j}\right)$$

may be computed using the following derivatives

$$\begin{aligned} \frac{\partial \log p_\eta(z)}{\partial \mu_i} &= \frac{p_i}{\sigma_i}\left(\frac{z-\mu_i}{\sigma_i}\right)\frac{p_{\eta_i}(z)}{p_\eta(z)}, \\ \frac{\partial \log p_\eta(z)}{\partial \sigma_i^2} &= \frac{p_i}{2\sigma_i}\left(\left(\frac{z-\mu_i}{\sigma_i}\right)^2 - 1\right)\frac{p_{\eta_i}(z)}{p_\eta(z)} \end{aligned}$$

for $i = 1, -1$. Using the method of Behboodian (1972) we obtain

$$\begin{aligned} I(\mu_i,\mu_j) &= \frac{p_i p_j}{\sigma_i \sigma_j} M_{11}\Big(p_{\eta_i}(z), p_{\eta_i}(z)\Big), \\ I(\mu_1,\sigma_i^2) &= \frac{p_1 p_i}{2\sigma_1\sigma_i^2}\Big[M_{12}\Big(p_{\eta_i}(z), p_{\eta_i}(z)\Big) - M_{10}\Big(p_{\eta_1}(z), p_{\eta_i}(z)\Big)\Big], \\ I(\mu_{-1},\sigma_i^2) &= \frac{p_{-1} p_i}{2\sigma_{-1}\sigma_i^2}\Big[M_{21}\Big(p_{\eta_i}(z), p_{\eta_{-1}}(z)\Big) - M_{01}\Big(p_{\eta_i}(z), p_{\eta_{-1}}(z)\Big)\Big], \\ I(\sigma_i^2,\sigma_i^2) &= \frac{p_i^4}{4\sigma_i^4}\Big[M_{00}\Big(p_{\eta_i}(z), p_{\eta_i}(z)\Big) - 2M_{11}\Big(p_{\eta_i}(z), p_{\eta_i}(z)\Big) + M_{22}\Big(p_{\eta_i}(z), p_{\eta_i}(z)\Big)\Big], \\ I(\sigma_1^2,\sigma_{-1}^2) &= \frac{p_1 p_{-1}}{4\sigma_1^2\sigma_{-1}^2}\Big[M_{00}\Big(p_{\eta_1}(z), p_{\eta_{-1}}(z)\Big) - M_{20}\Big(p_{\eta_1}(z), p_{\eta_{-1}}(z)\Big) \\ &\qquad - M_{02}\Big(p_{\eta_1}(z), p_{\eta_{-1}}(z)\Big) + M_{22}\Big(p_{\eta_1}(z), p_{\eta_{-1}}(z)\Big)\Big] \end{aligned}$$

where

$$M_{m,n}\Big(p_{\eta_i}(z), p_{\eta_j}(z)\Big) = \int_{-\infty}^{\infty} \left(\frac{z-\mu_i}{\sigma_i}\right)^m \left(\frac{z-\mu_j}{\sigma_j}\right)^n \frac{p_{\eta_i}(z)p_{\eta_j}(z)}{p_\eta(z)}\, dx.$$

In some cases it is more instructive to consider the asymptotic variance of the risk estimator $\hat{R}_n(\theta)$ rather than that of the parameter estimate for $\eta = (\mu, \sigma)$. This could be computed using the delta method and the above Fisher information matrix

$$\sqrt{n}(\hat{R}_n(\theta) - R(\theta)) \rightsquigarrow N(0, \nabla h(\eta^{\text{true}})^T I^{-1}(\eta^{true}) \nabla h(\eta^{\text{true}}))$$

where ∇h is the gradient vector of the mapping $R(\theta) = h(\eta)$. For example, in the case of the exponential loss (2) we get

$$\begin{aligned}
h(\eta) &= p(Y=1)\sigma_1\sqrt{2}\exp\Big(\frac{(\mu_1-1)^2}{2} - \frac{\mu_1^2}{2\sigma_1^2}\Big) + p(Y=-1)\sigma_{-1}\sqrt{2}\exp\Big(\frac{(\mu_{-1}-1)^2}{2} - \frac{\mu_{-1}^2}{2\sigma_{-1}^2}\Big),\\
\frac{\partial h(\eta)}{\partial \mu_1} &= \frac{\sqrt{2}P(Y=1)(\mu_1(\sigma_1^2-1)-\sigma_1^2)}{\sigma_1}\exp\left(\frac{(\mu_1-1)^2}{2} - \frac{\mu_1^2}{2\sigma_1^2}\right),\\
\frac{\partial h(\eta)}{\partial \mu_{-1}} &= \frac{\sqrt{2}P(Y=-1)(\mu_{-1}(\sigma_{-1}^2-1)+\sigma_{-1}^2)}{\sigma_{-1}}\exp\left(\frac{(\mu_{-1}+1)^2}{2} - \frac{\mu_{-1}^2}{2\sigma_{-1}^2}\right),\\
\frac{\partial h(\eta)}{\partial \sigma_1^2} &= \frac{P(Y=1)(\mu_1^2+\sigma_1^2)}{\sqrt{2\sigma_1}}\Big(\frac{(\mu_1-1)^2}{2} - \frac{\mu_1^2}{2\sigma_1^2}\Big),\\
\frac{\partial h(\eta)}{\partial \sigma_{-1}^2} &= \frac{P(Y=-1)(\mu_{-1}^2+\sigma_{-1}^2)}{\sqrt{2\sigma_{-1}}}\Big(\frac{(\mu_{-1}+1)^2}{2} - \frac{\mu_{-1}^2}{2\sigma_{-1}^2}\Big).
\end{aligned}$$

Figure 4 plots the asymptotic accuracy of $\hat{R}_n(\theta)$ for log-loss. The left panel shows that the accuracy of $\hat{R}_n$ increases with the imbalance of the marginal distribution $p(Y)$. The right panel shows that the accuracy of $\hat{R}_n$ increases with the difference between the means $|\mu_1 - \mu_{-1}|$ and the variances σ_1/σ_2.

2.5 Multiclass Classification

Thus far, we have considered unsupervised risk estimation in binary classification. In this section we describe a multiclass extension based on standard extensions of the margin concept to multiclass classification. In this case the margin vector associated with the multiclass classifier

$$\hat{Y} = \underset{k=1,\ldots,K}{\arg\max}\, f_{\theta^k}(X), \qquad X, \theta^k \in \mathbb{R}^d$$

is $f_{\boldsymbol{\theta}}(X) = (f_{\theta^1}(X), \ldots, f_{\theta^K}(X))$. Following our discussion of the binary case, $f_{\theta^k}(X)|Y, k = 1, \ldots, K$ is assumed to be normally distributed with parameters that are estimated by maximizing the likelihood of a Gaussian mixture model. We thus have K Gaussian mixture models, each one with K mixture components. The estimated parameters are plugged-in as before into the multiclass risk

$$R(\boldsymbol{\theta}) = E_{p(f_{\boldsymbol{\theta}}(X),Y)} \mathcal{L}(Y, f_{\boldsymbol{\theta}}(X))$$

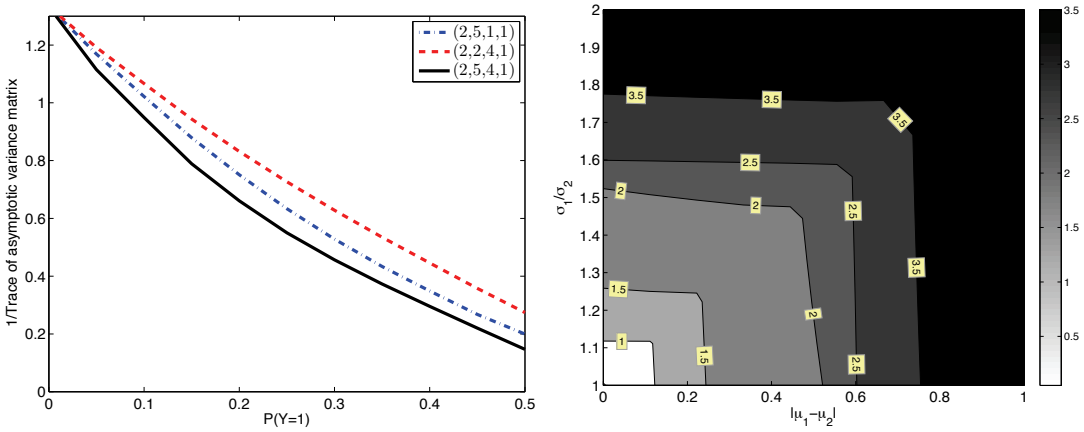


Figure 4: Left panel: asymptotic accuracy (inverse of trace of asymptotic variance) of $\hat{R}_n(\theta)$ for logloss as a function of the imbalance of the class marginal $p(Y)$. The accuracy increases with the class imbalance as it is easier to separate the two mixture components. Right panel: asymptotic accuracy (inverse of trace of asymptotic variance) as a function of the difference between the means $|\mu_1 - \mu_{-1}|$ and the variances σ_1/σ_2. See text for more information.

where $\mathcal{L}$ is a multiclass margin based loss function such as

$$\mathcal{L}(Y, f_{\boldsymbol{\theta}}(X)) = \sum_{k \neq Y} \log(1 + \exp(-f_{\theta^k}(X))), \tag{15}$$

$$\mathcal{L}(Y, f_{\boldsymbol{\theta}}(X)) = \sum_{k \neq Y} (1 + f_{\theta^k}(X))_+. \tag{16}$$

Care should be taken when defining the loss function for the multi-class case, as a straight-forward extension from the binary case might render the framework inconsistent. We use the specific extension which is proved to be consistent for various loss functions (including hinge-loss) by Tewari and Bartlett (2007). Since the MLE for a Gaussian mixture model with K components is consistent (assuming $P(Y)$ is known and all probabilities $P(Y = k), k = 1, \ldots, K$ are distinct) the MLE estimator for $f_{\theta^k}(X)|Y = k'$ are consistent. Furthermore, if the loss $\mathcal{L}$ is a continuous function of these parameters (as is the case for (15)-(16)) the risk estimator $\hat{R}_n(\boldsymbol{\theta})$ is consistent as well.

3. Application 1: Estimating Risk in Transfer Learning

We consider applying our estimation framework in two ways. The first application, which we describe in this section, is estimating margin-based risks in transfer learning where classifiers are trained on one domain but tested on a somewhat different domain. The transfer learning assumption that labeled data exists for the training domain but not for the test domain motivates the use of our unsupervised risk estimation. The second application, which we describe in the next section, is more ambitious. It is concerned with training classifiers without labeled data whatsoever.

In evaluating our framework we consider both synthetic and real-world data. In the synthetic experiments we generate high dimensional data from two uniform distributions $X|\{Y = 1\}$ and

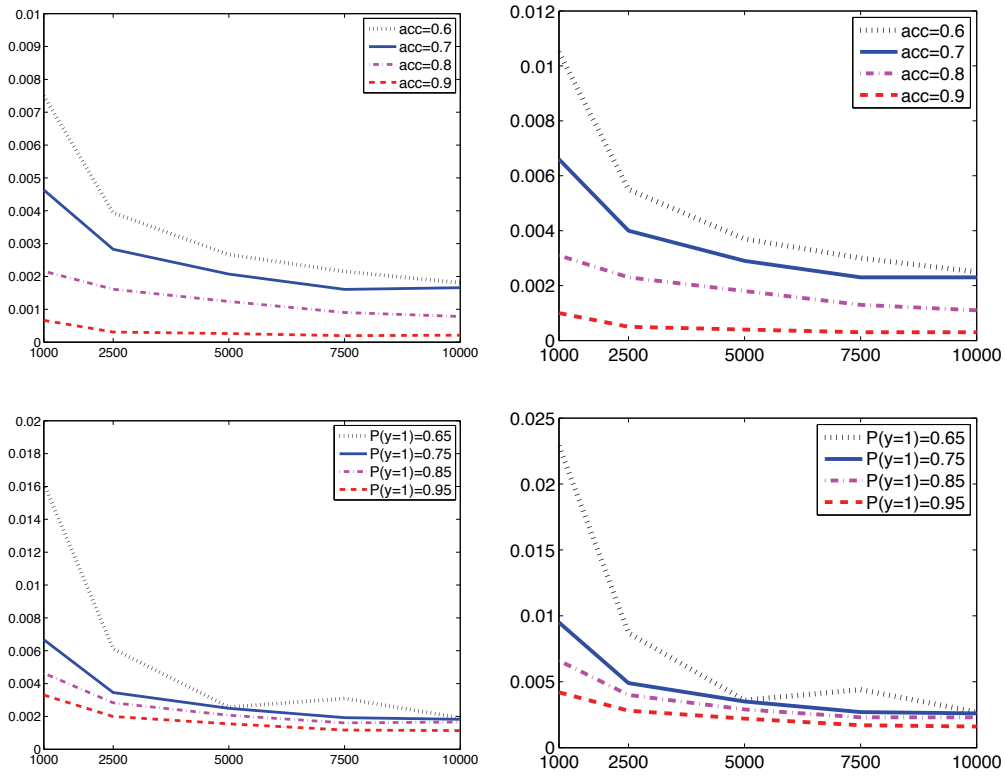


Figure 5: The relative accuracy of $\hat{R}_n$ (measured by $|\hat{R}_n(\theta) - R_n(\theta)|/R_n(\theta)$) as a function of n, classifier accuracy (acc) and the label marginal $p(Y)$ (left: logloss, right: hinge-loss). The estimation error nicely decreases with n (approaching 1% at $n = 1000$ and decaying further). It also decreases with the accuracy of the classifier (top) and non-uniformity of $p(Y)$ (bottom) in accordance with the theory of Section 2.4.

$X|\{Y = -1\}$ with independent dimensions and prescribed $p(Y)$ and classification accuracy. This controlled setting allows us to examine the accuracy of the risk estimator as a function of n, $p(Y)$, and the classifier accuracy.

Figure 5 shows that the relative error of $\hat{R}_n(\theta)$ (measured by $|\hat{R}_n(\theta) - R_n(\theta)|/R_n(\theta)$) in estimating the logloss (left) and hinge loss (right). The curves decrease with n and achieve accuracy of greater than 99% for $n > 1000$. In accordance with the theoretical results in Section 2.4 the figure shows that the estimation error decreases as the classifiers become more accurate and as $p(Y)$ becomes less uniform. We found these trends to hold in other experiments as well. In the case of exponential loss, however, the estimator performed substantially worse across the board, in some cases with an absolute error of as high as 10. This is likely due to the exponential dependency of the loss on $Yf_\theta(X)$ which makes it very sensitive to outliers.

Table 1 shows the accuracy of logloss estimation for a real world transfer learning experiment based on the 20-newsgroup data. We followed the experimental setup of used by Dai et al. (2007) in order to have different distributions for training and test sets. More specifically, 20-newsgroup

Data	R_n	$\lvert R_n - \hat{R}_n \rvert$	$\lvert R_n - \hat{R}_n \rvert / R_n$	n	$p(Y=1)$
sci vs. comp	0.7088	0.0093	0.013	3590	0.8257
sci vs. rec	0.641	0.0141	0.022	3958	0.7484
talk vs. rec	0.5933	0.0159	0.026	3476	0.7126
talk vs. comp	0.4678	0.0119	0.025	3459	0.7161
talk vs. sci	0.5442	0.0241	0.044	3464	0.7151
comp vs. rec	0.4851	0.0049	0.010	4927	0.7972

Table 1: Error in estimating logloss for logistic regression classifiers trained on one 20-newsgroup classification task and tested on another. We followed the transfer learning setup described by Dai et al. (2007) which may be referred to for more detail. The train and testing sets contained samples from two top categories in the topic hierarchy but with different subcategory proportions. The first column indicates the top category classification task and the second indicates the empirical log-loss R_n calculated using the true labels of the testing set (5). The third and forth columns indicate the absolute and relative errors of $\hat{R}_n$. The fifth and sixth columns indicate the train set size and the label marginal distribution.

data has a hierarchical class taxonomy and the transfer learning problem is defined at the top-level categories. We split the data based on subcategories such that the training and test sets contain data sampled from different subcategories within the same top-level category. Hence, the training and test distributions differ. We trained a logistic regression classifier on the training set and estimate its risk on the test set of a different distribution. Our unsupervised risk estimator was quite effective in estimating the risk with relative accuracy greater than 96% and absolute error less than 0.02.

4. Application 2: Unsupervised Learning of Classifiers

Our second application is a very ambitious one: training classifiers using unlabeled data by minimizing the unsupervised risk estimate $\hat{\theta}_n = \arg\min \hat{R}_n(\theta)$. We evaluate the performance of the learned classifier $\hat{\theta}_n$ based on three quantities: (i) the unsupervised risk estimate $\hat{R}_n(\hat{\theta}_n)$, (ii) the supervised risk estimate $R_n(\hat{\theta}_n)$, and (iii) its classification error rate. We also compare the performance of $\hat{\theta}_n = \arg\min \hat{R}_n(\theta)$ with that of its supervised analog $\arg\min R_n(\theta)$.

We compute $\hat{\theta}_n = \arg\min \hat{R}_n(\theta)$ using two algorithms (see Algorithms 1-2) that start with an initial $\theta^{(0)}$ and iteratively construct a sequence of classifiers $\theta^{(1)}, \ldots, \theta^{(T)}$ which steadily decrease $\hat{R}_n$. Algorithm 1 adopts a gradient descent-based optimization. At each iteration t, it approximates the gradient vector $\nabla \hat{R}_n(\theta^{(t)})$ numerically using a finite difference approximation (17). We compute the integral in the loss function estimator using numeric integration. Since the integral is one dimensional a variety of numeric methods may be used with high accuracy and fast computation. Algorithm 2 proceeds by constructing a grid search along every dimension of $\theta^{(t)}$ and set $[\theta^{(t)}]_i$ to the grid value that minimizes $\hat{R}_n$ (iteratively optimize one dimension at a time). This amounts to greedy search converging to local maxima. The same might hold for Algorithm 1, but we observe that Algorithm 1 works slightly better in practice, leading to lower test error with less number of training iterations.

Although we focus on unsupervised training of logistic regression (minimizing unsupervised logloss estimate), the same techniques may be generalized to train other margin-based classifiers such as SVM by minimizing the unsupervised hinge-loss estimate.

Algorithm 1 Unsupervised Gradient Descent

Input: $X^{(1)},\ldots,X^{(n)} \in \mathbb{R}^d$, $p(Y)$, step size α
Initialize $t=0$, $\theta^{(t)}=\theta^0 \in \mathbb{R}^d$
repeat
  Compute $f_{\theta^{(t)}}(X^{(j)}) = \langle \theta^{(t)}, X^{(j)} \rangle \; \forall j = 1,\ldots,n$
  Estimate $(\hat{\mu}_1, \hat{\mu}_{-1}, \hat{\sigma}_1, \hat{\sigma}_{-1})$ by maximizing (11)
  for $i=1$ **to** d **do**
    Plug-in the estimates into (14) to approximate

$$\frac{\partial \hat{R}_n(\theta^{(t)})}{\partial \theta_i} = \frac{\hat{R}_n(\theta^{(t)} + h_i e_i) - \hat{R}_n(\theta^{(t)} - h_i e_i)}{2h_i}$$
$$(e_i \text{ is an all zero vector except for } [e_i]_i = 1) \tag{17}$$

  end for
  Form $\nabla \hat{R}_n(\theta^{(t)}) = (\frac{\partial \hat{R}_n(\theta^{(t)})}{\partial \theta_1^{(t)}}, \ldots, \frac{\partial \hat{R}_n(\theta^{(t)})}{\partial \theta_d^{(t)}})$
  Update $\theta^{(t+1)} = \theta^{(t)} - \alpha \nabla \hat{R}_n(\theta^{(t)})$, $t = t+1$
until convergence
Output: linear classifier $\theta^{\text{final}} = \theta^{(t)}$

Algorithm 2 Unsupervised Grid Search

Input: $X^{(1)},\ldots,X^{(n)} \in \mathbb{R}^d$, $p(Y)$, grid-size τ
Initialize $\theta_i \sim \text{Uniform}(-2,2)$ for all i
repeat
  for $i=1$ **to** d **do**
    Construct τ points grid in the range $[\theta_i - 4\tau, \theta_i + 4\tau]$
    Compute the risk estimate (14) where all dimensions of $\theta^{(t)}$ are fixed except for $[\theta^{(t)}]_i$ which is evaluated at each grid point.
    Set $[\theta^{(t+1)}]_i$ to the grid value that minimized (14)
  end for
until convergence
Output: linear classifier $\theta^{\text{final}} = \theta$

Figures 6-7 display $\hat{R}_n(\hat{\theta}_n)$, $R_n(\hat{\theta}_n)$ and error-rate$(\hat{\theta}_n)$ on the training and testing sets as on two real world data sets: RCV1 (text documents) and MNIST (handwritten digit images) data sets. In the case of RCV1 we discarded all but the most frequent 504 words (after stop-word removal) and represented documents using their tfidf scores. We experimented on the binary classification task of distinguishing the top category (positive) from the next 4 top categories (negative) which resulted in $p(y=1) = 0.3$ and $n = 199328$. 70% of the data was chosen as a (unlabeled) training set and the rest was held-out as a test-set. In the case of MNIST data, we normalized each of the $28 \times 28 = 784$

pixels to have 0 mean and unit variance. Our classification task was to distinguish images of the digit one (positive) from the digit 2 (negative) resulting in 14867 samples and $p(Y=1)=0.53$. We randomly choose 70% of the data as a training set and kept the rest as a testing set.

Figures 6-7 indicate that minimizing the unsupervised logloss estimate is quite effective in learning an accurate classifier without labels. Both the unsupervised and supervised risk estimates $\hat{R}_n(\hat{\theta}_n), R_n(\hat{\theta}_n)$ decay nicely when computed over the train set as well as the test set. Also interesting is the decay of the error rate. For comparison purposes supervised logistic regression with the same n achieved only slightly better test set error rate: 0.05 on RCV1 (instead of 0.1) and 0.07 or MNIST (instead of 0.1).

In another experiment we examined the proposed approach on several different data sets and compared the classification performance with a supervised baseline (logistic regression) and Gaussian mixture modeling (GMM) clustering with known label proportions in the original data space (Table 2). The comparison was made under the same experimental setting (n, $p(Y)$) for all three approaches. We used data sets from UCI machine learning repository (Frank and Asuncion, 2010) and from previously cited sources, unless otherwise noted. The following tasks were considered for each data set.

- RCV1: top category versus next 4 categories
- MNIST: Digit 1 versus Digit 2
- 20 newsgroups: Comp category versus Recreation category
- USPS[1]: Digit 2 versus Digit 5
- Umist[1]: Male face (16 subjects) versus Female faces (4 subjects) with image resolution reduced to 40×40
- Arcene: Cancer versus Normal
- Isolet: Vowels versus Consonants
- Dexter: Documents about corporate acquisitions versus rest
- Secom: Semiconductor manufacturing defects versus good items
- Pham faces: Face versus Non-face images
- CMU pie face[2]: male (30 subjects) vs female (17 subjects)
- Madelon[3]: It consists of data points (artificially generated) grouped in 32 clusters placed on the vertices of a five dimensional hypercube and randomly labeled +1 or -1, corrupted with features that are not useful for classification.

1. Data set can be found at `http://www.cs.nyu.edu/~roweis/data.html`.
2. Data set can be found at `http://www.zjucadcg.cn/dengcai/Data/FaceData.html`.
3. Data set can be found at `http://archive.ics.uci.edu/ml/machine-learning-databases/madelon/Dataset.pdf`.

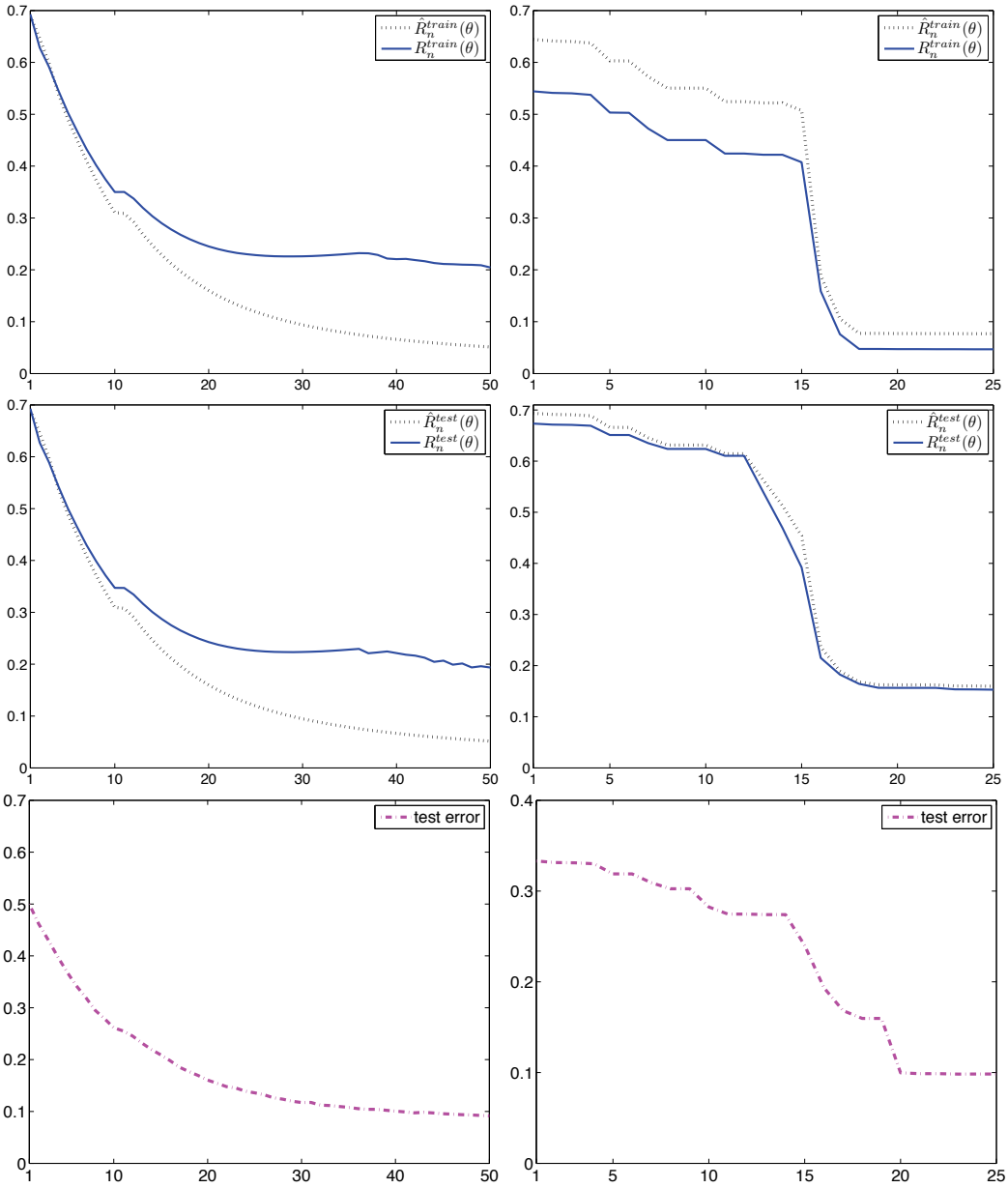


Figure 6: Performance of unsupervised logistic regression classifier $\hat{\theta}_n$ computed using Algorithm 1 (left) and Algorithm 2 (right) on the RCV1 data set. The top two rows show the decay of the two risk estimates $\hat{R}_n(\hat{\theta}_n)$, $R_n(\hat{\theta}_n)$ as a function of the algorithm iterations. The risk estimates of $\hat{\theta}_n$ were computed using the train set (top) and the test set (middle). The bottom row displays the decay of the test set error rate of $\hat{\theta}_n$ as a function of the algorithm iterations. The figure shows that the algorithm obtains a relatively accurate classifier (testing set error rate 0.1, and $\hat{R}_n$ decaying similarly to R_n) without the use of a single labeled example. For comparison, the test error rate for supervised logistic regression with the same n is 0.07.

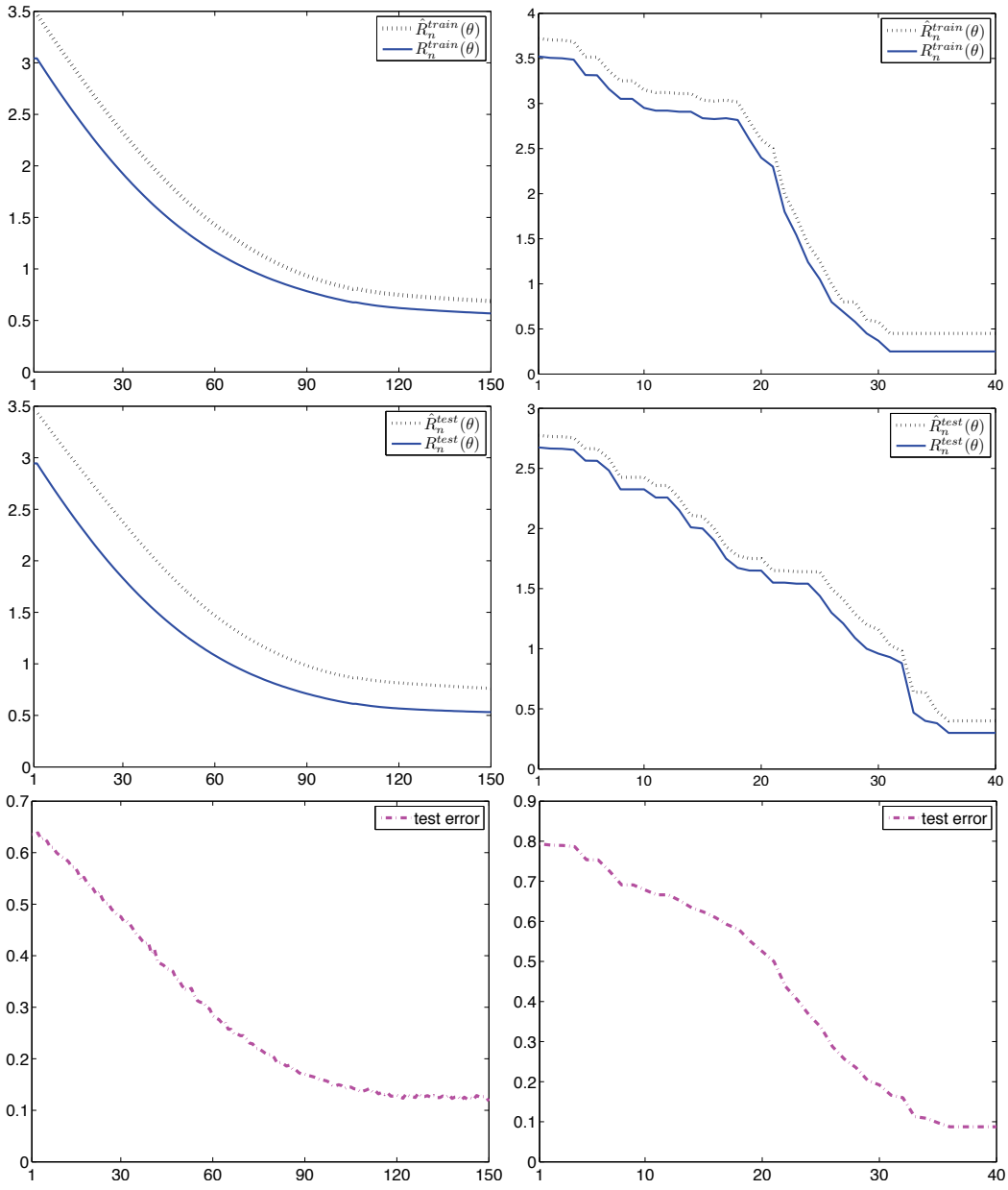


Figure 7: Performance of unsupervised logistic regression classifier $\hat{\theta}_n$ computed using Algorithm 1 (left) and Algorithm 2 (right) on the MNIST data set. The top two rows show the decay of the two risk estimates $\hat{R}_n(\hat{\theta}_n)$, $R_n(\hat{\theta}_n)$ as a function of the algorithm iterations. The risk estimates of $\hat{\theta}_n$ were computed using the train set (top) and the test set (middle). The bottom row displays the decay of the test set error rate of $\hat{\theta}_n$ as a function of the algorithm iterations. The figure shows that the algorithm obtains a relatively accurate classifier (testing set error rate 0.1, and $\hat{R}_n$ decaying similarly to R_n) without the use of a single labeled example. For comparison, the test error rate for supervised logistic regression with the same n is 0.05.

Data set	Dimensions	Supervised log-reg	USL-2	GMM
RCV1	top 504 words	0.0500	0.0923	0.2083
Mnist	784	0.0700	0.1023	0.3163
20 news group	top 750 words	0.0652	0.0864	0.1234
USPS	256	0.0348	0.0545	0.1038
Umist	400 PCA components	0.1223	0.1955	0.2569
Arcene	1000 PCA components	0.1593	0.1877	0.3843*
Isolet	617	0.0462	0.0568	0.1332
Dexter	top-700 words	0.0564	0.1865	0.2715
Secom	591	0.1246	0.1532	0.2674
Pham faces	400	0.1157	0.1669	0.2324
CMU pie face	1024	0.0983	0.1386	0.2682*
Madelon	500	0.0803	0.1023	0.1120

Table 2: Comparison (test set error rate) between supervised logistic regression, Unsupervised logistic regression and Gaussian mixture modeling in original data space. The unsupervised classifier performs better than the GMM clustering on the original space and compares well with its supervised counterpart on most data sets. See text for more details. The stars represent GMM with covariance $\sigma^2 I$ due to the high dimensionality. In all other cases we used a diagonal covariance matrix. Non-diagonal covariance matrix was impractical due to the high dimensionality.

Table 2 displays the test set error for the three methods on each data set. We note that our unsupervised approach achieves test set errors comparable to the supervised logistic regression in several data sets. The poor performance of the unsupervised technique on the Dexter data set is due to the fact that the data contains many irrelevant features. In fact it was engineered for a feature selection competition and has a sparse solution vector. In general our method significantly outperforms Gaussian mixture model clustering in the original feature space. A likely explanation is that (i) $f_\theta(X)|Y$ is more likely to be normal than $X|Y$ and (ii) it is easier to estimate in one dimensional space rather than in a high dimensional space.

4.1 Inaccurate Specification of $p(Y)$

Our estimation framework assumes that the marginal $p(Y)$ is known. In some cases we may only have an inaccurate estimate of $p(Y)$. It is instructive to consider how the performance of the learned classifier degrades with the inaccuracy of the assumed $p(Y)$.

Figure 8 displays the performance of the learned classifier for RCV1 data as a function of the assumed value of $p(Y=1)$ (correct value is $p(Y=1)=0.3$). We conclude that knowledge of $p(Y)$ is an important component in our framework but precise knowledge is not crucial. Small deviations of the assumed $p(Y)$ from the true $p(Y)$ result in a small degradation of logloss estimation quality and testing set error rate. Naturally, large deviation of the assumed $p(Y)$ from the true $p(Y)$ renders the framework ineffective.

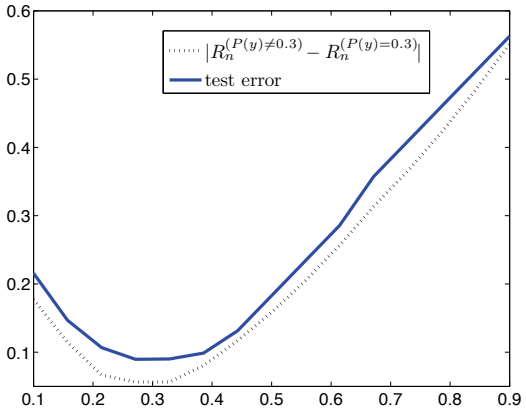


Figure 8: Performance of unsupervised classifier training on RCV1 data (top class vs. classes 2-5) for misspecified $p(Y)$. The performance of the estimated classifier (in terms of training set empirical logloss R_n (5) and test error rate measured using held-out labels) decreases with the deviation between the assumed and true $p(Y=1)$ (true $p(Y=1)=0.3$)). The classifier performance is very good when the assumed $p(Y)$ is close to the truth and degrades gracefully when the assumed $p(Y)$ is not too far from the truth.

4.2 Effect of Regularization and Dimensionality Reduction

In Figure 9 we examine the effect of regularization on the performance of the unsupervised classifier. In this experiment we use the L_1 regularization software available at `http://www.cs.ubc.ca/~schmidtm/Software/L1General.html`. Clearly, regularization helps in the supervised case. It appears that in the USL case weak regularization may improve performance but not as drastically as in the supervised case. Furthermore, the positive effect of L_1 regularization in the USL case appears to be weaker than L_2 regularization (compare the left and right panels of Figure 9). One possible reason is that the sparsity promoting nature of L_1 conflicts with the CLT assumption.

In Figure 10 we examine the effect of reducing the data dimensionality via PCA prior to training the unsupervised classifier. Specifically, the 256 dimensions USPS image data set was embedded in an increasingly lower dimensional space via PCA. For the original dimensionality of 256 or a slightly lower dimensionality the classification performance of the unsupervised classifier is comparable to the supervised. Once the dimensions are reduced to less than 150 a significant performance gap appears. This is consistent with our observation above that for lower dimensions the CLT approximation is less accurate. The supervised classifier also degrades in performance as less dimensions are used but not as fast as the unsupervised classifier.

5. Related Work

Semi-supervised approaches: Semisupervised learning is closely related to our work in that unsupervised classification may be viewed as a limiting case. One of the first attempts at studying the sample complexity of classification with unlabeled and labeled data was by Castelli and Cover (1995). They consider a setting when data is generated by mixture distributions and show that with infinite unlabeled data, the probability of error decays exponentially faster in the labeled data to the

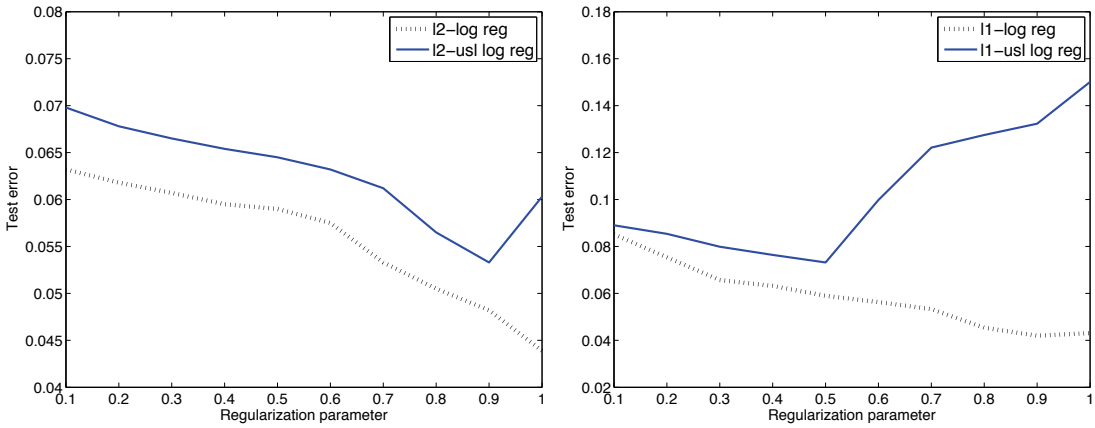


Figure 9: Test set error rate versus regularization parameter (L_2 on the left panel and L_1 on the right panel) for supervised and unsupervised logistic regression on RCV1 data set.

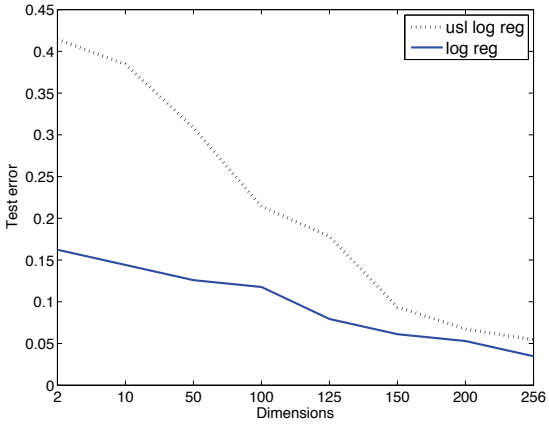


Figure 10: Test set error rate versus the amount of dimensions used (extracted via PCA) for supervised and unsupervised logistic regression on USPS data set. The original dimensionality was 256.

Bayes risk. They also analyze the case when there are only finite labeled and unlabeled data samples, with known class conditional densities but unknown mixing proportions (Castelli and Cover, 1996). A variant of the same scenario with known parametric forms for the class conditionals (specifically n-dimensional Gaussians) but unknown parameters and mixing proportions is also analyzed by J. Ratsaby and Venkatesh (1995). Some of the more recent work in the area concentrated on analyzing semisupervised learning under the cluster assumption or the manifold assumption. We refer the reader to a recent survey by Zhu and Goldberg (2009) for a discussion of recent approaches. However, none of the prior work consider mixture modeling in the projected 1-d space along with a CLT assumption which we exploit. In addition, assuming known mixing proportions, we propose

a framework for training a classifier with no labeled samples, while approaches above still need labeled samples for classification.

Unsupervised approaches: The most recent related research approaches are by Quadrianto et al. (2009), Gomes et al. (2010), and Donmez et al. (2010). The work by Quadrianto et al. (2009) aims to estimate the labels of an unlabeled testing set using known label proportions of several sets of unlabeled observations. The key difference between their approach and ours is that they require separate training sets from different sampling distributions with different and known label marginals (one for each label). Our method assumes only a single data set with a known label marginal but on the other hand assumed the CLT approximation. Furthermore, as noted previously (see comment after Proposition 8), our analysis is in fact valid when only the order of label proportions is known, rather than the absolute values.

A different attempt at solving this problem is provided by Gomes et al. (2010) which focuses on discriminative clustering. This approach attempts to estimate a conditional probabilistic model in an unsupervised way by maximizing mutual information between the empirical input distribution and the label distribution. A key difference is the focus on probabilistic classifiers and in particular logistic regression whereas our approach is based on empirical risk minimization which also includes SVM. Another key difference is that the work by Gomes et al. (2010) lacks consistency results which characterize when it works from a theoretical perspective. The approach by Donmez et al. (2010) focuses on estimating the error rate of a given stochastic classifier (not necessarily linear) without labels. It is similar in that it estimates the 0/1 risk rather than the margin based risk. However, it uses a different strategy and it replaces the CLT assumption with a symmetric noise assumption.

An important distinction between our work and the references above is that our work provides an estimate for the margin-based risk and therefore leads naturally to unsupervised versions of logistic regression and support vector machines. We also provide asymptotic analysis showing convergence of the resulting classifier to the optimal classifier (minimizer of (1)). Experimental results show that in practice the accuracy of the unsupervised classifier is on the same order (but slightly lower naturally) as its supervised analog.

6. Discussion

In this paper we developed a novel framework for estimating margin-based risks using only unlabeled data. We show that it performs well in practice on several different data sets. We derived a theoretical basis by casting it as a maximum likelihood problem for Gaussian mixture model followed by plug-in estimation.

Remarkably, the theory states that assuming normality of $f_\theta(X)$ and a known $p(Y)$ we are able to estimate the risk $R(\theta)$ without a single labeled example. That is the risk estimate converges to the true risk as the number of unlabeled data increase. Moreover, using uniform convergence arguments it is possible to show that the proposed training algorithm converges to the optimal classifier as $n \to \infty$ without any labeled data. The results in the paper are applicable only to linear classifiers, which are an extremely important class of classifiers especially in the high dimensional case. In the non-linear classification scenario, it is worth examining if the CLT assumptions on the mapped high-dimensional feature space could be used for building non-linear classifiers via the kernel trick.

On a more philosophical level, our approach points at novel questions that go beyond supervised and semi-supervised learning. What benefit do labels provide over unsupervised training? Can

our framework be extended to semi-supervised learning where a few labels do exist? Can it be extended to non-classification scenarios such as margin based regression or margin based structured prediction? When are the assumptions likely to hold and how can we make our framework even more resistant to deviations from them? These questions and others form new and exciting open research directions.

Acknowledgments

The authors thank the action editor and anonymous reviewers for their constructive comments, that not only helped at a conceptual level but also helped improve the presentation. In addition, we thank John Lafferty and Vladimir Koltchinskii for discussions and several insightful comments. This work was funded in part by NSF grant IIS-0906550.

References

J. Behboodian. Information matrix for a mixture of two normal distributions. *Journal of Statistical Computation and Simulation*, 1(4):295–314, 1972.

K. N. Berk. A central limit theorem for m-dependent random variables with unbounded m. *The Annals of Probability*, 1(2):352–354, 1973.

V. Castelli and T. M. Cover. On the exponential value of labeled samples. *Pattern Recognition Letters*, 16(1):105–111, 1995.

V. Castelli and T. M. Cover. The relative value of labeled and unlabeled samples in pattern recognition with an unknown mixing parameter. *IEEE Transactions on Information Theory*, 42(6): 2102–2117, 1996.

W. Dai, Q. Yang, G.-R. Xue, and Y. Yu. Boosting for transfer learning. In *Proc. of International Conference on Machine Learning*, 2007.

J. Davidson. *Stochastic Limit Theory: An Introduction for Econometricians*. Oxford University Press, USA, 1994.

P. Donmez, G. Lebanon, and K. Balasubramanian. Unsupervised supervised learning I: Estimating classification and regression error rates without labels. *Journal of Machine Learning Research*, 11(April):1323–1351, 2010.

T. S. Ferguson. *A Course in Large Sample Theory*. Chapman & Hall, 1996.

A. Frank and A. Asuncion. UCI machine learning repository. *University of California, School of Information and Computer Science, Irvine, CA. Available at http://archive.ics.uci.edu/ml/*, 2010.

R. Gomes, A. Krause, and P. Perona. Discriminative clustering by regularized information maximization. In *Advances in Neural Information Processing Systems 24*, 2010.

W. Hoeffding and H. Robbins. The central limit theorem for dependent random variables. *Duke Mathematical Journal*, 15:773–780, 1948.

J. J. Ratsaby and S. S. Venkatesh. Learning from a mixture of labeled and unlabeled examples with parametric side information. In *Annual conference on Computational learning theory*, 1995.

D. Lewis, Y. Yang, T. Rose, and F. Li. RCV1: A new benchmark collection for text categorization research. *Journal of Machine Learning Research*, 5:361–397, 2004.

T. V. Pham, M. Worring, and A. W. M. Smeulders. Face detection by aggregated bayesian network classifiers. *Pattern Recognition Letters*, 23(4):451–461, February 2002.

N. Quadrianto, A. J. Smola, T. S. Caetano, and Q. V. Le. Estimating labels from label proportions. *Journal of Machine Learning Research*, 10:2349–2374, 2009.

Y. Rinott. On normal approximation rates for certain sums of dependent random variables. *Journal of Computational and Applied Mathematics*, 55(2):135–143, 1994.

H. Teicher. Identifiability of finite mixtures. *The Annals of Mathematical Statistics*, 34(4):1265–1269, 1963.

A. Tewari and P. L. Bartlett. On the consistency of multiclass classification methods. *Journal of Machine Learning Research*, pages 1007–1025, 2007.

X. Zhu and A. B. Goldberg. *Introduction to Semi-supervised Learning*. Morgan & Claypool Publishers, 2009.

Journal of Machine Learning Research 12 (2011) 3147-3186 Submitted 9/10; Revised 5/11; Published 11/11

Adaptive Exact Inference in Graphical Models

Özgür Sümer OSUMER@CS.UCHICAGO.EDU
Department of Computer Science
University of Chicago
1100 E. 58th Street
Chicago, IL 60637, USA

Umut A. Acar UMUT@MPI-SWS.ORG
Max-Planck Institute for Software Systems
MPI-SWS Campus E 1 4
D-66123 Saarbruecken, Germany

Alexander T. Ihler IHLER@ICS.UCI.EDU
Donald Bren School of Information and Computer Science
University of California, Irvine
Irvine, CA 92697 USA

Ramgopal R. Mettu METTU@ECS.UMASS.EDU
Electrical and Computer Engineering Department
University of Massachusetts, Amherst
151 Holdsworth Way
Amherst, MA 01003, USA

Editor: Neil Lawrence

Abstract

Many algorithms and applications involve repeatedly solving variations of the same inference problem, for example to introduce new evidence to the model or to change conditional dependencies. As the model is updated, the goal of *adaptive inference* is to take advantage of previously computed quantities to perform inference more rapidly than from scratch. In this paper, we present algorithms for adaptive exact inference on general graphs that can be used to efficiently compute marginals and update MAP configurations under arbitrary changes to the input factor graph and its associated elimination tree. After a linear time preprocessing step, our approach enables updates to the model and the computation of any marginal in time that is logarithmic in the size of the input model. Moreover, in contrast to max-product our approach can also be used to update MAP configurations in time that is roughly proportional to the number of updated entries, rather than the size of the input model. To evaluate the practical effectiveness of our algorithms, we implement and test them using synthetic data as well as for two real-world computational biology applications. Our experiments show that adaptive inference can achieve substantial speedups over performing complete inference as the model undergoes small changes over time.

Keywords: exact inference, factor graphs, factor elimination, marginalization, dynamic programming, MAP computation, model updates, parallel tree contraction

1. Introduction

Graphical models provide a rich framework for describing structure within a probability distribution, and have proven to be useful in numerous application areas such as computational biology, statistical physics, and computer vision. Considerable efforts have been made to understand and minimize the computational complexity of inferring the marginal probabilities or most likely state of a graphical model. However, in many applications we may need to perform repeated computations over a collection of very similar models. For example, hidden Markov models are commonly used for sequence analysis of DNA, RNA and proteins, while protein structure requires the definition of a factor graph defined by the three-dimensional topology of the protein of interest. For both of these types of models, it is often desirable to study the effects of mutation on functional or structural properties of the gene or protein. In this setting, each putative mutation gives rise to a new problem that is nearly identical to the previously solved problem.

The changes described in the examples above can, of course, be handled by incorporating them into the model and then performing inference from scratch. However, in general we may wish to assess thousands of potential changes to the model—for example, the number of possible mutations in a protein structure grows exponentially with the number of considered sites—and minimize the total amount of work required. *Adaptive inference* refers to the problem of handling changes to the model (e.g., to model parameters and even dependency structure) more efficiently than performing inference from scratch. Performing inference in an adaptive manner requires a new algorithmic approach, since it requires us to balance the computational cost of the inference procedure with the reusability of its calculations. As a simple example, suppose that we wish to compute the marginal distribution of a leaf node in a Markov chain with n variables. Using the standard sum-product algorithm, upon a change to the conditional probability distribution at one end of the chain, we must perform $\Omega(n)$ computation to compute the marginal distribution of the node at the other end of the chain. In such a setting, it is worth using additional preprocessing time to restructure the underlying model in such a way that changes to the model can be handled in time that is logarithmic, rather than linear, in the size of the model.

In this paper, we focus on developing efficient algorithms for performing exact inference in the adaptive setting. Specifically, we present techniques for two basic inference tasks in general graphical models: marginalization and finding maximum *a posteriori* (MAP) configurations. Our high-level approach to enabling efficient updates of the model, and recalculation of marginals or a MAP configuration, is to "cluster" parts of the input model by computing partial eliminations, and construct a balanced-tree data structure with depth $O(\log n)$. We use a process based on factor elimination (Darwiche, 2009) that we call *hierarchical clustering* that takes as input a graph and elimination tree (equivalent to a tree-decomposition of the graphical model), and produces an alternative, balanced elimination sequence. The sufficient statistics of the balanced elimination are re-usable in the sense that they will remain largely unchanged by any small update to the model. In particular, changes to factors and the variables they depend on can be performed in time that is logarithmic in the size of the input model. Furthermore, we show that after such updates, the time necessary to compute marginal distributions is logarithmic in the size of the model, and the time to update a MAP configuration is roughly proportional to the number of variables whose values have changed.

1.1 Related Work

There are numerous machine learning and artificial intelligence problems, such as path planning problems in robotics, where new information or observations require changing a previously computed solution. As an example, problems solved by heuristic search techniques have benefited greatly from incremental algorithms (Koenig et al., 2004), in which solutions can be efficiently updated by reusing previously searched parts of the solution space. The problem of performing adaptive inference in graphical models was first considered by Delcher et al. (1995). In their work, they introduced a logarithmic time method for updating marginals under changes to observed variables in the model. Their algorithm relies on the input model being tree-structured, and can only handle changes to observations in the input model. At a high level their approach is similar to our own, in that they also use a linear time preprocessing step to transform the input tree-structured model into a balanced tree representation. However, their algorithm addresses only updates to "observations" in the model, and cannot update dependencies in the input model. Additionally, while their algorithm can be applied to general graphs by performing a tree decomposition, it is not clear whether the tree decomposition itself can be easily updated, as is necessary to remain efficient when modifying the input model. Adaptive exact inference using graph-cut techniques has also been studied by Kohli and Torr (2007). Although the running time of their method does not depend on the tree-width of the input model, it is restricted to pairwise models with binary variables or with submodular pairwise factors. Adaptivity for approximate inference has also been studied by Komodakis et al. (2008); in this work, adaptivity is achieved by performing "warm starts". That is, a change to model is simply made at the final iteration of approximate inference and the algorithm is restarted from this state and allowed to continue until convergence.

The preprocessing technique used by Delcher et al. (1995) is inspired by a method known as *parallel tree contraction*, devised by Miller and Reif (1985) to evaluate expressions on parallel architectures. In parallel tree contraction we must evaluate a given expression tree, where internal nodes are arithmetic operations and leaves are input values. The parallel algorithm of Miller and Reif (1985) works by "contracting" both leaves and internal nodes of the tree in rounds. At each round, the nodes to eliminate are chosen in a random fashion and it can be shown that, in expectation, a constant fraction of the nodes are eliminated in each round. By performing contractions in parallel, the expression tree can be evaluated in logarithmic time and linear total work. Parallel tree contraction can be applied to any semi-ring, including sum-product (marginalization) and max-product (maximization) operators, making it directly applicable to inference problems, and it has also been used to develop efficient parallel implementations of inference (Pennock, 1998; Namasivayam et al., 2006; Xia and Prasanna, 2008).

An interesting property of tree contraction is that it can also be made to be *adaptive* to changes in the input (Acar et al., 2004, 2005). In particular, the techniques of self-adjusting computation (Acar, 2005; Acar et al., 2006, 2009a; Hammer et al., 2009) show that tree contraction can, for example, be used to derive an efficient and reasonably general data structure for dynamic trees (Sleator and Tarjan, 1983). In this paper we apply similar techniques to develop a new algorithm for adaptive inference that can handle arbitrary changes to the input model and can be used for both marginalization and for computing MAP configurations.

1.2 Contributions

In this paper, we present a new framework for adaptive exact inference, building upon the work of Delcher et al. (1995). Given a factor graph G with n nodes, and domain size d (each variable can take d different values), we require the user to specify an elimination tree T on factors. Our framework for adaptive inference requires a preprocessing step in which we build a balanced representation of the input elimination tree in $O(d^{3w}n)$ time where w is the width of the input elimination tree T. We show that this balanced representation, which we call a *cluster tree*, is essentially equivalent to a tree decomposition. For marginal computations, a change to the model can be processed in $O(d^{3w} \cdot \log n)$ time, and the marginal for particular variable can be computed in $O(d^{2w} \cdot \log n)$ time. For a change to the model that induces ℓ changes to a MAP configuration, our approach can update the MAP configuration in $O(d^{3w} \log n + d^w \ell \log(n/\ell))$ time, without knowing ℓ or the changed entries in the configuration.

As in standard approaches for exact inference in general graphs, our algorithm has an exponential dependence on the tree-width of the input model. The dependence in our case, however is stronger: if the input elimination tree has width w, our balanced representation is guaranteed to have width at most $3w$. As a result the running time of our algorithms for building the cluster tree as well as the updates have a $O(d^{3w})$ multiplicative factor; updates to the model and queries however require logarithmic, rather than linear, time in the size of the graph. Our approach is therefore most suitable for settings in which a single build operation is followed by a large number of updates and queries.

Since d and w can often be bounded by reasonably small constant factors, we know that there exists some n beyond which we would achieve speedups, but where exactly the speedups materialize is important in practice. To evaluate the practical effectiveness of our approach, we implement the proposed algorithms and present an experimental evaluation by considering both synthetic data (Section 6.1) and real data (Sections 6.2 and 6.3). Our experiments using synthetically generated factor graphs show that even for modestly-sized graphs ($10 - 1000$ nodes) our algorithm provides orders of magnitude speedup over computation from scratch for computing both marginals and MAP configurations. Thus, the overhead observed in practice is negligible compared to the speedup possible using our framework. Given that the asymptotic difference between linear and logarithmic run-times can be large, it is not surprising that our approach yields speedups for large models. The reason for the observed speedups in the smaller graphs is due to the fact that constant factors hidden by the asymptotic bounds associated with the exponential bounds are small (because they involve fast floating point operations) and because our worst-case bounds are often not attained for relatively small graphs (Section 6.1.5).

In addition, we also show the applicability of our framework to two problems in computational structural biology (Sections 6.2 and 6.3). First, we apply our algorithm to protein secondary structure prediction using an HMM, showing that secondary structure types can be efficiently updated as mutations are made to the primary sequence. For this application, our algorithm is one to two orders of magnitude faster than computation from scratch. We also apply our algorithm to protein sidechain packing, in which a (general) factor graph defines energetic interactions in a three-dimensional protein structure and we must find a minimum-energy conformation of the protein. For this problem, our algorithm can be used to maintain a minimum-energy conformation as changes are being made to the underlying protein. In our experiments, we show that for a subset of the SCWRL benchmark

(Canutescu et al., 2003), our algorithm is nearly 7 times faster than computing minimum-energy conformations from scratch.

Several elements of this work have appeared previously in conference versions (Acar et al., 2007, 2008, 2009b). In this paper we unify these into a single framework and improve our algorithms and our bounds in several ways. Specifically, we present deterministic versions of the algorithms, including a key update algorithm and its proof of correctness; we derive upper bounds in terms of the tree-width, the size of the model, and the domain size; and we give a detailed experimental analysis.

1.3 Outline

The remainder of the paper is organized as follows. In Section 2, we give the definitions and notation used throughout this paper, along with some background on the factor elimination algorithm and tree decompositions. In Section 3, we describe our algorithm and the cluster tree data structure and how they can be used for marginalization. Then, in Section 4, we describe how updates to the underlying model can be performed efficiently. In Section 5, we extend our algorithm to compute and maintain MAP configurations under model changes. In Section 6, we show experimental results for our approach on three synthetic benchmarks and two applications in computational biology. We conclude with a discussion of future directions in Section 7.

2. Background

Factor graphs (Kschischang et al., 2001) describe the factorization structure of the function $g(X)$ using a bipartite graph consisting of *variable* nodes and *factor* nodes. Specifically, suppose such a graph $G = (X, F)$ consists of variable nodes $X = \{x_1, \ldots, x_n\}$ and factor nodes $F = \{f_1, \ldots, f_m\}$ (see Figure 1a). We denote the adjacency relationship in graph G by $\sim_G$, and let $X_{f_j} = \{x_i \in X : x_i \sim_G f_j\}$ be the set of variables adjacent to factor f_j. For example, in Figure 1a, $X_{f_5} = \{x, v\}$. G is said to be consistent with a function $g(\cdot)$ if and only if

$$g(x_1, \ldots, x_n) = \prod_j f_j$$

for some functions f_j whose arguments are the variable sets X_{f_j}. We omit the arguments X_{f_j} of each factor f_j from our formulas. In a common abuse of notation, we use the same symbol to denote a variable (resp., factor) node and its associated variable x_i (resp., factor f_j). We assume that each variable x_i takes on a finite set of values.

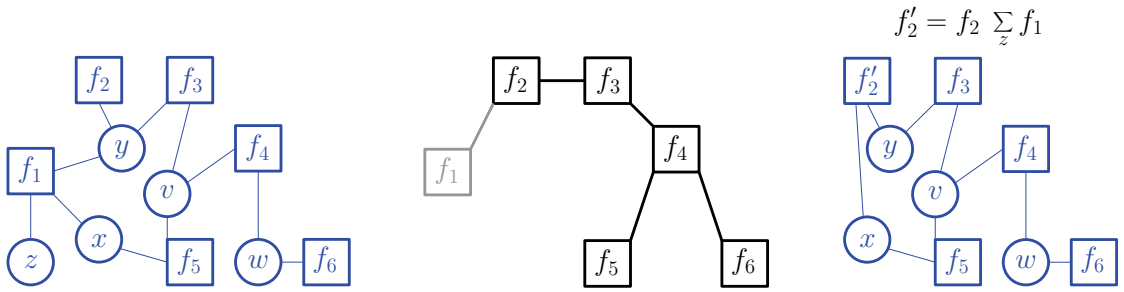
In this paper we first study the problem of marginalization of the function $g(X)$. Specifically, for any x_i we are interested in computing the marginal function

$$g^i(x_i) = \sum_{X \setminus x_i} g(X).$$

Once we establish the basic results for performing adaptive inference, we will also show how our methods can be applied to another commonly studied inference problem, that of finding the configuration of the variables that maximizes g, that is,

$$X^* = \arg\max_X g(X).$$

In this paper, we call the vector X^* the *maximum a posteriori* (MAP) configuration of X.



(a) G_t (b) T_t (c) G_{t+1}

Figure 1: *Factor elimination*. Factor elimination takes a factor graph G_1 and an elimination tree T_1 as input and sequentially eliminates the leaf factors in the elimination tree. As an example, to eliminate f_1 in iteration t, we first marginalize out any variables that are only adjacent to the eliminated factor, and then propagate this information to the unique neighbor in T_t, that is, $f_2' = f_2 \sum_z f_1$.

2.1 Factor Elimination

There are various essentially equivalent algorithms proposed for solving marginalization problems, including belief propagation (Pearl, 1988) or sum-product (Kschischang et al., 2001) for tree-structured graphs, or more generally bucket elimination (Dechter, 1998), recursive conditioning (Darwiche and Hopkins, 2001), junction-trees (Lauritzen and Spiegelhalter, 1988) and factor elimination (Darwiche, 2009). The basic structure of these algorithms is iterative; in each iteration partial marginalizations are computed by eliminating variables and factors from the graph. The set of variables and factors that are eliminated at each iteration is typically guided by some sort of auxiliary structure on either variables or factors. For example, the sum-product algorithm simply eliminates variables starting at leaves of the input factor graph. In contrast, factor elimination uses an *elimination tree* T on the factors and eliminates factors starting at leaves of T; an example elimination tree is shown in Figure 1b.

For a particular factor f_j, the basic operation of *factor elimination* eliminates f_j in the given model and then propagates information associated with f_j to neighboring factors. At iteration t, we pick a leaf factor f_j in T_t and eliminate it from the elimination tree forming T_{t+1}. We also remove f_j along with all the variables $\mathcal{V}_j \subseteq X$ that appear only in factor f_j from G_t forming G_{t+1}. Let f_k be f_j's unique neighbor in T_t. We then partially marginalize f_j, and update the value of f_k in G_{t+1} and T_{t+1} with

$$\lambda_j = \sum_{\mathcal{V}_j} f_j, \qquad\qquad f_k' = f_k \lambda_j.$$

For reasons that will be explained in Section 3.1, we use the notation λ_i to represent the partially marginalized functions; for standard factor elimination these operations are typically combined into a single update to f_k. Finally, since multiplying by λ_j may make f_k' depend on additional variables, we expand the argument set of f_k' by making the arguments of λ_j adjacent to f_k' in G_{t+1}, that is, $X_{f_k'} := X_{f_k} \cup X_{f_j} \setminus \mathcal{V}_j$. Figure 1 gives an example where we apply factor elimination to a leaf factor f_1 in the elimination tree. We marginalize out the variables that are only adjacent to f_1 (i.e., $\mathcal{V}_1 = \{z\}$)

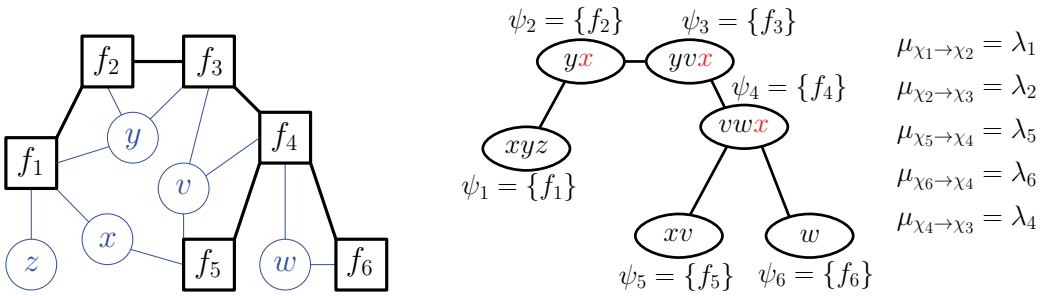


Figure 2: *Factor trees and tree decompositions*. A tree-decomposition (right) that is equivalent to a given elimination tree (left) can be obtained by first replacing each factor with a hyper-node that contains the variables adjacent to that factor node and then adding variables to the hyper-nodes so that the running intersection property is satisfied.

and update f_1's neighbor f_2 in the elimination tree with $f_2' = f_2 \sum_{\mathcal{V}_1} f_1$. Finally, we add an edge between the remaining variables $X_{f_1} \setminus \mathcal{V}_1 = \{x\}$ and the updated factor f_2'.

Suppose we wish to compute a particular marginal $g^i(x_i)$. We root the elimination tree at a factor f_j such that $x_i \sim_G f_j$, then eliminate leaves of the elimination tree one at a time, until only one factor remains. By definition the remaining factor f_j' corresponds to f_j multiplied by the results of the elimination steps. Then, we have that $g^i(x_i) = \sum_{X \setminus x_i} f_j'$. All of the marginals in the factor graph can be efficiently computed by re-rooting the tree and reusing the values propagated during the previous eliminations.

Factor elimination is equivalent to bucket (or variable) elimination (Kask et al., 2005; Darwiche, 2009) in the sense that we can identify a correspondence between the computations performed in each algorithm. In particular, the factor elimination algorithm marginalizes out a variable x_i when there is no factor left in the factor graph that is adjacent to x_i. Therefore, if we consider the operations from the variables' point of view, this sequence is also a valid bucket (variable) elimination procedure. With a similar argument, one can also interpret any bucket elimination procedure as a factor elimination sequence. In all of these algorithms, while marginal calculations are guaranteed to be correct, the particular auxiliary structure or ordering determines the worst-case running time. In the following section, we analyze the performance consequences of imposing a particular elimination tree.

2.2 Viewing Elimination Trees as Tree-decompositions

For tree-structured factor graphs, the typical choice for the elimination tree is based on the factor graph itself. However, when the input factor graph is not tree-structured, we must choose an elimination ordering that ensures that the propagation of variables over the course of elimination is not too costly. In this section, we outline how a particular elimination tree can be related to a tree decomposition on the input graph (e.g., as in Darwiche and Hopkins, 2001 and Kask et al., 2005), thereby allowing us to use the quality of the associated tree decomposition as a measure of quality for elimination trees. In subsequent sections, this relationship will enable us to compare the constant-factor overhead associated with our algorithm against that of the original input elimination tree.

Let $G = (X, F)$ be a factor graph. A *tree-decomposition* for G is a triplet $(\chi, \psi, \mathcal{D})$ where $\chi = \{\chi_1, \chi_2, \ldots, \chi_m\}$ is a family of subsets of X and $\psi = \{\psi_1, \psi_2, \ldots, \psi_m\}$ is a family of subsets of F such that $\cup_{f \in \psi_i} X_f \subseteq \chi_i$ for all $i = 1, 2, \ldots, m$ and $\mathcal{D}$ is a tree whose nodes are the subsets χ_i satisfying the following properties:

1. *Cover property:* Each variable x_i is contained in some subset belonging to χ and each factor $f_j \in F$ is contained in exactly one subset belonging to ψ.
2. *Running Intersection property:* If $\chi_s, \chi_t \in \chi$ both contain a variable x_i, then all nodes χ_u of the tree in the (unique) path between χ_s and χ_t contain x_i as well. That is, the nodes associated with vertex x_i form a connected sub-tree of $\mathcal{D}$.

Any factor elimination algorithm can be viewed in terms of a message-passing algorithm in a tree-decomposition. For a factor graph G, we can construct a tree decomposition $(\chi, \psi, \mathcal{D})$ that corresponds to an elimination tree $T = (F, E)$ on G. First, we set $\psi_i = \{f_i\}$ and $\mathcal{D} = (\chi, E')$ where $(\chi_i, \chi_j) \in E'$ is an edge in the tree-decomposition if and only if $(f_i, f_j) \in E$ is an edge in the elimination tree T. We then initialize $\chi = \left\{X_{f_1}, X_{f_2}, \ldots, X_{f_m}\right\}$ and add the minimal number of variables to each set χ_j so that the running intersection property is satisfied. By construction, the final triplet $(\chi, \psi, \mathcal{D})$ satisfies all the conditions of a tree-decomposition. This procedure is illustrated in Figure 2. The factor graph (light edges) and its elimination tree (bold edges) on the left is equivalent to the tree-decomposition on the right. We first initialize $\chi_j = X_{f_j}$ for each $j = 1, \ldots, 6$ and add necessary variables to sets χ_j to satisfy the running intersection property: x is added to χ_2, χ_3 and χ_4. Finally, we set $\psi_j = \left\{f_j\right\}$ for each $j = 1, \ldots, 6$.

Using a similar procedure, it is also possible to obtain an elimination tree equivalent to the messages passed on a given tree-decomposition. We define two messages for each edge (χ_i, χ_j) in the tree decomposition: the message $\mu_{\chi_i \to \chi_j}$ from χ_i to χ_j is the partial marginalization of the factors on the χ_i side of $\mathcal{D}$, and the message $\mu_{\chi_j \to \chi_i}$ from χ_j to χ_i is the partial marginalization of the factors on the χ_j side of $\mathcal{D}$. The outgoing message $\mu_{\chi_i \to \chi_j}$ from χ_i can be computed recursively using the incoming messages $\mu_{\chi_k \to \chi_i}$ except for $k = j$, that is,

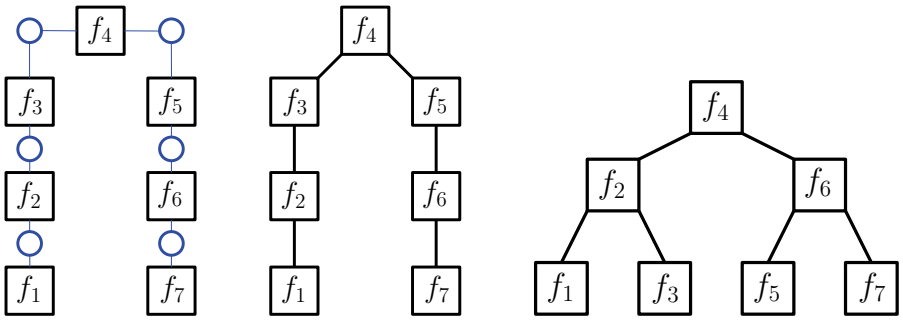
$$\mu_{\chi_i \to \chi_j} = \sum_{\chi_j \setminus \chi_i} f_i \prod_{(\chi_k, \chi_i) \in E' \setminus \left\{(\chi_j, \chi_i)\right\}} \mu_{\chi_k \to \chi_i}. \tag{1}$$

The factor elimination process can then be interpreted as passing messages from leaves to parents in the corresponding tree-decomposition. The partial marginalization function λ_i computed during the elimination of f_i is identical to the message $\mu_{\chi_i \to \chi_j}$ where f_j is the parent of f_i in the elimination tree. This equivalence is illustrated in Figure 2 where each partial marginalization function λ_j is equal to a sum-product message $\mu_{\chi_j \to \chi_k}$ for some k. This example assumes that f_3 is eliminated last.

For an elimination tree T, suppose that the corresponding tree decomposition is $(\chi, \psi, \mathcal{D})$. For the remainder of this paper, we will define the *width* of T to be the size of the largest set contained in χ minus 1. Inference performed using T incurs a constant-factor overhead that is exponential in its width; for example, computing marginals using an elimination tree T of width w takes $O(d^{w+1} \cdot n)$ time and space where n is the number of variables and d is the domain size.

3. Computing Marginals with Deferred Factor Elimination

When performing inference with factor elimination, one typically attempts to select an elimination tree to minimize its associated width. However, such an elimination ordering may not be optimal



(a) Chain graph (b) Unbalanced elimination tree (c) Balanced elimination tree

Figure 3: *Balanced and unbalanced elimination trees*. For the chain factor graph in (a), the elimination tree in (b) has width 1 but requires $O(n)$ steps to propagate information from leaves to the root. The balanced elimination tree in (c), for the same factor graph, has width 2 but takes only $O(\log n)$ steps to propagate information from a leaf to the root, since f_3 and f_5 are eliminated earlier. If f_1 is modified, then using a balanced elimination tree, we only need to update $O(\log n)$ elimination steps, while an unbalanced tree requires potentially $O(n)$ updates.

for repeated inference tasks. For example, an HMM typically used for sequence analysis yields a chain-structured factor graph as shown in Figure 3a. The obvious elimination tree for this graph is also chain-structured (Figure 3b). While this elimination tree is optimal for a single computation, suppose that we now modify the leaf factor f_1. Then, recomputing the marginal for the leaf factor f_7 requires time that is linear in the size in the model, even though only a single factor has changed. However, if we use the *balanced* elimination tree shown in Figure 3c, we can compute the marginalization for f_7 in time that is logarithmic in the size of the model. While the latter elimination tree increases the width by one (increasing the dependence on d), for fixed d and as n grows large we can achieve a significant speedup over the unbalanced ordering if we wish to make changes to the model.

In this section we present an algorithm that generates a logarithmic-depth representation of a given elimination tree. Our primary technique, which we call *deferred factor elimination*, generalizes factor elimination so that it can be applied to non-leaf nodes in the input elimination tree. Deferred factor elimination introduces ambiguity, however, since we cannot determine the "direction" that a factor should be propagated until one of its neighbors is also eliminated. We refer to the local information resulting from each deferred factor elimination as a *cluster function* (or, more succinctly, as a *cluster*), and store this information along with the balanced elimination tree. We use the resulting data structure, which we call a *cluster tree*, to perform marginalization and efficiently manage structural and parameter updates. Pseudocode is given in Figure 4.

For our algorithm, we assume that the user provides both an input factor graph G and an associated elimination tree T. While the elimination tree is traditionally computed from an input model, in an adaptive setting it may be desirable to change the elimination tree to take advantage of changes made to the factors (see Figure 9 for an example). Furthermore, domain-specific knowledge of the changes being made to the model may also inform how the elimination tree should be chosen and

DeferredFactorElimination(G, T, f_j)

Compute cluster λ_j using Equation (3)
if f_j is a leaf in elimination tree T
    Let f_k be f_j's unique neighbor in T
    Attach λ_j to f_k in T
end if
if f_j is a degree-2 node in T
    Let f_i and f_k be f_j's neighbors in T
    Create a new edge (f_i, f_k) in T
    Attach λ_j to the newly created edge (f_i, f_k)
endif
Remove factor f_j from factor graph G and T
for each variable x_i that is connected to only f_j in G
    Remove x_i from G
endfor

Figure 4: *Deferred factor elimination*. In addition to eliminating leaves, deferred factor elimination also eliminates degree-two nodes. This operation can be simultaneously applied to an independent set of leaves and degree-two nodes.

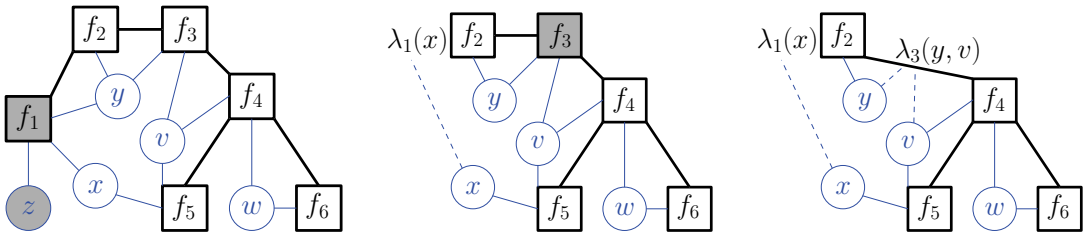
updated. Thus, in the remainder of the paper we separate the discussion of updates applied to the input model from updates that are applied to the input elimination tree. As we will see in Section 4, the former prove to be relatively easy to deal with, while the latter require a reorganization of the cluster tree data structure.

3.1 Deferred Factor Elimination and Cluster Functions

Consider the elimination of a degree-two factor f_j, with neighbors f_i and f_k in the given elimination tree. We can perform a partial marginalization for f_j to obtain λ_k, but cannot yet choose whether to update f_i or f_k—whichever is eliminated first will need λ_k for its computation. To address this, we define *deferred factor elimination*, which removes the factor f_j and saves the partial marginalization λ_j as a *cluster*, leaving the propagation step to be decided at a later time. In this section, we show how deferred factor elimination can be performed on the elimination tree, and how the intermediate cluster information can be saved and also used to efficiently compute marginals.

For convenience, we will segregate the process of deferred factor elimination on the input model into rounds. In a particular round t ($1 \leq t \leq n$), we begin with a factor graph G_t and an elimination tree T_t, and after performing some set of deferred factor eliminations, we obtain a resulting factor graph G_{t+1} and elimination tree T_{t+1} for the next round. For the first round, we let $G_1 = G$ and $T_1 = T$. Note that since each factor is eliminated exactly once, the number of total rounds depends on the number of the factors eliminated in each round.

To construct T_{t+1} from T_t, we modify the elimination tree as follows. When we eliminate a degree-one (leaf) factor f_j, we attach λ_j to the neighbor vertex f_k. When a degree-two factor f_j is removed, we attach λ_j to a newly created edge (f_i, f_k) where f_i and f_k are f_j's neighbors in elimination tree T. We define $\mathcal{C}_T(f_j)$ to be the set of clusters that are attached either directly to f_j or



(a) T_1 and G_1 (b) T_2 and G_2 (c) T_3 and G_3

Figure 5: *Deferred factor elimination*. (a) An elimination tree T_1 (bold elges), with variable dependencies shown with light edges for reference. To eliminate a leaf node f_1, we sum out variables that are not attached to any other factors (shaded), resulting in the cluster function λ_1 and new elimination tree T_2 in (b). To eliminate a degree-two node f_3, we replace it with λ_3 attached to the edge (f_2, f_4), giving tree T_3 shown in (c).

to an edge incident to f_j. In the factor graph G_{t+1}, we remove all $\lambda_k \in \mathcal{C}_{T_t}(f_j)$ and variables $\mathcal{V}_j \subset X$ that do not depend on any factors other than f_j or $\lambda_k \in \mathcal{C}_{T_t}(f_j)$. Finally, we replace f_j with λ_j, given by

$$\lambda_j = \sum_{\mathcal{V}_j} f_j \prod_{\lambda_k \in \mathcal{C}_{T_t}(f_j)} \lambda_k. \tag{2}$$

The cluster λ_j is referred as a *root cluster* if $\deg_{T_t}(f_j) = 0$, a *degree-one cluster* if $\deg_{T_t}(f_j) = 1$, and a *degree-two cluster* if $\deg_{T_t}(f_j) = 2$. Figure 5 illustrates the creation of degree-one and degree-two clusters, and the associated changes to the elimination tree and factor graph. We first eliminate f_1 by replacing it with degree-one cluster $\lambda_1(x) = \sum_z f_1(x)$. Cluster λ_1 is attached to factor f_2 and the set of clusters around f_2 is $\mathcal{C}_{T_2}(f_2) = \{\lambda_1, \lambda_3\}$. We then eliminate a degree-two factor f_3 by replacing it with degree-two cluster $\lambda_3(y,v) = f_3(y,v)$. This connects f_2 to f_4 in the elimination tree, and places λ_3 on the newly created edge.

We note that the correctness of deferred factor elimination follows from the correctness of standard factor elimination. To perform marginalization for any particular variable, we can simply instantiate a series of propagations, at each step using a cluster function that has already been computed in one of the aforementioned rounds.

To establish the overall running time of deferred factor elimination we first explain how the clusters we compute can be interpreted in the tree-decomposition framework. Recall that in Section 2.2, we established an equivalence between clusters and messages in the tree-decomposition in the case where only leaf factors in the elimination tree are eliminated. We can generalize this relationship to the case where degree-two factors are also eliminated. As discussed earlier in Section 2.2, the equivalent tree-decomposition $(\chi, \psi, \mathcal{D})$ of an elimination tree $T = (F, E)$ consists of a tree $\mathcal{D}$ on hyper-nodes $\chi = \{\chi_1, \ldots, \chi_m\}$ with the same adjacency relationship with the factors $\{f_1, \ldots, f_m\}$ in T.

A degree-one cluster λ_j produced after eliminating a leaf f_j factor in T is a partial marginalization of the factors on a sub-tree of T. Let f_k be f_j's unique neighbor in the elimination tree when it is eliminated. This implies $\lambda_j = \mu_{\chi_t \to \chi_k}$ for some t as previously shown in Section 2.2. Note that the index t may not equal j, since there may be a cluster attached to the edge (f_j, f_k) (for example in Figure 5, $\lambda_1(x) = \mu_{\chi_1 \to \chi_2}(x)$).

A degree-two cluster λ_j produced after eliminating a degree-two factor f_j in T is a partial marginalization of the factors in a connected subgraph $S \subset T$ such that S and $T \setminus S$ are connected by exactly two edges. Let (f_i, f_c) and (f_d, f_k) be these edges, where f_c and f_d belong to S and f_i and f_k are outside of S (we will show how these "boundary" edges can be efficiently computed in Section 3.2). We interpret λ_j as an intermediary function that enables us to compute an outgoing message $\mu_{\chi_d \to \chi_k}$ by using only λ_j and the incoming message $\mu_{\chi_j \to \chi_c}$, that is, $\mu_{\chi_d \to \chi_k} = \sum_{\chi_k \setminus \chi_j} \lambda_i \mu_{\chi_j \to \chi_c}$. These intermediate functions are in fact the mechanism that allows us avoid long sequences of message passing. For example in Figure 5, λ_3 can be used to compute the message $\mu_{\chi_3 \to \chi_4}$ using only $\mu_{\chi_2 \to \chi_3}$, that is, $\mu_{\chi_3 \to \chi_4}(x, v) = \sum_y \mu_{\chi_2 \to \chi_3}(x, y) \lambda_3(y, v)$.

Finally, we note that we have a single root cluster that is just a marginalization of all of the factors in the factor graph. Using the relationships established above between cluster functions and messages in a tree decomposition, we give the running time of deferred factor elimination on a given elimination tree and input factor graph.

Lemma 1 *For an elimination tree with width w, the elimination of leaf factors takes $\Theta(d^{2w})$ time and produces a cluster of size $\Theta(d^w)$, where d is the domain size of the variables in the input factor graph. The elimination of degree-two vertices takes $\Theta(d^{3w})$ time and produces a cluster of size $\Theta(d^{2w})$.*

Proof Each degree-one cluster has size $O(d^w)$ because it is equal to a sum-product message in the equivalent tree-decomposition. For a degree-two vertex f_j, the cluster λ_j can be interpreted as an intermediary function that enables us to compute the outgoing messages $\mu_{\chi_c \to \chi_i}$ and $\mu_{\chi_d \to \chi_k}$ using the incoming messages $\mu_{\chi_k \to \chi_d}$ and $\mu_{\chi_i \to \chi_c}$ for some χ_c, χ_d, χ_i and χ_k where f_i and f_k are neighbors of f_j in the elimination tree during its elimination. The set of variables involved in these computations is $(\chi_i \cap \chi_c) \cup (\chi_k \cap \chi_d)$ which is bounded by $2w$. Hence, the cluster f_i that computes the partial marginalization of the factors that are between (f_d, f_k) and (f_i, f_c) has size $O(d^{2w})$. Moreover, these bounds are achieved if $\chi_i \cap \chi_c$ and $\chi_k \cap \chi_k$ are disjoint and each has w variables.

We now establish the running times of calculating cluster functions, by bounding the number of variables involved in computing a cluster. We first show that when a leaf node f_j is eliminated, the set of variables involved in the computation is $\chi_j \cup \chi_k$ where f_k is f_j's neighbor. For all the degree-one clusters of f_j, their argument set is a subset of χ_j, so the product in Equation (2) can be computed in $O(d^w)$ time. There can be a cluster λ_c on the edge (f_j, f_k) whose argument set has to be subset of $\chi_j \cup \chi_k$. If there is such a cluster, the cost of computing the product in Equation (2) becomes $O(d^{2w})$. This bound is achieved when there is a degree-two cluster and χ_j and χ_k are disjoint.

When a degree-two factor f_j is eliminated, the set of variables involved in the computation is $\chi_i \cup \chi_j \cup \chi_k$ where f_i and f_k are neighbors of f_j. As shown above, the argument set of degree-one clusters is a subset of χ_j. This cluster can have degree-two clusters on edges (f_i, f_j) and (f_j, f_k), and in this case, computation of a degree-two cluster takes $O(d^{3w})$ time. This upper bound is achieved when the sets χ_i, χ_j and χ_k are disjoint.

We note that in the above discussion we assumed that the number of operands in Equation (2) is bounded, that is, for any factor f, $|C_T(f)| = O(1)$. This assumption is valid because for any given elimination tree, we can construct an equivalent elimination tree with degree 3 by adding dummy factors. For example, suppose the input elimination tree has degree $n-1$ (i.e., it is star-shaped); then Equation (2) has n multiplication operands hence requires $O(nd^w)$ time to compute. By adding dummy factors in the shape of a complete binary tree between the center factor and the leaf factors,

BuildClusterTree(G, T)

$G_0 := G, T_0 := T$
Initialize $\mathcal{H}$ as an empty rooted tree
for round t = 1 up to k
    $G_t := G_{t-1}, T_t := T_{t-1}$
    S := A maximal independent set of leaves and degree two nodes in T_t
    for each factor f_j in S
        call *DeferredFactorElimination*(G_t, T_t, f_j)
        for each cluster λ_i that is used to compute λ_j
            Add edge (λ_i, λ_j) in $\mathcal{H}$ where λ_j is the parent.
        endfor
        for each variable x_i eliminated along with f_j
            Add edge (x_i, λ_j) in $\mathcal{H}$ where λ_j is the parent
        endfor
    endfor
endfor
return $\mathcal{H}$ as the cluster tree

Figure 6: *Hierarchical clustering*. Using deferred factor elimination, we can construct a balanced cluster tree data structure that can be used for subsequent marginal queries.

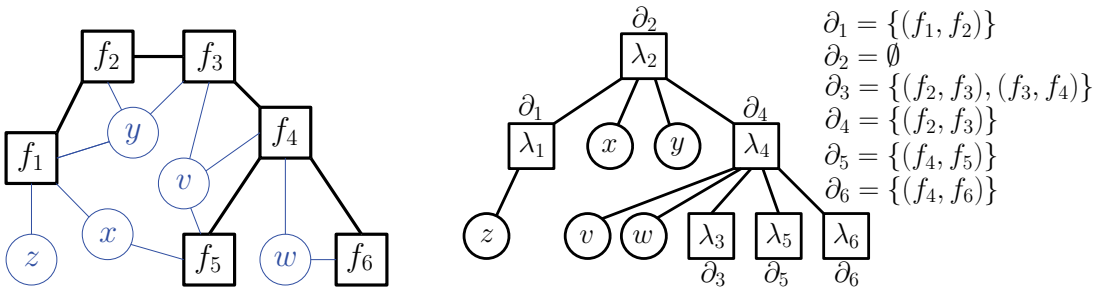
we can bring the complexity of computing Equation (2) down to $O(d^w)$ for each factor. ■

3.2 Constructing a Balanced Cluster Tree

In this section, we show how performing deferred factor elimination in rounds can be used to create a data structure we call a *cluster tree*. As variables and factors are eliminated through deferred factor elimination, we build the cluster tree using the dependency relationships among clusters (see Figure 6). The cluster tree can then be used to compute marginals efficiently, and as we will see, it can also be used to efficiently update the original factor graph or elimination tree.

For a factor graph $G = (X, F)$ and an elimination tree T, a cluster tree $\mathcal{H} = (X \cup C, E)$ is a rooted tree on variables and clusters $X \cup C$ where C is the set of clusters. The edges E represent the dependency relationships among the quantities computed while performing deferred factor elimination. When a factor f_j is eliminated, cluster λ_j is produced by Equation (2). All the variables $\mathcal{V}_j$ and clusters $\mathcal{C}(f_j)$ removed in this computation become λ_j's children. For a cluster λ_j, the *boundary* ∂_j is the set of edges in T that separates the collection of factors that is contracted into λ_j from the rest of the factors.

In Equation (2), we gave a recursive formula to compute λ_j in terms of its children in the cluster tree. In order to use the cluster tree in our computations, we need to derive a similar recursive formula for the boundary ∂_j for each cluster λ_j. Let clusters $\lambda_1, \lambda_2, \ldots, \lambda_k$ and variables $x_1, x_2, \ldots, x_t$ be λ_j's children in the cluster tree. Let $E(f_j)$ be the set of edges incident to f_j in T. Then the



(a) Factor Graph G (b) Cluster Tree $\mathcal{H}$

Figure 7: *Cluster Tree Construction*. To obtain the cluster tree in (b), eliminations are performed in the factor graph G (a) in the following order: f_1, f_3, f_5 and f_6 in round 1, f_4 in round 2 and f_2 in round 3. The cluster-tree (b) representing this elimination is annotated by boundaries.

boundary of λ_j can be computed by

$$\partial_j = E(f_j) \triangle \partial_1 \triangle \partial_2 \triangle \ldots \triangle \partial_k$$

where ∂_i is the boundary of cluster λ_i and $\triangle$ is the symmetric set difference operator. An example cluster tree, along with explicitly computed boundaries, is given in Figure 7b. For example the boundary of the cluster λ_4 is computed by $\partial_4 = E(f_4) \triangle \partial_3 \triangle \partial_5 \triangle \partial_6$ where $E(f_4) = \{(f_2, f_4), (f_4, f_5), (f_4, f_6)\}$.

Theorem 2 *Let $G = (X, F)$ be a factor graph with n nodes and T be an elimination tree on G with width w. Constructing a cluster tree takes $\Theta(d^{3w} \cdot n)$ time.*

Proof During the construction of the cluster tree, every factor is eliminated once. By Lemma 1, each such elimination takes $O(d^{3w})$ time. ■

For our purposes it is desirable to perform deferred factor elimination so that we obtain a cluster tree with logarithmic depth. We call this process *hierarchical clustering* and define it as follows. We start with $T_1 = T$ and at each round i we identify a set K of degree-one or -2 factors in T_i and apply deferred factor elimination to this independent set of factors to construct T_{i+1}. This procedure ends once we eliminate the last factor, say f_r. We make λ_r the root of the cluster tree. At each round, the set $K \subset F$ is chosen to be a maximal independent set, that is, for $f_i, f_j \in K$, $f_i \not\sim f_j$ in T, and no other factor f_k can be added to K without violating independence. The sequence of elimination trees created during the hierarchical clustering process will prove to be useful in Section 4, when we show how to perform structural updates to the elimination tree. As an example, a factor graph G, along with its associated elimination tree $T = T_1$, is given in Figure 7a. In round 1, we eliminate a maximal independent set $\{f_1, f_3, f_5, f_6\}$ and obtain T_2. In round 2 we eliminate f_4, and finally in round 3 we eliminate f_2. This gives us the cluster tree shown in Figure 7b.

As we show with the following lemma, the cluster tree that results from hierarchical clustering has logarithmic depth. We will make use of this property throughout the remainder of the paper to establish the running times for updating and computing marginals and MAP configurations.

QueryMarginal($\mathcal{H}$, x_i)

Let $x_i, \lambda_1, \ldots, \lambda_k$ be the path from x_i to the root λ_k of cluster tree $\mathcal{H}$
for j = k down to 1
    Let f_j be the factor associated with cluster λ_j
    Compute downward marginalization function M_{f_j} using Equation (4)
endfor
Compute the marginal at x_i using Equation (5)

Figure 8: *Performing Marginalization with a Cluster Tree*. Computing any particular marginal in the input factor graph corresponds to a root-to-leaf path in the cluster tree.

Lemma 3 *For any factor graph $G = (X,F)$ with n nodes and any elimination tree T, the cluster tree obtained by hierarchical clustering has depth $O(\log n)$.*

Proof Let the elimination tree $T = (F,E)$ have a leaves, b degree-two nodes and c degree-3 or more nodes, that is, $m = a+b+c$ where m is the number of factors. Using the fact that the sum of the degrees of the vertices is twice the number of edges, we get $2|E| \geq a+2b+3c$. Since a tree with m vertices have $m-1$ edges, we get $2a+b-2 \geq m$. On the other hand, a maximal independent set of degree-one and degree-two vertices must have size at least $a-1+(b-a)/3 \geq m/3$, since we can eliminate at least a third of the degree-two vertices that are not adjacent to leaves. Therefore at each round, we eliminate at least a third of the vertices, which in turn guarantees that the depth of the cluster tree is $O(\log n)$. ■

3.3 Computing Marginals

Once a balanced cluster tree $\mathcal{H}$ has been constructed from the input factor graph and elimination tree, as in standard approaches we can compute the marginal distribution of any variable x_i by propagating information (i.e., partial marginalizations) through the cluster tree. For any fixed variable x_i, let $\lambda_1, \lambda_2, \ldots, \lambda_k$ be the sequence from x_i to the root λ_k in the cluster tree $\mathcal{H}$. We now describe how to compute the marginal for x_i (see Figure 8 for pseudocode). For each factor f_j, let ∂_j contain neighbors f_a and f_b of f_j (i.e., neighboring factors at the time f_j is eliminated). This information can be obtained easily, since f_a and f_b are ancestors of f_j in the cluster tree, that is, $f_a, f_b \in \{f_{j+1}, f_{j+2}, \ldots, f_k\}$. For convenience we state our formulas as if there are two neighbors in the boundary; in the case of degree-one clusters, terms associated with one of the neighbors, say f_b, can be ignored in the statements below. First, we compute a downward pass of marginalization functions from λ_k to λ_1 given by

$$M_{f_j} = \sum_{Y \setminus X_{\lambda_j}} f_j M_{f_a} M_{f_b} \prod_{f \in C_j \setminus \{f_{j-1}\}} f, \tag{3}$$

where Y is the set of variables that appear in the summands and X_{λ_j} is the set of variables that cluster λ_j depends on. Therefore each marginalization function M_j from parent λ_j is computed using only

information in the path above λ_j. Then, the marginal for variable x_i is

$$g^i(x_i) = \sum_{Y \setminus \{x_i\}} M_{f_1} \prod_{f \in C_1} f \tag{4}$$

where Y is the set of variables that appear in the summands. Combining this approach with Lemmas 1 and 3, we have the following theorem.

Theorem 4 *Consider a factor graph G with n nodes and let T be an elimination tree with width w. Then, Equation (4) holds for any variable x_i and can be computed in $O(d^{2w} \log n)$ time.*

Proof The correctness of Equation (4) follows when each marginalization function M_{f_j} is viewed as a sum-product message in the equivalent tree-decomposition. To prove the latter, we will show that for $\partial_j = \{(f_c, f_a), (f_d, f_b)\}$, M_{f_a} and M_{f_b} are equal to the tree-decomposition messages $\mu_{\chi_a \to \chi_c}$ and $\mu_{\chi_b \to \chi_d}$, respectively. This can be proven inductively starting with M_{f_k}. First, note that the base case holds trivially. Then, using the inductive hypothesis, we assume that $M_{f_a} = \mu_{\chi_a \to \chi_c}$ and $M_{f_b} = \mu_{\chi_b \to \chi_d}$. Now, there has to be a descendant λ_ℓ of λ_j such that $(f_e, f_j) \in \partial_\ell$. By multiplying with the degree-two clusters in $C_j \setminus \{f_{j-1}\}$, we can convert the messages $\mu_{\chi_a \to \chi_c}$ and $\mu_{\chi_b \to \chi_d}$ to the messages into f_j. Applying Equation (1) then gives $M_{f_j} = \mu_{\chi_j \to \chi_e}$ as desired.

For the running time, we observe that each message computation is essentially the same procedure as eliminating a leaf factor, therefore each message has size $O(d^w)$ and takes $O(d^{2w})$ time to compute by Lemma 1. ■

We note that it is also possible to speed-up successive marginal queries by caching the downward marginalization functions in Equation (3). For example, if we query all variables as described above, we compute $O(n \log n)$ many downward marginalization messages. However, by caching the downward marginalization functions in the cluster tree, we can compute all marginals in $O(d^{2w} \cdot n)$ time, which is optimal given the elimination ordering. As we will see in Section 4.1, the balanced nature of the cluster tree allows us to perform batch operations efficiently. In particular, for marginal computation, using the caching strategy above, any set of ℓ marginals can be computed in $O(d^{2w} \ell \log(n/\ell))$ time.

4. Updates

The preceding sections described the process of constructing a balanced, cluster tree elimination ordering from a given elimination tree, and how to use the resulting cluster tree to compute marginal distributions. However, the primary advantage of a balanced ordering lies in its ability to adapt to changes and incorporate updates to the model. In this section, we describe how to efficiently update the cluster tree data structure after changes are made to the input factor graph or elimination tree.

We divide our update process into two algorithmic components. We first describe how to make changes to the factors, whether changing the parameters of the factor or its arguments (and thus the structure of the factor graph), but leaving the original elimination tree (and thus the cluster tree) fixed. We then describe how to make changes to the elimination tree and efficiently update the cluster tree. In practice these two operations may be combined; for example when modifying a tree-structured graph such that it remains a tree we are likely to change the elimination tree to reflect the new structure. Similarly, for a general input factor graph we may also wish to change the

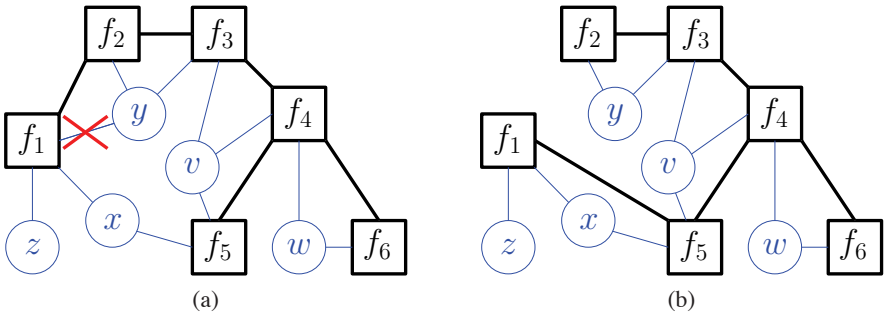


Figure 9: *Modifying the Elimination Tree*. If the factor graph in (a) is modified by removing the edge (y, f_1), we can reduce the width of the elimination tree (from 3 to 2) by replacing the edge (f_1, f_2) by (f_1, f_5).

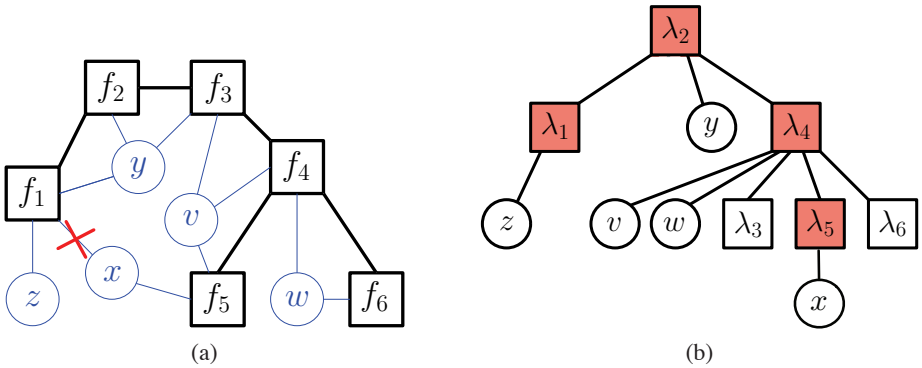


Figure 10: *Modifying the arguments of factors*. If the factor graph in (a) is modified by removing the edge (x, f_1), we update two paths in the cluster tree, as shown in (b), from both x and λ_1 to the root. The position in which x is eliminated is found by bottom-up traversing of the factors adjacent to x.

elimination tree upon changes to factors. Figure 9 illustrates such an example, in which changing a dependency in the factor graph makes it possible to reduce the width of the elimination tree.

4.1 Updating Factors With a Fixed Elimination Tree

For a fixed elimination tree, suppose that we change the parameters of a factor f_j (but not its arguments), and consider the new cluster tree created for the resulting graph. As suggested in the discussion in Section 3, the first change in the clustering process occurs when computing λ_j; a change to λ_j changes its parent, and so on upwards to the root. Thus, the number of affected functions that need to be recalculated is at most the depth of the cluster tree. Since the cluster tree is of depth $O(\log n)$ by Lemma 3, and each operation takes at most $O(d^{3w})$, the total recomputation is at most $O(d^{3w} \log n)$.

If we change the structure of graph G by modifying the arguments of a factor f_j by adding or removing some variable x_i, then the point at which x_i is removed from the factor graph may

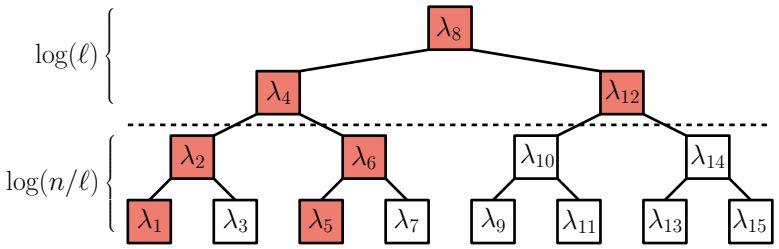


Figure 11: *Batch updates*. After modifying $\ell = 3$ factors, f_1, f_5 and f_{12}, we update the corresponding clusters and their ancestors in a bottom-up fashion. The total number of nodes visited is $O(\ell \log(\frac{n}{\ell}) + 2^{\log(\ell)}) = O(\ell \log(\frac{n}{\ell}))$.

also change. Since x_i is eliminated (i.e., summed out) once every factor that depends on it has been eliminated, adding an edge may postpone elimination, while removing an edge may lead to an earlier elimination. To update the cluster tree as a result of this change, we must update all clusters affected by the change to f_j, and we must also identify and update the clusters affected by earlier, or later, removal of x_i from the factor graph. In both edge addition and removal, we can update clusters from λ_j to the root in $O(d^{3w} \log n)$ time.

We describe how to identify the new elimination point for x_i in $O(\log n)$ time. Observe that the original cluster λ_k at which x_i is eliminated is the topmost cluster in the cluster tree with the property that either f_k, the associated factor, depends on x_i, or λ_k has two children clusters that both depend on x_i. The procedure to find the new point of elimination differs for edge insertion and edge removal. First, suppose we add edge (x_i, f_j) to the factor graph. We must traverse upward in the cluster tree until we find the cluster satisfying the above condition. For edge removal, suppose that we remove the dependency (x_i, f_j). Then, x_i can only need to be removed earlier in the clustering process, and so we traverse downwards from the cluster where x_i was originally eliminated. At any cluster λ_k during the traversal, if the above condition is not satisfied then λ_k must have one or no children clusters that depend on x_i. If λ_k has a single child that depends on x_i, we continue traversing in that direction. If λ_k has no children that depend on x_i, then we continue traversing towards λ_j. Note that this latter case occurs only when the paths of x_i and λ_j to the root overlap, and thus is always possible to traverse toward λ_j.

Once we have identified the new cluster at which x_i is eliminated, we can recalculate cluster functions upwards in $O(d^{3w} \log n)$ time. Therefore the total cost of performing an edge insertion or removal $O(d^{3w} \log n)$. Figure 10 illustrates how the cluster tree is updated after deleting an edge in a factor graph keeping the elimination tree fixed. After deleting (x, f_1) we first update the clusters upwards starting from λ_1. Then traverse downwards to find the point at which x_i is eliminated, which is λ_5 because f_5 depends on x. Finally, we update λ_5 and its ancestors.

We can also extend the above arguments to handle multiple, simultaneous updates to the factor graph. Suppose that we make ℓ changes to the model, either to the definition of a factor or its dependencies. Each change results in a set of affected nodes that must be recomputed; these nodes are the ancestors of the changed factor, and thus form a path upwards through the cluster tree. This situation is illustrated in Figure 11. Now, we count the number of affected nodes by grouping them into two sets. If our cluster tree has branching factor b, level $\log_b(\ell)$ has ℓ nodes; above this point,

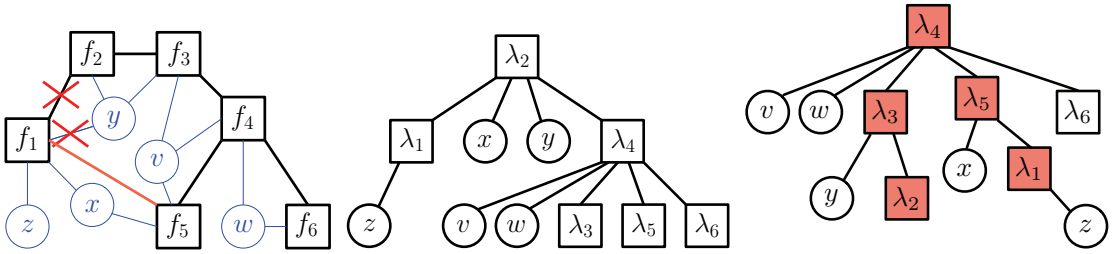


Figure 12: *Updating the elimination tree*. Suppose we modify the input factor graph by removing (y, f_1) from the factor graph and replacing (f_1, f_2) by (f_1, f_5) in the elimination tree as shown in (a). The original cluster tree (b) must be changed to reflect these changes. We must revisit the decisions made during the hierarchical clustering for the affected factors (shaded).

paths *must* merge, and all clusters may need to be recalculated. Below level $\log_b(\ell)$, each path may be separate. Thus the total number of affected clusters is $\ell + \ell \log_b(n/\ell)$.

Note that for edge modifications, we must also address how to find new elimination points efficiently. As stated earlier, any elimination point λ_k for x_i satisfies the condition that it is the topmost cluster in the cluster tree with the property that either f_k depends on x_i, or λ_k has two children clusters that both depend on x_i. As we update the clusters in batch, we can determine the variables for which the above condition is not satisfied until we reach the root cluster. In addition, we also mark the bottommost clusters at which the above condition is not satisfied. Starting from these marked clusters, we search downwards level-by-level until we find the new elimination points. At each step λ_k, we check if there is a variable x_i such that $x_i \not\sim f_j$ and only one child cluster of λ_k depends on x_i. If there is not, we stop the search; if there is, we continue searching towards those clusters. Since each step takes $O(w)$ time, the total time to find all new elimination points is $O(w\ell \log(n/\ell))$. We then update the clusters upwards starting from the new elimination points until the root, which takes $O(d^{3w}\ell \log(n/\ell))$ time.

Combining the arguments above, we have the following theorem.

Theorem 5 *Let $G = (X, F)$ be a factor graph with n nodes and $\mathcal{H}$ be the cluster tree obtained using an elimination tree T with width w. Suppose that we make ℓ changes to the model, each consisting of either adding or removing an edge or modifying the parameters of some factor, while holding T fixed. Then, we can recompute the cluster tree $\mathcal{H}'$ in $O(d^{3w}\ell \log(n/\ell))$ time.*

4.2 Structural Changes to the Elimination Tree

Many changes to the graphical model will be accompanied by some change to the desired elimination ordering. For example, changing the arguments of a factor may suggest some more efficient ordering that we may wish to exploit. However, changing the input elimination order also requires modifying the cluster tree constructed from it. Figure 12 shows such a scenario, where removing a dependency suggests an improved elimination tree. In this section we prove that it is possible to efficiently reorganize the cluster tree after a change to the elimination tree.

As in the previous section, we wish to recompute only those nodes in the cluster tree whose values have been affected by the update. In particular we construct the new cluster tree by stepping through the creation of the original sequence $T_1, T_2, \ldots$, marking some nodes as *affected* if we need to revisit the deferred elimination decision we made in constructing the cluster tree, and leaving the rest unchanged. We first describe the algorithm itself, then prove the required properties: that the original clustering remains valid outside the affected set; that after re-clustering the affected set, our clustering remains a valid maximal independent set and is thus consistent with the theorems in Section 3; and finally that the total affected set is again only of size $O(\log n)$. Since the elimination tree can be arbitrarily modified by performing edge deletions and insertions successively, for ease of exposition we first focus on how the cluster tree can be efficiently updated when a single edge in the elimination tree is inserted or deleted. For the remainder of the section, we assume that the hierarchical clustering process produced intermediate trees $(T_1, T_2, \ldots, T_k)$ and that (f_i, f_j) is the edge being inserted or deleted.

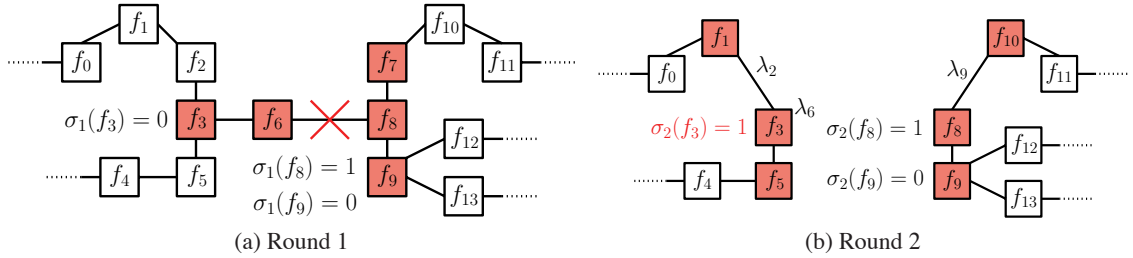
Observe that, to update any particular round of the hierarchical clustering, for any factor f_k we must be able to efficiently determine whether its associated cluster must be recomputed due to the insertion or deletion of an edge (f_i, f_j). A trivial way to check this would be to compute a new hierarchical clustering $(T'_1, T'_2, \ldots, T'_l)$ using the changed elimination tree. Then, the cluster λ_k that is generated after eliminating f_k depends only on the set of clusters around f_k at the time of the elimination. If $\mathcal{C}_i(f_k)$ and $\mathcal{C}'_i(f_k)$ are the set of clusters around f_k on T_i and T'_i, respectively, then f_k is affected at round i if the sets $\mathcal{C}_i(f_k)$ and $\mathcal{C}'_i(f_k)$ are different. Note that we consider $\mathcal{C}_i(f_k) = \mathcal{C}_i(f_k)$ if $\lambda_j \in \mathcal{C}_i(f_k) \iff \lambda_j \in \mathcal{C}'_i(f_k)$ and the values of λ_j are identical in both sets. Clearly, this approach is not efficient, but motivates us to (incrementally) track whether or not $\mathcal{C}_i(f_k)$ and $\mathcal{C}'_i(f_k)$ are identical in a more efficient manner. To do this, we define the *degree-status* of the neighbors of f_k, and maintain it as we update the cluster tree. Given two hierarchical clusterings $(T_1 = (F_1, E_1), T_2 = (F_2, E_2), \ldots, T_k = (F_k, \emptyset))$ and $(T_1 = (F'_1, E'_1), T_2 = (F'_2, E'_2), \ldots, T_l = (F'_l, \emptyset))$, we define the degree-status $\sigma_i(f)$ of a factor f at round i as

$$\sigma_i(f) = \begin{cases} 1 & \text{if} \quad \deg_{T_i}(f) \leq 2 \text{ or } \deg_{T'_i}(f) \leq 2 \text{ or } f \notin F_i \cap F'_i, \\ 0 & \text{if} \quad \deg_{T_i}(f) \geq 3 \text{ and } \deg_{T'_i}(f) \geq 3. \end{cases}$$

The degree status tells us whether f is a candidate for elimination in either the previous or the new cluster tree.

At a high level, we step through the original clustering, marking factors as affected according to their degree-status. For a factor f_j, if $\sigma_i(f_j) = 1$, then f_j is either eliminated or a candidate for elimination at round i in one or both of the previous and new hierarchical clusterings. Since we must recompute clusters for affected factors, if we mark f_j as affected, then its unaffected neighbors should also be marked as affected in the next round. An example is shown in Figure 13. This approach conservatively tracks how affectedness "spreads" from one round to the next; we may mark factors as affected unnecessarily. However, we will be able to show that any round of the new clustering has a constant number of factors for which we must recompute clusters.

We now describe our algorithm for updating a hierarchical clustering after a change to the elimination tree. We first insert or remove the edge (f_i, f_j) in the original elimination tree and obtain $T'_1 = (V'_1, E'_1)$ where $E'_1 = E_1 \cup \{(f_i, f_j)\}$ if the edge is inserted or $E'_1 = E_1 \setminus \{(f_i, f_j)\}$ if deleted. For $i = 1, 2, \ldots, l$, the algorithm proceeds by computing the affected set A_i, an independent set $M_i \subseteq A_i$ of affected factors of degree at most two in T'_i, and then eliminating M_i to form T'_{i+1}. We let $A_0 = \{f_i, f_j\}$, $M_0 = \emptyset$ and $T'_0 = T'_1$. For round $i = 1, 2, \ldots l$ we do the following:



(a) Round 1 (b) Round 2

Figure 13: *Affected nodes in the clustering*. By rule 2 for marking factors as affected, eliminating f_6 in the first round makes $\sigma_2(f_3) = 1$, thereby making f_1 and f_5 affected. In contrast, since $\sigma_2(f_9) = 0$, f_{12} and f_{13} are not marked as affected. By rule 1, eliminating f_7 in the first round makes f_{10} affected.

- We obtain the new elimination tree $T_i' = (F_i', E_i')$ by eliminating the factors in M_{i-1} from T_{i-1} via deferred factor elimination subroutine.
- All affected factors left in T_i' remain affected, namely the set $A_{i-1} \setminus M_{i-1}$. We mark a previously unaffected factor f as affected if
  1. f has an affected neighbor g in T_{i-1}' such that $g \in M_{i-1}$ or
  2. f has an affected neighbor g in T_i' such that $g \in A_{i-1} \setminus M_{i-1}$ with $\sigma_i(g) = 1$.

  Let N_i be the set of factors that are marked in this round according to these two rules, then $A_i = (A_{i-1} \setminus M_{i-1}) \cup N_i$.
- Initialize $M_i = \emptyset$ and greedily add affected factors to M_i starting with the factors that are adjacent to an unaffected factor. Let $f \in A_i$ be an affected factor with an unaffected neighbor $g \in V_i' \setminus A_i$. If g is being eliminated at round i we skip f, otherwise f is included in M_i if $\deg_{T_i'}(f) \leq 2$. We continue traversing the set of affected factors with degree at most two and add as many of them as we can to M_i, subject to the independence condition.

Observe that a factor f in T_i' becomes affected either if an affected neighbor of f is eliminated at round $i-1$ or if f has neighbor that was affected in earlier rounds with degree-status one in T_i'. Once a factor becomes affected, it stays affected. For an unaffected factor f at round i, f's neighbors have to be (i) unaffected, (ii) affected with degree-status zero, or (iii) have become affected at round i.

In order to establish that the procedure above correctly updates the hierarchical clustering, we first prove that we are able to correctly identify unaffected factors, and incrementally maintain maximal independent sets.

Lemma 6 *Given $T = (T_1, T_2, \ldots, T_k)$, let $T' = (T_1', T_2', \ldots, T_l')$ be the updated hierarchical clustering. For any round $i = 1 \ldots l$, let $T_i' = (F_i', E_i')$, let $P_i = F_i' \setminus A_i$ be the set of unaffected factors and $R_i = P_i \setminus F_{i+1}'$ be the ones that are eliminated at round i. Then, the following statements hold:*

- *$R_i \cup M_i$ is a maximal independent set among vertices of degree at most two in F_i'.*
- *For any $f \in P_i$, the set of clusters around f and the set of neighbors of f are the same in T_i as in T_i'.*

Proof For the first claim, we first observe that R_i is an independent since it is contained in M_i. For maximality, assume that $R_i \cup M_i$ is not a maximal independent set among degree ≤ 2 vertices of F_i'. Then there must be a factor f with two neighbors g,h with degrees ≤ 2 and none of which are eliminated at round i. This triplet (f,g,h) cannot be entirely in A_i or $F_i' \setminus A_i$, because the sets R_i and M_i are maximal on their domain, namely R_i is a maximal independent set over $F_i' \setminus A_i$ and M_i is a maximal independent set over A_i. On the other hand, the triplet (f,g,h) cannot be on the boundary either because the update algorithm eliminates any factor with $\deg_{T_i'} \leq 2$ if it is adjacent to an unaffected factor that is not eliminated at round i. Therefore, $R_i \cup M_i$ is a maximal independent set over degree ≤ 2 vertices of F_i'.

We now prove the first part of the second claim by induction on i. Let $\mathcal{C}_i(f)$ and $\mathcal{C}_i'(f)$ be the set of clusters around f in T_i and T_i', respectively. The claim is trivially true for $i = 1$ because $\mathcal{C}_i(f) = \mathcal{C}_i'(f) = \emptyset$ for all factors. Assume that $\mathcal{C}_j(f) = \mathcal{C}_j'(f)$ for all unaffected factors at round j where $j = 1,\ldots,i-1$. Since $f \in P_i$ implies that $f \in P_{i-1}$, we have that $\mathcal{C}_{i-1}(f) = \mathcal{C}_{i-1}'(f)$. Since the set of clusters around a factor changes only if any of its neighbors are eliminated, we must prove that if a neighbor of f is eliminated in T_{i-1}, then it must be eliminated in T_{i-1}' and vice versa; additionally we must prove that they also generate the same clusters. Since $f \in P_{i-1}$, the neighbors of f in T_i' can be unaffected, affected with degree-status zero or newly affected in round i. When an unaffected factor g is eliminated in T_{i-1}, it is eliminated in T_i' as well, so the resulting clusters are identical since $\mathcal{C}_{i-1}(g) = \mathcal{C}_{i-1}'(g)$. So any change to $\mathcal{C}_i(f)$ due to f's unaffected neighbors is replicated in $\mathcal{C}_i'(f)$. On the other hand, by definition we cannot eliminate a factor with degree-status zero, so they do not pose a problem even if they are affected. The last case is a newly affected neighbor g of f in T_{i-1} with $\sigma_{i-1}(g) = 1$. But this case is impossible because, if g is eliminated then we would have marked f as affected in T_i via the first rule, or if g is not eliminated then by the second rule and the fact that $\sigma_i(g) = 1$, we would have marked f as affected in T_i. Therefore $\mathcal{C}_i(f) = \mathcal{C}_i'(f)$ for all unaffected factors. This implies that clusters of unaffected factors are identical and do not have to be recalculated in T_i'.

Let $\mathcal{N}_i(f)$ and $\mathcal{N}_i'(f)$ be the set of neighbors of f in T_i and T_i', respectively. Proving the second part of the second claim (i.e., $\mathcal{N}_i(f) = \mathcal{N}_i'(f)$) proceeds similarly to that for $\mathcal{C}_i(f) = \mathcal{C}_i(f)$. The only difference is the initial round when $i = 1$. In round 1, the update algorithm marks all the factors that are incident to the added or removed edges as affected, so for all unaffected factors their neighbor set must be identical in T_i and T_i'. ■

Using this lemma, we can now prove the correctness of our method to incrementally update a hierarchical clustering.

Theorem 7 *Given a valid hierarchical clustering T, let $T' = (T_1', T_2', \ldots, T_l')$ be the updated hierarchical clustering, where $T_i' = (F_i', E_i')$. Then, T' is a valid hierarchical clustering, that is,*

- *the set $M_i = F_i' \setminus F_{i+1}'$ is a maximal independent set containing vertices of degree at most two, and*
- *T_{i+1}' is obtained from T_i' by applying deferred factor elimination to the factors in M_i.*

Proof Recall that A_i is the set of affected factors marked and $M_i' \in A_i$ be the independent set chosen by the algorithm. Let $P_i = F_i' \setminus A_i$ be the set of unaffected factors and $R_i = P_i \setminus F_{i+1}'$ be the ones that are eliminated at round i. The fact that M_i is a maximal independent set follows from Lemma 6

because $M_i = R_i \cup M_i'$. Since the update algorithm keeps the decisions made for the unaffected factors, the set of eliminated vertices are precisely $M_i = R_i \cup M_i'$ and by Lemma 6, M_i is a maximal independent set over degree-one and degree-two in T_i'. The update algorithm applies the deferred factor elimination subroutine on the set M_i', so what remains to be shown is the saved values for R_i are the same as if we eliminate them explicitly. By Lemma 6, the factors in R_i have the same set of clusters around them in T_i and T_i', which means that deferred factor elimination procedure will produce the same result in both elimination trees when unaffected factors are eliminated. Therefore, we can reuse the clusters in R_i. ■

Theorem 7 shows that our update method correctly modifies the cluster tree, and thus marginals can be correctly computed. Note that, by Lemma 3, we also have that the resulting cluster tree also has logarithmic depth. It remains to show that we can efficiently update the clustering itself. We do this by first establishing a bound on the number of affected nodes in each round.

Lemma 8 *For $i = 1, 2, \ldots, l$, let A_i be the set of affected nodes computed by our algorithm after inserting or deleting edge (f_i, f_j) in the elimination tree. Then, $|A_i| \leq 12$.*

Proof First, we observe that the edge (f_i, f_j) defines two connected components, that are either created or merged, in the elimination tree. Since an unaffected node becomes affected only if it is adjacent to an affected factor, the set of affected nodes forms a connected sub-tree throughout the elimination procedure. For the remained of the proof, we focus on the component associated with f_i, and show that it has at most six affected nodes. A similar argument can be applied to the component associated with f_j, thereby proving the lemma.

For round i, let B_i be the set of affected neighbors of with at least one unaffected neighbor and let N_i be the set of newly affected factors. We claim that $|B_i| \leq 2$ and $|N_i| \leq 2$ at every round i. This can be proven inductively: assume that $|B_i|$ and $|N_i|$ are at most two in round $i \geq 0$. Rule 1 for marking a factor affected can make only one newly affected factor at round $i+1$, in which case it is eliminated, and hence $|B_i|$ cannot increase. Rule 2 for marking a factor affected can make two newly affected factors, as shown in the example Figure 13. What is left to be shown is that if $|B_i| = 2$, then rule 2 cannot create two newly affected factors and make $|B_i| > 2$. Let $B_i = \{f_a, f_b\}$ and suppose f_a can force two previously unaffected factors affected in the next round. For this to happen, the degree-status of f_a has to be one in round $i+1$. However, this cannot because f_a must have at least three neighbors in both T_{i+1} and T_{i+1}'. This is because it has two unaffected neighbors plus an affected neighbor that is eventually connected to another unaffected factor through f_b. Note that Figure 13 has $|B_i| = 1$, so we can increase $|B_i|$ by one.

We have now established the fact that the number of affected nodes can increase at most by two in each round, and it remains to be shown that the number of affected nodes is at most six in each connected component.

To prove this, we argue that if there are more than six affected nodes in the connected component, our algorithm eliminates at least two factors. Since affected nodes form a sub-tree that interacts with the rest of the tree on at most two factors, what remains to be shown is that in any tree with at least four nodes, the size of a maximal independent set over the nodes with degree at most two is at least two. To see this, observe that every tree has two leaves, and if the size of the tree is at least four, the distance between these two leaves is at least two or the tree is star-shaped. In either case, any maximal independent set must include at least two nodes, proving the claim. ■

Combining the above arguments, we now conclude that a cluster tree can be efficiently updated if the elimination tree is modified.

Theorem 9 *Let $G = (X, F)$ be a factor graph with n nodes and $\mathcal{H}$ be the cluster tree obtained using an elimination tree T. If we insert or delete a single edge from T, it suffices to re-compute $O(\log n)$ clusters in $\mathcal{H}$ to reflect the changes.*

Proof Since the number of affected factors is constant at each round by Lemma 8 and the number of rounds is $O(\log n)$ by Lemma 3, the result follows. ■

We can easily generalize these results to multiple edge insertions and deletions by considering each connected component resulting from a modification separately. As we discussed in Section 4.1, we only need to recalculate $O(\ell \log(n/\ell))$ many clusters where ℓ is the number of modifications to the elimination tree. We can now state the running time efficiency of our update algorithm under multiple changes to the elimination tree.

Theorem 10 *Let $G = (X, F)$ be a factor graph with n nodes and $\mathcal{H}$ be the cluster tree obtained using an elimination tree T. If we make ℓ edge insertions or deletions in T, we can recompute the new cluster tree in $O(d^{3w} \ell \log(n/\ell))$ time.*

5. Maintaining MAP Configurations

The previous sections provide for efficient marginal queries to user-specified variables and can be extended to compute max-marginals when each sum is replaced with max in the formulas. While we can query each max-marginal, since we do not know *a priori* which entries of the MAP configuration have changed, in the worst case it may take linear time to update the entire MAP configuration. In this section, we show how to use the cluster tree data structure along with a tree traversal approach to efficiently update the entries of the MAP configuration. More precisely, for a change to the model that induces m changes to a MAP configuration, our algorithm computes the new MAP configuration in time proportional to $m \log(n/m)$, without requiring *a priori* knowledge of m or which entries in a MAP configuration will change.

5.1 Computing MAP Configurations Using a Cluster Tree

In Section 3, we described how to compute a cluster tree for computing marginals for any given variable. In this section, we show how this cluster tree can be modified to compute a MAP configuration. First, we modify Equation (2) for computing a cluster λ_j to be

$$\lambda_j = \max_{\mathcal{V}_j} f_j \prod_{\lambda_k \in \mathcal{C}_T(f_j)} \lambda_k \tag{5}$$

where $\mathcal{V}_j \subseteq X$ is the set of children variables of λ_j and $\mathcal{C}_T(f_j)$ is the set of children clusters of λ_j in the cluster-tree. For MAP computations, rather than using boundaries we make use of the argument set of clusters. The argument set X_{λ_j} of a cluster λ_j is the set of variables λ_j depends on at time it was created and it is implicitly computed as we perform hierarchical clustering.

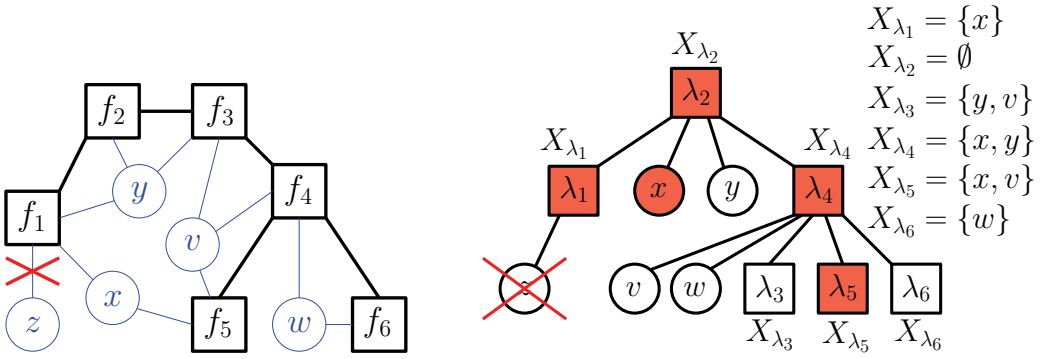


Figure 14: *Updating a MAP configuration*. Factor f_1 is modified and no longer depends on z on the factor graph (left). We first update the clusters on the path from modified clusters to the root, namely, λ_1 and λ_2. Then, we check for changes to the MAP configuration using a top-down traversal in the cluster-tree (right). Here x is assumed to have a different MAP configuration than before, which requires us to check for changes to the MAP configuration in clusters with x in their argument sets, namely λ_4, λ_5. The argument set for each cluster is annotated in the cluster tree.

We now perform a downward pass, in which we select an optimal configuration for the variables associated with the root of the cluster tree, then at its children, and so on. During this downward pass, as we reach each cluster λ_j, we choose the optimal configurations for its children variables $\mathcal{V}_j$ using

$$\mathcal{V}_j^* = \arg\max_X \delta(X_{\lambda_j} = X_{\lambda_j}^*) \, f_j \prod_{\lambda_k \in C_T(f_j)} \lambda_k \tag{6}$$

where $\delta(\cdot)$ is the Kronecker delta, ensuring that λ_j's argument set X_{λ_j} takes on value $X_{\lambda_j}^*$. By the recursive nature of the computation, we are guaranteed that the optimal configuration $X_{\lambda_j}^*$ is selected before reaching the cluster λ_j. This can be proven inductively: assume that X_{λ_j} has an optimal assignment when the recursion reaches the cluster λ_j. We are conditioning on X_{λ_j}, which is the Markov blanket for λ_j, and can therefore optimize the subtree of λ_j independently. The value in Equation (6) is thus the optimal configuration for $\mathcal{V}_j$ (which by definition includes the Markov blanket) for each child cluster λ_k; see Figure 14 for an example.

Theorem 11 *Let G be a factor graph with n nodes and T be an elimination tree on G with tree-width w. The MAP configuration can be computed in $O(nd^{3w})$ time.*

Proof Computation of the formulas in Equations (5) and (6) takes $O(d^{3w})$ by Lemma 3. Since the algorithm visits each node twice, once bottom-up using Equation (5) and once top-down using Equation (6) the total cost is $O(nd^{3w})$. ■

5.2 Updating MAP Configurations Under Changes

In this section we show, somewhat surprisingly, that the time required to update a MAP configuration after a change to the model is proportional to the number of changed entries in the MAP

configuration, rather than the size of the model. Furthermore, the cost of updating the MAP configuration is in the worst case linear in the number of nodes in the factor graph, ensuring that changes to model result in no worse cost than computing the MAP from scratch. This means that, although the extent of any changed configurations is not known *a priori*, it is identified automatically during the update process. For the sake of simplicity, we present the case where we modify a single factor. However, with little alteration the algorithm also applies to an arbitrary number of modifications both to the factors and to the structure of the model.

Let $G = (X, F)$ be a factor graph and $\mathcal{H}$ be its cluster tree. Suppose that we modify a factor $f_1 \in F$ and let λ_1 be the cluster formed after eliminating f_1. Let $\lambda_1, \lambda_2, \ldots, \lambda_k$ be the path from λ_1 to the root λ_k in $\mathcal{H}$. As in Section 4, we recompute each cluster along the path using Equation (5). We additionally mark these clusters *dirty* to indicate that they have been modified. In the top-down phase we search for changes to and update the optimal configuration for the children variables of each cluster. Beginning at the root, we move downward along the path, checking for a MAP change. At each node, we recompute the optimal MAP configuration for the children variables and recurse on any children cluster who is marked as dirty or whose argument set has a variable with a changed MAP configuration.

Figure 14 shows an example of how a MAP configuration changes after a factor (e.g., f_1) is changed in the factor graph. The bottom-up phase marks λ_1 and λ_2 dirty and updates them. The top-down phase starts at the root and re-computes the optimal configuration for x and y. Assuming that the configuration for x is changed, the recursion proceeds on λ_1 due to the dirty cluster and λ_4 due to the modified argument set. At λ_4 we re-compute the optimal MAP configurations for v and w and assuming nothing has changed, we proceed to λ_5 and terminate.

We now prove the correctness and overall running time of this procedure.

Theorem 12 *Suppose that we make a single change to a factor in the input factor graph G, and that a MAP configuration of the new model differs from our previous result on at most m variables. Let* $\gamma = min(1 + rm, n)$*, where r is the maximum degree of any node in G. After updating the cluster tree, the MAP update algorithm can find m variables and their new MAP configurations in* $O(\gamma(1 + \log(\frac{n}{\gamma}))d^w)$ *time.*

Proof Suppose that after the modified factor is changed, we update the cluster tree as described in Section 4. To find the new MAP, we revisit our decision for the configuration of any variables along this path.

Consider how we can rule out any changes in the MAP configuration of a subtree rooted at λ_j in the cluster tree. First, suppose that we have found all changed configurations above λ_j. The decision at λ_j is based on its children clusters and the configuration of its argument set: if none of these variables have changed, and no clusters used in calculating λ_j have changed, then the configuration for all nodes in the subtree rooted λ_j remains valid. Thus, our dynamic MAP update procedure correctly finds all the changed m variables and their new MAP configurations.

Now suppose that m variables have changed the value they take on in the new MAP configuration. The total number of paths with changed argument set is then at most rm. These paths are of height $O(\log n)$, and every node is checked at most once, ensuring that the total number nodes visited is at most $O(\gamma \log(\frac{n}{\gamma}))$ where $\gamma = \min(1 + rm, n)$. Each visit to a cluster λ_j decodes the optimal configuration for its children variables $\mathcal{V}_j$ using Equation (6). Since we are conditioning on the argument set, this computation takes $O(d^{|\mathcal{V}_j|})$ time. Using arguments as in the proof of Lemma 1,

we can show that $\left|\mathcal{V}_j\right| \leq w$. Therefore the top-down phase takes $O(\gamma(1+\frac{n}{\gamma})d^w)$ time. ■

It is also possible, using essentially the same procedure, to process batch updates to the input model. Suppose we modify G or its elimination tree T by inserting and deleting a total of ℓ edges and nodes. First, we use the method described in Section 4 to update the clusters. Then, the total number of nodes recomputed (and hence marked dirty) is guaranteed to be $O(\ell \log(n/\ell))$. Note that we also require $O(\ell \log n)$ time to identify new points of elimination for at most ℓ variables. Therefore, the bottom-up phase will take $O(d^{3w}\ell \log(n/\ell))$ time. The top-down phase works exactly as before and can check an additional $O(rm)$ paths for MAP changes where m is the number of variables with changed MAP value and r is the maximum degree in G. Therefore the top-down phase takes $O(\gamma \log(\frac{n}{\gamma})d^w)$ time where $\gamma = \min(\ell + rm, n)$.

6. Experiments

In this section, we evaluate the performance of our approach by comparing the running times for building, querying, and updating the cluster-tree data structure against (from-scratch or complete) inference using the standard sum- or max-product algorithms. For the experiments, we implemented our proposed approach as well as the sum- and max-product algorithms in Python.[1] In our implementation, all algorithms take the elimination tree as input; when it is not possible to compute the optimal elimination tree for a given input, we use a simple greedy method to construct it (the algorithm grows the tree incrementally while minimizing width). To evaluate our algorithm, we performed experiments with both synthetic data (Section 6.1) and real-world applications (Sections 6.2 and 6.3).

First, we evaluate the practical effectiveness of our proposed approach by considering synthetically generated graphs to compute marginals (Section 6.1.3) and MAP configurations (Section 6.1.4). These experiments show that adaptive inference can yield significant speedups for reasonably chosen inputs. To further explore the limits of our approach, we also perform a more detailed analysis in which we compute the speedup achievable by our method for a range tree-width, dimension, and size parameters. This analysis allows us to better interpret how the asymptotic bounds derived in the previous sections fare in practice.

Second, we evaluate the effectiveness of our approach for two applications in computational biology. The first application studies adaptivity in the context of using an HMM for the standard task of protein secondary structure prediction. For this task, we show how a MAP configuration that corresponds to the maximum likelihood secondary structure can be maintained as mutations are applied to the primary sequence. The second application evaluates our approach on higher-order graphical models that are derived from three-dimensional protein structure. We show our algorithm can efficiently maintain the minimum-energy conformation of a protein as its structure undergoes changes to local sidechain conformations.

6.1 Experiments with Synthetic Data

For our experiments with synthetically generated data, we randomly generate problems consisting of either tree-structured graphs or loopy graphs and measure the running-time for the operations supported by the cluster tree data structure and compare their running times to that of the sum-

1. The source code of our implementation can be obtained by contacting the authors.

product algorithm. Since we perform exact inference, the sum-product algorithm offers an adequate basis for comparison.

6.1.1 DATA GENERATION

For our experiments on synthetically generated data, we randomly generate input instances consisting of either tree-structured graphs or loopy graphs, consisting of n variables, each of which takes on d possible values. For tree-structured graphs, we define how a factor f_i ($1 \leq i < n$) depends on any particular variable x_j ($1 \leq j < n$) through the following distribution:

$$\Pr\{f_i \text{ depends on } x_j\} = \begin{cases} 1 & \text{if} \quad j = i+1, \\ p(1-p)^{i-j} & \text{if} \quad j = 2,\ldots,i, \\ 1-\sum_{s=2}^{i} p(1-p)^{i-s} & \text{if} \quad j = 1. \end{cases}$$

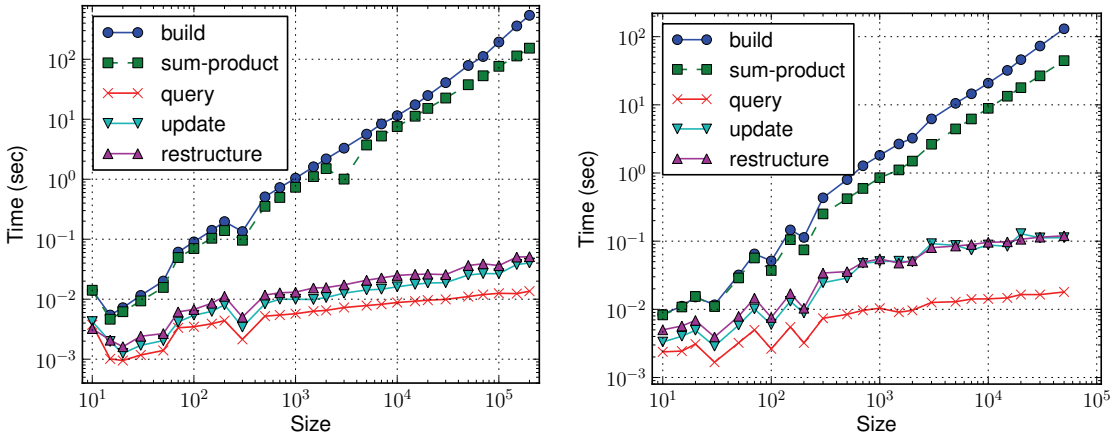
Here, p is a parameter that when set to 1 results in a linear chain. More generally, the parameter p determines how far back a node is connected while growing the random tree. The i^{th} node is expected to connect as far back as the j^{th} node where $j = i - 1/p$, due to the truncated geometric distribution. In our experiments we chose $p = .2$ and $d = 25$ when generating trees.

For loopy graphs, we start with a simple Markov chain, where each factor f_i depends on variables x_i and x_{i+1}, where $1 \leq i < n$. Then for parameters w and p, we add a cycle to this graph as follows: if i is even and less than $n-2(w-1)$, with probability p we create a cycle by adding a new factor g_i that depends on x_i and $x_{i+2(w-1)}$. This procedure is guaranteed to produce a random loopy graph whose width along the chain $x_1,\ldots,x_n$ is at most w; to ensure that the induced width is exactly w we then discard any created loopy graph with width strictly less than w. In our experiments, we set $p = (0.2)^{1/(w-1)}$ so that the maximum width is attained by 20% of the nodes in the chain regardless of the width parameter w. We use an elimination tree $T = (F,E)$ that eliminates the variables $x_1,\ldots,x_n$ in order. More specifically, E includes $\{(f_i, f_{i+1}) : i = 1,\ldots,n-1\}$ and any (f_i, g_i) with $2 \leq i \leq n-2(w-1)$ that is selected by the random procedure above. In our experiments, we varied n between 10 and 50000.

For both tree-structured and loopy factor graphs, we generate the entries of the factors (i.e., the potentials) by sampling a log-normal distribution, that is, each entry is randomly chosen from e^Z where Z is a Gaussian distribution with zero mean and unit variance.

6.1.2 MEASUREMENTS

To compare our approach to sum- and max-product algorithms when the underlying models undergo changes, we measure the running times for build, update, structural update, and query operations. To perform inference with a graphical model that undergoes changes, we start by performing an initial *build* operation that constructs the cluster-tree data structure on the initial model. As the model changes, we reflect these changes to the cluster tree by issuing *update* operations that change the factors, or *structural-update* operations that change the dependencies in the graph (by inserting/deleting edges) accordingly, and retrieve the updated inference results by issuing *query* operations. We are interested in applications where after an initial build, graphical models undergo many small changes over time. Our goal therefore is to reduce the update and query times, at the cost of a slightly slower initial build operation.



(a) tree-structured factor graphs

(b) loopy factor graphs with tree-width 3

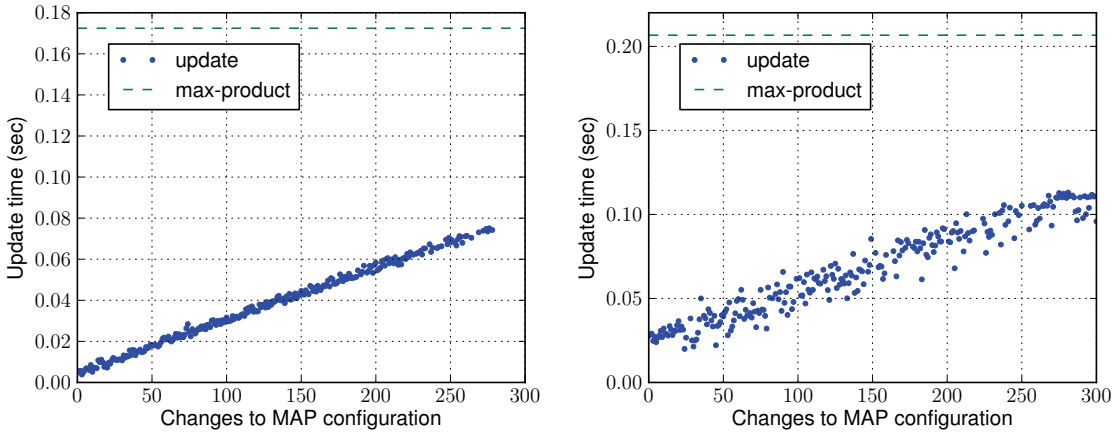
Figure 15: *Marginalization queries and model updates*. We measure the running times for naive sum-product, building the cluster tree, computing marginal queries, updating factors, and restructuring (adding and deleting edges to the elimination tree) for tree-structured and loopy factor graphs. Building the cluster tree is slightly more expensive than a single execution of sum-product, but subsequent updates and queries are much more efficient than recomputing from scratch. For both tree-structured and loopy graphs, our approach is about three orders of magnitude faster than sum-product.

6.1.3 MARGINAL COMPUTATIONS

We consider marginal computation and how we can compute marginals of graphical models that undergo changes using the proposed approach. To this end we measure the running-time for the build, update, structural-update and query operations and compare them to the sum-product algorithm. We consider graphs with tree-width one (trees) and three, with between 10 and 200,000 nodes. For trees, we set $d = 25$, and for graphs we set $d = 6$.

For the build time, we measure the time to build the cluster tree data structure for graphs generated for various input sizes. The running-time of sum-product is defined as the time to compute messages from leaves to a chosen root node in the factor graph. To compute the average time for a query operation, we take the average time over 100 trials to perform a query for a randomly chosen marginal. To compute the update time, we take the average over 100 trials of the time required to change a modify a randomly chosen factor (to a new factor that is randomly generated). To compute the average time required for a structural updates (i.e, restructure operations), we take the average over 100 trials of the total time required to remove a randomly chosen edge, update the cluster tree, and to add the same edge back to the cluster tree.

Figure 15 shows the result of our measurements for tree-structured factor graphs and loopy graphs with tree-width 3. We observe that the running time for the build operations, which constructs the initial cluster tree, is comparable to the time required to perform sum-product. Since we perform exact inference, sum-product is the best we can expect in general. We observe that all of our query and update operations exhibit running times that are logarithmic in n, and are between one



(a) tree-structured (b) loopy

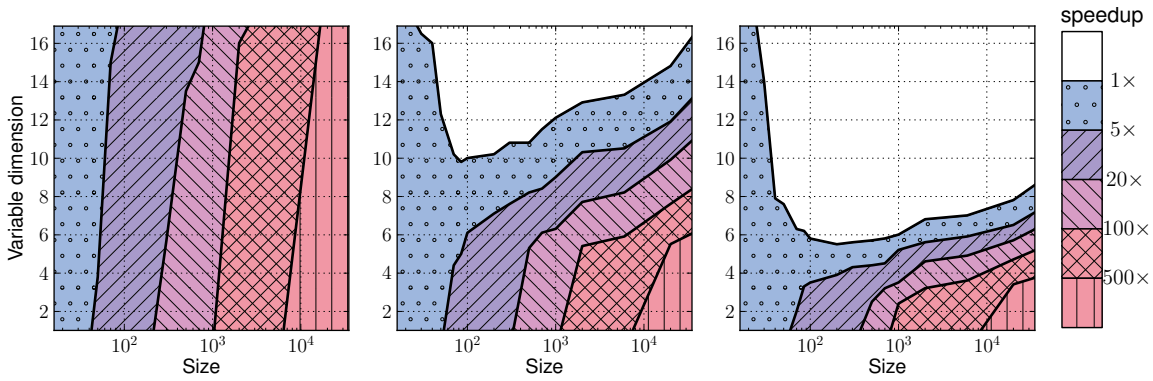
Figure 16: *Updates to MAP configurations*. We report the time required to update a MAP configuration after a single change is made to the input model, in both tree-structured and loopy factor graphs, with 300 variables. Our algorithm takes time that is roughly linear in the number of changed entries, unlike the standard max-product algorithm, which takes time that is linear in the size of the model.

to four orders of magnitude faster than a from-scratch inference with the sum-product algorithm. Update and restructuring operations are costlier than the query operation, as predicted by our complexity bounds on updates ($O(d^{3w} \log n)$, Theorem 5) and queries ($O(d^{2w} \log n)$, Theorem 4). The overall trend is logarithmic in n, and even for small graphs (100–1000 nodes) we observe a factor of 10–30 speedup. In the scenario of interest, where we perform an initial build operation followed by a large number of updates and queries, these results suggest that we can achieve significant speedups in practice.

6.1.4 MAP CONFIGURATIONS

We also tested the approach for computing and maintaining MAP configurations, as outlined in Section 5. For these experiments we generated factor graphs with tree-width one (trees) and three comprised of $n = 300$ variables. For trees, we choose $d = 25$ and for graphs we choose $d = 6$. We compute the update time by uniformly randomly selecting a factor and replacing with another factor, averaging over 100 updates. We compare the update time to the running-time of the max-product algorithm, which computes messages from leaves to a chosen root node in the factor graph and then performs maximization back to the leaves.

Figure 16 show the results of our experiments. For both tree-structured and loopy factor graphs, we observed strong linear dependence between the time required to update the MAP on the number of changed entries in the MAP configuration. We note that while there is an additional logarithmic factor in the running time, it is likely negligible since n was set to be small enough to observe changes to the entire MAP configuration. Overall, our method of updating MAP configurations were substantially faster than computing a MAP configuration from scratch in all cases, for both tree-structured and loopy graphs.



(a) tree width 2 (b) tree width 3 (c) tree width 4

Figure 17: *Speedup Analysis*. The regions where we obtain speedup, defined as the ratio of running time of our algorithm for a single update and query to the running time of standard sum-product, are shown for loopy graphs with width 2, 3 and 4 and variable dimensions 2–16.

6.1.5 EFFICIENCY TRADE-OFFS AND CONSTANT FACTORS

Our experiments with the computations of marginals and MAP configurations (Sections 6.1.3 and 6.1.4) suggest that our proposed approach can lead to efficiency improvements and significant speedups in practice. In this section, we present a more detailed analysis by considering a broader range of graphs and by presenting a more detailed analysis by considering constant factors and realized exponents.

For a graph of n nodes with tree-width w and dimension d, inference of marginals using sum product algorithm requires $O(d^{w+1}n)$ time. With adaptive inference, the preprocessing step takes $O(d^{3w}n)$ time whereas updates and queries after unit changes require $O(d^{3w}\log n)$ and $O(d^{2w}\log n)$ time respectively. These asymptotic bounds imply that using updates and queries, as opposed to performing inference with sum-product, would yield a speedup of $O(\frac{n}{d^{3w}\log n})$, where d is the dimension (domain size) and w and n is the tree-width and the size of the graphical model. In the case that d and w can be bounded by constants, this speedup would result in a near linear efficiency increase as the size of the graphical model increases. At what point and with what inputs exactly the speedups materialize, however, depends on the constant factors hidden by our asymptotic analysis. For example in Figure 15, we obtain speedups for nearly all graphs considered.

Speedups for varying input parameters.

To assess further the practical effectiveness of adaptive inference, we have measured the performance of our algorithm versus sum-product for graphical models generated at random with varying values of d, w and n. Specifically, for a given d, w, n we generate a random graphical model as previously described and measure the average time for ten randomly generated updates plus queries, and compare this to the time to perform from-scratch inference using the sum-product algorithm. The resulting speedup is defined as the ratio of the time for the from-scratch inference to the time for the random update plus query.

Figure 17 illustrates a visualization of this speedup information. For tree-widths, $2,3,4$, we show the speedup expected for each pair of values (n,d). Given fixed w,d we expect the speedup to increase as n increases. The empirical evaluation illustrates this trend; for example, at $w = 3$ and $d = 4$, we see a five-fold or more speedup starting with input graphs with $n \approx 100$. As the plots illustrate, we observe that when the tree-width is 2 or less, as in Figure 17a, adaptive inference is preferable in many cases even for small graphs. With tree-widths 3 and 4, we obtain speedups for dimensions below 10 and 6 respectively. We further observe that for a given width w, we obtain higher speedups as we reduce the dimensionality d and as we increase n, except for small values of n. Disregarding such small graphs, this is consistent with our theoretical bounds. In small graphs ($n < 100$) we see higher speedups than predicted because our method's worst-case exponential dependence is often not achieved, a phenomenon we examine in more detail shortly.

Constant Factors. The experiments shown in Figures 17 and 15 show that adaptive inference can deliver speedups even for modest input sizes. To understand these result better, it helps to consider the constant factors hidden in our asymptotic bounds. Taking into account the constant factors, we can write the dynamic update times with adaptive inference as $\alpha_a d^{3w} \log n + \beta_a \log n$, where α_a, β_a are constants dependent on the cost of operations involved. The first term $\alpha_a d^{2w} \log n$ accounts for the cost of matrix computations (when computing the cluster functions) at each node and the term $\beta_a \log n$ accounts for the time to locate and visit the $\log n$ nodes to be updated in the cluster-tree data structure. In comparison, sum-product algorithm requires $\alpha_s d^{w+1} n + \beta_s n$ time for some constants α_s, β_s which again represent matrix computation at each node and the finding and visiting of the nodes. Thus the speedup would be $\frac{\alpha_s d^{w+1} n + \beta_s n}{\alpha_a d^{3w} \log n + \beta_a \log n}$.

These bounds suggest that for fixed d, w, there will be some n_0 beyond which speedups will be possible. The value of n_0 depends on the relationships between the constants. First, constants α_a and α_s are similar because they both involve similar matrix operations. Also, the constants β_a and β_s are similar because they both involve traversing a tree in memory by following pointers. Given this relationship between the constants, if the non-exponential terms dominate, that is, $\beta \gg \alpha$, then we can obtain speedups even for small n.

Our experiments showing that speedups are realized at relatively modest input sizes suggest that the βs dominate the αs. To test this hypothesis, we measured separately the time required for the matrix operations. For an example model with $n = 10000, w = 3, d = 6$, the matrix operations (the first term in the formulas) consumed roughly half the total time: 8.3 seconds, compared to 7.4 seconds for the rest of the algorithm. This suggests that βs are indeed larger than the αs. This should be expected: the constant factor for matrix computation, performed locally and in machine registers, should be far smaller than the parts of the code that include more random memory accesses (e.g., for finding nodes) and likely incur cache misses as well, which on modern machines can be hundreds of times slower than register computations.

While this analysis compares the dynamic update times of adaptive inference, comparing the pre-processing (build) time of our cluster tree data structures (Figure 15) suggests that a similar case holds. Specifically, in Theorem 2 we showed that the building the cluster tree takes in the worst case $\Theta(d^{3w} \cdot n)$ whereas the standard sum-product takes $\Theta(d^{w+1} \cdot n)$. Thus the worst-case build time could be $d^{2w} = 6^{2 \cdot 3} = 46656$ times slower than standard sum-product. In our experiments, this ratio is significantly lower. For a graph of size 50,000, for example, it is only 3.05. Figure 15(b) also shows a modest increase in build time as the input size grows. For example at $n = 100$, our build time is about 1.20 slower than performing sum-product. Another 100-fold increase in the size makes

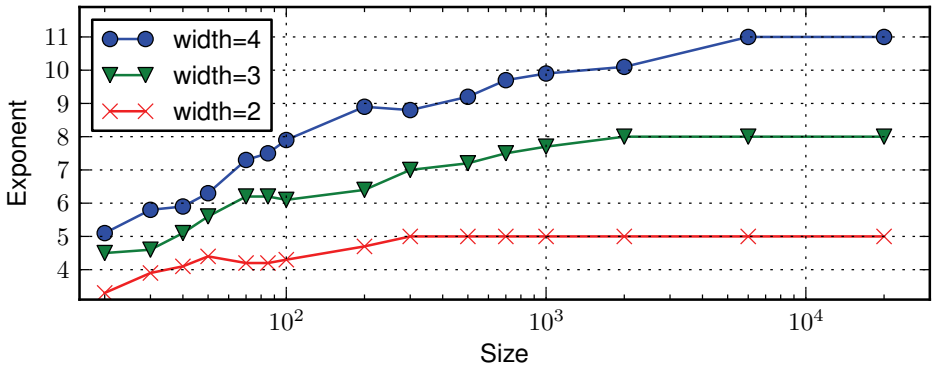


Figure 18: *Cost of cluster computation*. The maximum exponent e during the computation of clusters, which takes $O(d^e)$ time, is plotted as a function of the input size. As can be seen, the exponent starts relatively small and increases to reach the theoretical maximum of three times the tree-width as the graph size increases. Since the cost of computing clusters in our algorithm is $O(d^e)$, our approach can yield speedup even for small and medium-sized models. This shows that our worst-case bound of $O(d^{3w})$ for computing clusters can be pessimistic, that is, it is not tight except in larger graphs.

our build time about 2.05 slower. As we illustrate in next this section, this is due to our bounds not being tight in small graphs.

It is also worth noting that the differences between the running times of query and update operations are also low in practice, in contrast to the results of Theorems 10 and 4. According to Theorems 10 and 4, the query operation could, in the worst-case, be $d^w = 6^3 = 216$ times faster than an update operation. However, in practice we see that, for example at $n = 100$, the queries are about 2.5 times faster than updates. This gap does increase as n increases, for example, at $n = 50000$, queries are about 6.7 times faster than updates; this is again due to our bounds not being tight in small graphs (described in detail next).

Tightness of our bounds in small graphs. Our experiments with varying sizes of graphs show some unexpected behavior. For example, contrary to our bound that predicts speedup to increase as the input size increases, we see in Figure 17 that speedups occur for very small graphs (less than 100 nodes) then disappear as the graph size increases. To understand the reasons for this we calculated the actual exponential factor in our bounds occurring in our randomly generated graphs, by building each cluster-tree and calculating the maximum exponent encountered during the computation. Figure 18 shows the measurements, which demonstrate that for small graphs the worst case asymptotic bound is not realized because the exponent remains small. In other words, we perform far fewer computations than would be predicted by our worst-case bound. As the graph size grows, the worst case configurations become increasingly likely to occur, and the exponent eventually reaches the bound predicted by our analysis. This suggests that our bounds may be loose for small graphs, but more accurate for larger graphs, and explains why speedups are possible even for small graphs.

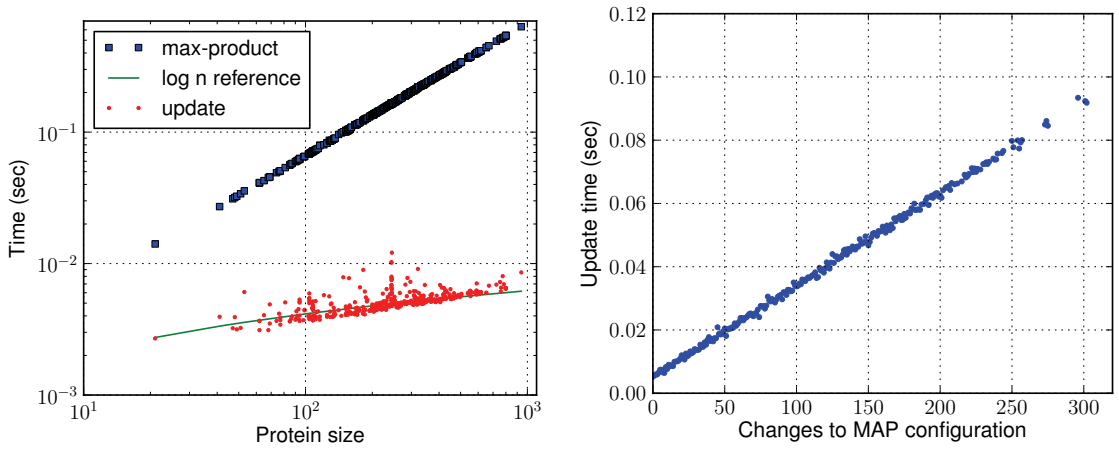
6.2 Sequence Analysis with Hidden Markov Models

HMMs are a widely-used tool to analyze DNA and amino acid sequences; typically an HMM is trained using a sequence with known function or annotations, and new sequences are analyzed by inferring hidden states in the resulting HMM. In this context, our algorithm for updating MAP configuration can be used to study the effect of changes to the model and observations on hidden states of the HMM. We consider the application of secondary structure prediction from the primary amino acid sequence of a given protein. This problem has been studied extensively (Frishman and Argos, 1995), and is an ideal setting to demonstrate the benefits of our adaptive inference algorithm. An HMM for protein secondary structure prediction is constructed by taking the observed variables to be the primary sequence and setting the hidden variables (i.e., one hidden state per amino acid) to be the type of secondary structure element (α-helix, β-strand, or random coil) of the corresponding amino acid. Then, a MAP configuration of the hidden states in this model identifies the regions with α helix and β strands in the given sequence. This general approach has been studied and refined (Chu et al., 2004; Martin et al., 2005), and is capable of accurately predicting secondary structure. In the context of secondary structure prediction, our algorithm to adaptively update the model could be used in protein design applications, where we make "mutations" to a starting sequence so that the resulting secondary structure elements match a desired topology. Or, more conventionally, our algorithm could be applied to determine which residues in the primary sequence of a given protein are critical to preserving the native pattern of secondary structure elements. It is also worth pointing out that our approach is fully general and can be used at any application where biological sequences are represented by HMMs (e.g., DNA or RNA sequence, exon-intron chains, CpG islands) and we want to study the effects of changes to these sequences.

For our experiments, we constructed an HMM for secondary structure prediction by constructing an observed state for each amino acid in the primary sequence, and a corresponding hidden state indicating its secondary structure type. We estimated the model parameters using 400 protein sequences labeled by the DSSP algorithm (Kabsch and Sander, 1983), which annotates a three-dimensional protein structure with secondary structure types using standard geometric criteria. Since repeated modification to a protein sequence typically causes small updates to the regions with α helices and β strands, we expect to gain significant speedup by using our algorithm. To test this hypothesis, we compared the time to update MAP configuration in our algorithm against the standard max-product algorithm. The results of this experiment are given in Figure 19(a). We observed that overall the time to update secondary structure predictions were 10-100 times faster than max-product. The overall trend of running times, when sorted by protein size, is roughly logarithmic. In some cases, smaller proteins required longer update times; in these cases it is likely that due to the native secondary structure topology, a single mutation induced a large number of changes in the MAP configuration. We also studied the update times for a single protein, *E. coli hemolysin* (PDB id: 1QOY), with 302 amino acids, as we apply random mutations (see Figure 19(b)). As in Section 6.1.4 above, we see that the update time scales linearly with the number of changes to a MAP configuration, rather than depending on the size of the primary sequence.

6.3 Protein Sidechain Packing with Factor Graphs

In the previous section, we considered an application where the input model was a chain-structured representation of the protein primary sequence. In this section, we consider a higher-order representation that defines a factor graph to model the three-dimensional structure of protein, which



(a) Update runtimes for 400 proteins

(b) Update times for protein 1QOY

Figure 19: *Secondary structure prediction using HMMs*. We applied our algorithm to perform updates in HMMs for secondary structure prediction. For our data set, we can perform MAP updates about 10-100 faster than max-product, and we see a roughly logarithmic trend as the size of the protein increases. For a single protein, *E. coli hemolysin*, we see that the time required to update the MAP configuration is linear in the number of changes to the MAP configuration, rather than in the size of the HMM.

essentially defines its biochemical function. Graphical models constructed from protein structures have been used to successfully predict structural properties (Yanover and Weiss, 2002) as well as free energy (Kamisetty et al., 2007). These models are typically constructed by taking each node as an amino acid whose states represent a discrete set of local conformations called *rotamers* (Dunbrack Jr., 2002), and basing conditional probabilities on a physical energy function (e.g., Weiner et al., 1984 and Canutescu et al., 2003).

The typical goal of using these models is to efficiently compute a maximum-likelihood (i.e., minimum-energy) conformation of the protein in its native environment. Our algorithmic framework for updating MAP configurations allows us to study, for example, the effects of amino acid mutations, and the addition and removal of edges corresponds directly to allowing backbone motion in the protein. Applications that make use of these kinds of perturbations include protein design and ligand-binding analysis. The common theme of these applications is that, given an input protein structure with a known backbone, we wish to characterize the effects of changes to the underlying model (e.g., by modifying amino acid types or their local conformations), in terms of their effect on a MAP configurations (i.e., the minimum energy conformation of the protein).

For our experiments, we studied the efficiency of adaptively updating the optimal sidechain conformation after a perturbation to the model in which a random group of sidechains are fixed to new local conformations. This experiment is meant to mimic a ligand-binding study, in which we would like to test how introducing ligands to parts of the protein structure affect the overall minimum-energy conformation. For our data set, we took about 60 proteins from the SCWRL benchmark or varying sizes (between 26 and 244 amino acids) and overall topology.

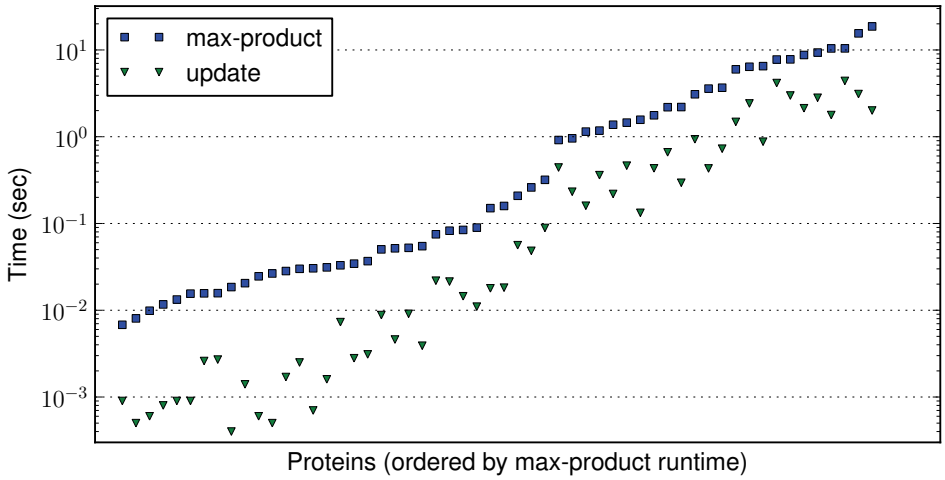


Figure 20: *Adaptive sidechain packing for protein structures*. For 60 proteins from the SCWRL benchmark, we compared the time to adaptively update a MAP configuration against max-product. Since this set of proteins have a diverse set of folds (and thus graph structures), we order the inputs by the time taken by max-product. The speedup achieved by our algorithm varies due to the diversity of protein folds, but on average our approach is 6.88 times faster than computation from scratch.

For each protein, we applied updates to a random group within a selected set amino acids (e.g., to represent an active site) by choosing a random rotameric state for each. With appropriate preprocessing (using Goldstein dead-end elimination), we were able to obtain accurate models with an induced width of about 5 on average. For the cluster tree corresponding to each protein we selected a set of 10 randomly chosen amino acids for modification, and recorded the average time, over 100 such trials, to update a MAP configuration and compared it against computing the latter from scratch. The results of our experiment are given in Figure 20. Due to the diversity of protein folds, and thus the resulting factor graphs, we sort the results according to the time required for max-product. We find that our approach consistently outperforms max-product, and was on average 6.88 times faster than computation from scratch.

We note that the overall trend for our algorithm versus max-product is somewhat different than the results in Sections 6.1.4 and 6.2. In those experiments we observed a clear logarithmic trend in running times for our algorithm versus max-product, since the constant-factor overheads (e.g., for computing cluster functions) grew as a function of a model size. For adaptive sidechain packing, it is difficult to make general statements about the complexity of a particular input model with respect to its size: a small protein may be very tightly packed and induce a very dense input model, while a larger protein may be more loosely structured and induce a less dense model.

7. Conclusion

In this paper, we have presented an adaptive framework for performing exact inference that efficiently handles changes to the input factor graph and its associated elimination tree. Our approach

to adaptive inference requires a linear preprocessing step in which we construct a cluster-tree data structure by performing a generalized factor elimination; the cluster tree offers a balanced representation of an elimination tree annotated with certain statistics. We can then make arbitrary changes to the factor graph or elimination tree, and update the cluster tree in logarithmic time in the size of the input factor graph. Moreover, we can also calculate any particular marginal in time that is logarithmic in the size of the input graph, and update MAP configurations in time that is roughly proportional to the number of entries in the MAP configuration that are changed by the update.

As with all methods for exact inference, our algorithms carry a constant factor that is exponential in the width of the input elimination tree. Compared to traditional methods, this constant factor is larger for adaptive inference; however the running time of critical operations are logarithmic, rather than linear, in the size of the graph in the common case. In our experiments, we establish that for any fixed tree-width and variable dimension, adaptive inference is preferable as long as the input graph is sufficiently large. For reasonable values of these input parameters, our experimental evaluation shows that adaptive inference can offer a substantial speedup over traditional methods. Moreover, we validate our algorithm using two real-world computational biology applications concerned with sequence and structure variation in proteins.

At a high level, our cluster-tree data structure is a replacement for the junction tree in the typical sum-product algorithm. A natural question, then, is whether our data structure, can be extended to perform approximate inference. The approach does appear to be amenable to methods that rely on approximate elimination (e.g., Dechter, 1998), since these approximations can incorporated be into the cluster functions in the cluster tree. Approximate methods that are iterative in nature (e.g., Wainwright et al., 2005a,b and Yedidia et al., 2004), however, may be more difficult, since they often make a large number of changes to messages in each successive iteration.

Another interesting direction is to tune the cluster tree construction based on computational concerns. While deferred factor elimination gives rise to a balanced elimination tree, it also incurs a larger constant factor dependent on the tree width. While our benchmarks show that this overhead can be pessimistic, it is also possible to tune the number of deferred factor eliminations performed, at the expense of increasing the depth of the resulting cluster tree. It would be interesting to incorporate additional information into the deferred elimination procedure used to build the cluster tree to reduce this constant factor. For example, we can avoid creating a cluster function if its run-time complexity is high (e.g., its dimension or the domain sizes of its variables are large), preferring instead a cluster tree that has a greater depth but will yield overall lower costs for queries and updates.

Acknowledgments

This research was supported in part by gifts from Intel and Microsoft Research (U. A.) and by the National Science Foundation through award IIS-1065618 (A. I.) and the CAREER award IIS-0643768 (R. M.).

References

U. Acar, A. T. Ihler, R. R. Mettu, and Ö. Sümer. Adaptive Bayesian inference in general graphs. In *Proceedings of the 24th Annual Conference on Uncertainty in Artificial Intelligence*, pages 1–8, 2008.

U. A. Acar. *Self-Adjusting Computation*. PhD thesis, Department of Computer Science, Carnegie Mellon University, May 2005.

U. A. Acar, G. Blelloch, R. Harper, J. Vittes, and M. Woo. Dynamizing static algorithms with applications to dynamic trees and history independence. In *ACM-SIAM Symposium on Discrete Algorithms (SODA)*, 2004.

U. A. Acar, G. Blelloch, and J. Vittes. An experimental analysis of change propagation in dynamic trees. In *Proc. 7th ACM-SIAM W. on Algorithm Eng. and Exp'ts*, 2005.

U. A. Acar, Guy E. Blelloch, and Robert Harper. Adaptive functional programming. *ACM Trans. Prog. Lang. Sys.*, 28(6):990–1034, 2006.

U. A. Acar, A. T. Ihler, R. R. Mettu, and Ö Sümer. Adaptive Bayesian inference. In *Advances in Neural Information Processing Systems 20*. MIT Press, 2007.

U. A. Acar, Guy E. Blelloch, Matthias Blume, Robert Harper, and Kanat Tangwongsan. An experimental analysis of self-adjusting computation. *ACM Trans. Prog. Lang. Sys.*, 32(1):3:1–3:53, 2009a.

U. A. Acar, A. T. Ihler, R. R. Mettu, and Ö. Sümer. Adaptive updates for maintaining MAP configurations with applications to bioinformatics. In *Proceedings of the IEEE Workshop on Statistical Signal Processing*, pages 413–416, 2009b.

A. A. Canutescu, A. A. Shelenkov, and R. L. Dunbrack Jr. A graph-theory algorithm for rapid protein side-chain prediction. *Protein Sci*, 12(9):2001–2014, Sep 2003.

W. Chu, Z. Ghahramani, and D. Wild. A graphical model for protein secondary structure prediction. In *Proc. 21st International Conference on Machine Learning*, 2004.

A. Darwiche. *Modeling and Reasoning with Bayesian Networks*. Cambridge, 1st edition, 2009.

A. Darwiche and M. Hopkins. Using recursive decomposition to construct elimination orders, jointrees, and dtrees. In *Trends in Artificial Intelligence, Lecture Notes in AI*, pages 180–191. Springer-Verlag, 2001.

R. Dechter. Bucket elimination: A unifying framework for probabilistic inference. In M. I. Jordan, editor, *Learning in Graphical Models*, pages 75–104. MIT Press, 1998.

A. L. Delcher, A. J. Grove, S. Kasif, and J. Pearl. Logarithmic-time updates and queries in probabilistic networks. *J. Artificial Intelligence Research*, 4:37–59, 1995.

R. L. Dunbrack Jr. Rotamer libraries in the 21st century. *Curr Opin Struct Biol*, 12(4):431–440, 2002.

D. Frishman and P. Argos. Knowledge-based protein secondary structure assignment. *Proteins: Structure, Function and Genetics*, 23:566–579, 1995.

M. A. Hammer, U. A. Acar, and Y. Chen. CEAL: a C-based language for self-adjusting computation. In *Proceedings of the 2009 ACM SIGPLAN Conference on Programming Language Design and Implementation*, June 2009.

W. Kabsch and C. Sander. Dictionary of protein secondary structure: pattern recognition of hydrogen-bonded and geometrical features. *Biopolymers*, 22(12):2577–2637, 1983.

H. Kamisetty, E. P. Xing, and C. J. Langmead. Free energy estimates of all-atom protein structures using generalized belief propagation. In *Proc. 11th Ann. Int'l Conf. Research in Computational Molecular Biology*, pages 366–380, 2007.

K. Kask, R. Dechter, J. Larrosa, and A. Dechter. Unifying cluster-tree decompositions for reasoning in graphical models. *Artificial Intelligence*, 166:165–193, 2005.

S. Koenig, M. Likhachev, Y. Liu, and David Furcy. Incremental heuristic search in artificial intelligence. *Artificial Intelligence Magazine*, 25:99–112, 2004.

P. Kohli and P. H. S. Torr. Dynamic graph cuts for efficient inference in markov random fields. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29:2079–2088, 2007.

N. Komodakis, G. Tziritas, and N. Paragios. Performance vs computational efficiency for optimizing single and dynamic mrfs: Setting the state of the art with primal-dual strategies. *Comput. Vis. Image Underst.*, 112:14–29, October 2008.

F. Kschischang, B. Frey, and H.-A. Loeliger. Factor graphs and the sum-product algorithm. *IEEE Trans. Inform. Theory*, 47(2):498–519, February 2001.

S. Lauritzen and D. Spiegelhalter. Local computations with probabilities on graphical structures and their applications to expert systems. *J. Royal Stat. Society, Ser. B*, 50:157–224, 1988.

J. Martin, J.-F. Gibrat, and F. Rodolphe. Choosing the optimal hidden Markov model for secondary-structure prediction. *IEEE Intelligent Systems*, 20(6):19–25, 2005.

G. L. Miller and J. H. Reif. Parallel tree contraction and its application. In *Proc. 26th IEEE Symp. Found. of Comp. Sci.*, pages 487–489, 1985.

V. Namasivayam, A. Pathak, and V. Prasanna. Scalable parallel implementation of bayesian network to junction tree conversion for exact inference. In *Information Retrieval: Data Structures and Algorithms*, pages 167–176. Prentice-Hall PTR, 2006.

J. Pearl. *Probabilistic Reasoning in Intelligent Systems*. Morgan Kaufman, San Mateo, 1988.

D. M. Pennock. Logarithmic time parallel Bayesian inference. In *Proc. 14th Annual Conf. on Uncertainty in Artificial Intelligence*, pages 431–438, 1998.

D. D. Sleator and R. E. Tarjan. A data structure for dynamic trees. *Journal of Computer and System Sciences*, 26(3):362–391, 1983.

M. Wainwright, T. Jaakkola, and A. Willsky. MAP estimation via agreement on (hyper)trees: message-passing and linear programming approaches. *IEEE Trans Info Theory*, 51(11):3697–3717, 2005a.

M. J. Wainwright, T. S. Jaakkola, and A. S. Willsky. A new class of upper bounds on the log partition function. *IEEE Trans Info Theory*, 51(7):2313–2335, July 2005b.

S. J. Weiner, P.A. Kollman, D.A. Case, U.C. Singh, G. Alagona, S. Profeta Jr., and P. Weiner. A new force field for the molecular mechanical simulation of nucleic acids and proteins. *J. Am. Chem. Soc.*, 106:765–784, 1984.

Y. Xia and V. K. Prasanna. Junction tree decomposition for parallel exact inference. In *IEEE International Parallel and Distributed Preocessing Symposium*, pages 1–12, 2008.

C. Yanover and Y. Weiss. Approximate inference and protein folding. In *Proc. NIPS*, pages 84–86, 2002.

J. S. Yedidia, W. T. Freeman, and Y. Weiss. Constructing free energy approximations and generalized belief propagation algorithms. Technical Report 2004-040, MERL, May 2004.

Journal of Machine Learning Research 12 (2011) 3187-3225 Submitted 10/10; Revised 5/11; Published 11/11

Group Lasso Estimation of High-dimensional Covariance Matrices

Jérémie Bigot JEREMIE.BIGOT@MATH.UNIV-TOULOUSE.FR
Institut de Mathématiques de Toulouse
Université Paul Sabatier
118, route de Narbonne
31062 Toulouse, Cedex 9, France

Rolando J. Biscay ROLANDO.BISCAY@UV.CL
DEUV-CIMFAV
Facultad de Ciencias
Gran Bretana 1091, 4to piso
Playa Ancha, Valparaíso, Chile.

Jean-Michel Loubes JEAN-MICHEL.LOUBES@MATH.UNIV-TOULOUSE.FR
Institut de Mathématiques de Toulouse
Université Paul Sabatier
118, route de Narbonne
31062 Toulouse, Cedex 9, France

Lilian Muñiz-Alvarez LILIAN@MATCOM.UH.CU
Facultad de Matemática y Computación
San Lázaro y L. Municipio Plaza de la Revolución
Universidad de La Habana
Habana, Cuba

Editor: Tong Zhang

Abstract

In this paper, we consider the Group Lasso estimator of the covariance matrix of a stochastic process corrupted by an additive noise. We propose to estimate the covariance matrix in a high-dimensional setting under the assumption that the process has a sparse representation in a large dictionary of basis functions. Using a matrix regression model, we propose a new methodology for high-dimensional covariance matrix estimation based on empirical contrast regularization by a group Lasso penalty. Using such a penalty, the method selects a sparse set of basis functions in the dictionary used to approximate the process, leading to an approximation of the covariance matrix into a low dimensional space. Consistency of the estimator is studied in Frobenius and operator norms and an application to sparse PCA is proposed.

Keywords: group Lasso, ℓ^1 penalty, high-dimensional covariance estimation, basis expansion, sparsity, oracle inequality, sparse PCA

1. Introduction

Let $\mathbb{T}$ be some subset of $\mathbb{R}^p$, $p \in \mathbb{N}$, and let $X = (X(t))_{t\in\mathbb{T}}$ be a stochastic process with values in $\mathbb{R}$. Assume that X has zero mean $\mathbb{E}(X(t)) = 0$ for all $t \in \mathbb{T}$, and finite covariance $\sigma(s,t) = \mathbb{E}(X(s)X(t))$ for all $s,t \in \mathbb{T}$. Let $t_1,\ldots,t_n$ be fixed points in $\mathbb{T}$ (deterministic design), $X_1,...,X_N$

independent copies of the process X, and suppose that we observe the noisy processes

$$\widetilde{X}_i(t_j) = X_i(t_j) + \mathcal{E}_i(t_j) \text{ for } i = 1,...,N,\ j = 1,...,n, \tag{1}$$

where $\mathcal{E}_1,...,\mathcal{E}_N$ are independent copies of a second order Gaussian process $\mathcal{E}$ with zero mean and independent of X, which represent an additive source of noise in the measurements. Based on the noisy observations (1), an important problem in statistics is to construct an estimator of the covariance matrix $\mathbf{\Sigma} = \mathbb{E}\left(\boldsymbol{X}\boldsymbol{X}^\top\right)$ of the process X at the design points, where $\boldsymbol{X} = (X(t_1),...,X(t_n))^\top$. This problem is a fundamental issue in many applications, ranging from geostatistics, financial series or epidemiology for instance (see Stein, 1999, Journel, 1977 or Cressie, 1993; Wikle and Cressie, 1999 for general references and applications). Estimating such a covariance matrix has also important applications in dimension reduction by principal component analysis (PCA) or classification by linear or quadratic discriminant analysis (LDA and QDA).

In Bigot et al. (2010), using N independent copies of the process X, we have proposed to construct an estimator of the covariance matrix $\mathbf{\Sigma}$ by expanding the process X into a dictionary of basis functions. The method in Bigot et al. (2010) is based on model selection techniques by empirical contrast minimization in a suitable matrix regression model. This new approach to covariance estimation is well adapted to the case of low-dimensional covariance estimation when the number of replicates N of the process is larger than the number of observations points n. However, many application areas are currently dealing with the problem of estimating a covariance matrix when the number of observations at hand is small when compared to the number of parameters to estimate. Examples include biomedical imaging, proteomic/genomic data, signal processing in neurosciences and many others. This issue corresponds to the problem of covariance estimation for high-dimensional data. This problem is challenging since, in a high-dimensional setting (when $n >> N$ or $n \sim N$), it is well known that the sample covariance matrices

$$\boldsymbol{S} = \frac{1}{N}\sum_{i=1}^{N} \boldsymbol{X}_i\boldsymbol{X}_i^\top \in \mathbb{R}^{n\times n}, \text{ where } \boldsymbol{X}_i = (X_i(t_1),...,X_i(t_n))^\top, i = 1,\ldots,N$$

and

$$\widetilde{\boldsymbol{S}} = \frac{1}{N}\sum_{i=1}^{N} \widetilde{\boldsymbol{X}}_i\widetilde{\boldsymbol{X}}_i^\top \in \mathbb{R}^{n\times n}, \text{ where } \widetilde{\boldsymbol{X}}_i = \left(\widetilde{X}_i(t_1),...,\widetilde{X}_i(t_n)\right)^\top, i = 1,\ldots,N$$

behave poorly, and are not consistent estimators of $\mathbf{\Sigma}$. For example, suppose that the $\boldsymbol{X}_i$'s are independent and identically distributed (i.i.d.) random vectors in $\mathbb{R}^n$ drawn from a multivariate Gaussian distribution. Then, when $\frac{n}{N} \to c > 0$ as $n,N \to +\infty$, neither the eigenvalues nor the eigenvectors of the sample covariance matrix $\boldsymbol{S}$ are consistent estimators of the eigenvalues and eigenvectors of $\mathbf{\Sigma}$ (see Johnstone, 2001). This topic has thus recently received a lot of attention in the statistical literature. To achieve consistency, recently developed methods for high-dimensional covariance estimation impose sparsity restrictions on the matrix $\mathbf{\Sigma}$. Such restrictions imply that the true (but unknown) dimension of the model is much lower than the number $\frac{n(n+1)}{2}$ of parameters of an unconstrained covariance matrix. Under various sparsity assumptions, different regularizing methods of the empirical covariance matrix have been proposed. Estimators based on thresholding or banding the entries of the empirical covariance matrix have been studied in Bickel and Levina (2008b) and Bickel and Levina (2008a). Thresholding the components of the empirical covariance matrix has also been proposed by El Karoui (2008) and the consistency of such estimates is studied using

tools from random matrix theory. Fan et al. (2008) impose sparsity on the covariance via a factor model which is appropriate in financial applications. Levina et al. (2008) and Rothman et al. (2008) propose regularization techniques with a Lasso penalty to estimate the covariance matrix or its inverse. More general penalties have been studied in Lam and Fan (2009). Another approach is to impose sparsity on the eigenvectors of the covariance matrix which leads to sparse PCA. Zou et al. (2006) use a Lasso penalty to achieve sparse representation in PCA, d'Aspremont et al. (2008) study properties of sparse principal components by convex programming, while Johnstone and Lu (2009) propose a PCA regularization by expanding the empirical eigenvectors in a sparse basis and then apply a thresholding step.

In this paper, we propose to estimate $\mathbf{\Sigma}$ in a high-dimensional setting by using the assumption that the process X has a sparse representation in a large dictionary of basis functions. Using a matrix regression model as in Bigot et al. (2010), we propose a new methodology for high-dimensional covariance matrix estimation based on empirical contrast regularization by a group Lasso penalty. Using such a penalty, the method selects a sparse set of basis functions in the dictionary used to approximate the process X. This leads to an approximation of the covariance matrix $\mathbf{\Sigma}$ into a low dimensional space, and thus to a new method of dimension reduction for high-dimensional data. Group Lasso estimators have been studied in the standard linear model and in multiple kernel learning to impose a group-sparsity structure on the parameters to recover (see Nardi and Rinaldo, 2008, Bach, 2008 and references therein). However, to the best of our knowledge, it has not been used for the estimation of covariance matrices using a functional approximation of the process X.

The rest of the paper is organized as follows. In Section 2, we describe a matrix regression model for covariance estimation, and we define our estimator by group Lasso regularization. The consistency of such a procedure is investigated in Section 3 using oracle inequalities and a non-asymptotic point of view by holding fixed the number of replicates N and observation points n. Consistency of the estimator is studied in Frobenius and operator norms. Various results existing in matrix theory show that convergence in operator norm implies convergence of the eigenvectors and eigenvalues (for example through the use of the $\sin(\theta)$ theorems in Davis and Kahan, 1970). Consistency in operator norm is thus well suited for PCA applications. Numerical experiments are given in Section 4, and an application to sparse PCA is proposed. A technical Appendix contains all the proofs.

2. Model and Definition of the Estimator

To impose sparsity restrictions on the covariance matrix $\mathbf{\Sigma}$, our approach is based on an approximation of the process in a finite dictionary of (not necessarily orthogonal) basis functions $g_m : \mathbb{T} \to \mathbb{R}$ for $m = 1, \ldots, M$. Suppose that

$$X(t) \approx \sum_{m=1}^{M} a_m g_m(t), \tag{2}$$

where $a_m, m = 1, \ldots, M$ are real valued random variables, and that for each trajectory X_i

$$X_i(t_j) \approx \sum_{m=1}^{M} a_{i,m} g_m(t_j). \tag{3}$$

The notation $\approx$ means that the process X can be well approximated into the dictionary. A precise meaning of this will be discussed later on. Then (3) can be written in matrix notation as:

$$\boldsymbol{X}_i \approx \boldsymbol{G}\boldsymbol{a}_i,\ i=1,...,N$$

where $\boldsymbol{G}$ is the $n \times M$ matrix with entries

$$\boldsymbol{G}_{jm} = g_m(t_j) \text{ for } 1 \leq j \leq n \text{ and } 1 \leq m \leq M,$$

and $\boldsymbol{a}_i$ is the $M \times 1$ random vector of components $a_{i,m}$, with $1 \leq m \leq M$.

Recall that we want to estimate the covariance matrix $\boldsymbol{\Sigma} = \mathbb{E}\left(\boldsymbol{X}\boldsymbol{X}^\top\right)$ from the noisy observations (1). Since $\boldsymbol{X} \approx \boldsymbol{G}\boldsymbol{a}$ with $\boldsymbol{a} = (a_m)_{1\leq m\leq M}$ with a_m as in (2), it follows that

$$\boldsymbol{\Sigma} \approx \mathbb{E}\left(\boldsymbol{G}\boldsymbol{a}(\boldsymbol{G}\boldsymbol{a})^\top\right) = \mathbb{E}\left(\boldsymbol{G}\boldsymbol{a}\boldsymbol{a}^\top\boldsymbol{G}^\top\right) = \boldsymbol{G}\boldsymbol{\Psi}^*\boldsymbol{G}^\top \text{ with } \boldsymbol{\Psi}^* = \mathbb{E}\left(\boldsymbol{a}\boldsymbol{a}^\top\right).$$

Given the noisy observations $\widetilde{\boldsymbol{X}_i}$ as in (1) with $i=1,...,N$, consider the following matrix regression model

$$\widetilde{\boldsymbol{X}_i}\widetilde{\boldsymbol{X}_i}^\top = \boldsymbol{\Sigma} + \boldsymbol{U}_i + \boldsymbol{W}_i\ i=1,\ldots,N, \tag{4}$$

where $\boldsymbol{U}_i = \boldsymbol{X}_i\boldsymbol{X}_i^\top - \boldsymbol{\Sigma}$ are i.i.d centered matrix errors, and

$$\boldsymbol{W}_i = \mathcal{E}_i\mathcal{E}_i^\top \in \mathbb{R}^{n\times n} \text{ where } \mathcal{E}_i = (\mathcal{E}_i(t_1),...,\mathcal{E}_i(t_n))^\top, i=1,\ldots,N.$$

The size M of the dictionary can be very large, but it is expected that the process X has a sparse expansion in this basis, meaning that, in approximation (2), many of the random coefficients a_m are close to zero. We are interested in obtaining an estimate of the covariance $\boldsymbol{\Sigma}$ in the form $\widehat{\boldsymbol{\Sigma}} = \boldsymbol{G}\widehat{\boldsymbol{\Psi}}\boldsymbol{G}^\top$ such that $\widehat{\boldsymbol{\Psi}}$ is a symmetric $M \times M$ matrix with many zero rows (and so, by symmetry, many corresponding zero columns). Note that setting the k-th row of $\widehat{\boldsymbol{\Psi}}$ to $\mathbf{0} \in \mathbb{R}^M$ means to remove the function g_k from the set of basis functions $(g_m)_{1\leq m\leq M}$ in the function expansion associated to $\boldsymbol{G}$.

Let us now explain how to select a sparse set of rows/columns in the matrix $\widehat{\boldsymbol{\Psi}}$. For this, we use a group Lasso approach to threshold some rows/columns of $\widehat{\boldsymbol{\Psi}}$ which corresponds to removing some basis functions in the approximation of the process X. For two $p \times p$ matrices $\boldsymbol{A}, \boldsymbol{B}$ define the inner product $\langle \boldsymbol{A}, \boldsymbol{B}\rangle_F := tr\left(\boldsymbol{A}^\top\boldsymbol{B}\right)$ and the associated Frobenius norm $\|\boldsymbol{A}\|_F^2 := tr\left(\boldsymbol{A}^\top\boldsymbol{A}\right)$. Let $\mathcal{S}_M$ denote the set of $M \times M$ symmetric matrices with real entries. We define the group Lasso estimator of the covariance matrix $\boldsymbol{\Sigma}$ by

$$\widehat{\boldsymbol{\Sigma}}_\lambda = \boldsymbol{G}\widehat{\boldsymbol{\Psi}}_\lambda\boldsymbol{G}^\top \in \mathbb{R}^{n\times n}, \tag{5}$$

where $\widehat{\boldsymbol{\Psi}}_\lambda$ is the solution of the following optimization problem:

$$\widehat{\boldsymbol{\Psi}}_\lambda = \underset{\boldsymbol{\Psi}\in\mathcal{S}_M}{\operatorname{argmin}} \left\{ \frac{1}{N}\sum_{i=1}^N \left\| \widetilde{\boldsymbol{X}_i}\widetilde{\boldsymbol{X}_i}^\top - \boldsymbol{G}\boldsymbol{\Psi}\boldsymbol{G}^\top \right\|_F^2 + 2\lambda \sum_{k=1}^M \gamma_k \sqrt{\sum_{m=1}^M \Psi_{mk}^2} \right\}, \tag{6}$$

where $\boldsymbol{\Psi} = (\Psi_{mk})_{1\leq m,k\leq M} \in \mathbb{R}^{M\times M}$, λ is a positive number and γ_k are some weights whose values will be discuss later on. In (6), the penalty term imposes to give preference to solutions with components $\boldsymbol{\Psi}_k = \mathbf{0}$, where $(\boldsymbol{\Psi}_k)_{1\leq k\leq M}$ denotes the columns of $\boldsymbol{\Psi}$. Recall that $\widetilde{\boldsymbol{S}} = \frac{1}{N}\sum_{i=1}^N \widetilde{\boldsymbol{X}_i}\widetilde{\boldsymbol{X}_i}^\top$ denotes

the sample covariance matrix from the noisy observations (1). It can be checked that minimizing the criterion (6) is equivalent to

$$\widehat{\boldsymbol{\Psi}}_{\lambda} = \underset{\boldsymbol{\Psi}\in\mathcal{S}_M}{\operatorname{argmin}} \left\{ \left\| \widetilde{\boldsymbol{S}} - \boldsymbol{G}\boldsymbol{\Psi}\boldsymbol{G}^{\top} \right\|_F^2 + 2\lambda \sum_{k=1}^{M} \gamma_k \sqrt{\sum_{m=1}^{M} \Psi_{mk}^2} \right\}. \tag{7}$$

Thus $\widehat{\boldsymbol{\Psi}}_{\lambda} \in \mathbb{R}^{M\times M}$ can be interpreted as a group Lasso estimator of $\boldsymbol{\Sigma}$ in the following matrix regression model

$$\widetilde{\boldsymbol{S}} = \boldsymbol{\Sigma} + \boldsymbol{U} + \boldsymbol{W} \approx \boldsymbol{G}\boldsymbol{\Psi}^{*}\boldsymbol{G}^{\top} + \boldsymbol{U} + \boldsymbol{W}, \tag{8}$$

where $\boldsymbol{U} \in \mathbb{R}^{n\times n}$ is a centered error matrix given by $\boldsymbol{U} = \frac{1}{N}\sum_{i=1}^{N} \boldsymbol{U}_i$ and $\boldsymbol{W} = \frac{1}{N}\sum_{i=1}^{N} \boldsymbol{W}_i$. In the above regression model (8), there are two errors terms of a different nature. The term $\boldsymbol{W}$ corresponds to the additive Gaussian errors $\mathcal{E}_1, ..., \mathcal{E}_N$ in model (1), while the term $\boldsymbol{U} = \boldsymbol{S} - \boldsymbol{\Sigma}$ represents the difference between the (unobserved) sample covariance matrix $\boldsymbol{S}$ and the matrix $\boldsymbol{\Sigma}$ that we want to estimate.

This approach can be interpreted as a thresholding procedure of the entries of an empirical matrix. To see this, consider the simple case where $M = n$ and the basis functions and observations points are chosen such that the matrix $\boldsymbol{G}$ is orthogonal. Let $\boldsymbol{Y} = \boldsymbol{G}^{\top}\widetilde{\boldsymbol{S}}\boldsymbol{G}$ be a transformation of the empirical covariance matrix $\widetilde{\boldsymbol{S}}$. In the orthogonal case, the following proposition shows that the group Lasso estimator $\widehat{\boldsymbol{\Psi}}_{\lambda}$ defined by (7) consists in thresholding the columns/rows of $\boldsymbol{Y}$ whose ℓ_2-norm is too small, and in multiplying the other columns/rows by weights between 0 and 1. Hence, the group Lasso estimate (7) can be interpreted as covariance estimation by soft-thresholding the columns/rows of $\boldsymbol{Y}$.

Proposition 1 *Suppose that $M = n$ and that $\boldsymbol{G}^{\top}\boldsymbol{G} = \boldsymbol{I}_n$ where $\boldsymbol{I}_n$ denotes the identity matrix of size $n \times n$. Let $\boldsymbol{Y} = \boldsymbol{G}^{\top}\widetilde{\boldsymbol{S}}\boldsymbol{G}$. Then, the group Lasso estimator $\widehat{\boldsymbol{\Psi}}_{\lambda}$ defined by (7) is the $n \times n$ symmetric matrix whose entries are given by*

$$\left(\widehat{\boldsymbol{\Psi}}_{\lambda}\right)_{mk} = \begin{cases} 0 & \text{if} \quad \sqrt{\sum_{j=1}^{M} \boldsymbol{Y}_{jk}^2} \leq \lambda\gamma_k, \\ Y_{mk}\left(1 - \frac{\lambda\gamma_k}{\sqrt{\sum_{j=1}^{M} \boldsymbol{Y}_{mk}^2}}\right) & \text{if} \quad \sqrt{\sum_{j=1}^{M} \boldsymbol{Y}_{jk}^2} > \lambda\gamma_k, \end{cases}$$

for $1 \leq k, m \leq M$.

3. Consistency of the Group Lasso Estimator

In this section, we describe the statistical properties of the group Lasso estimator.

3.1 Notations and Main Assumptions

Let us begin by some definitions. For a symmetric $p \times p$ matrix $\boldsymbol{A}$ with real entries, $\rho_{min}(\boldsymbol{A})$ denotes the smallest eigenvalue of $\boldsymbol{A}$, and $\rho_{max}(\boldsymbol{A})$ denotes the largest eigenvalue of $\boldsymbol{A}$. For $\beta \in \mathbb{R}^q$, $\|\beta\|_{\ell_2}$ denotes the usual Euclidean norm of β. For $p \times q$ matrix $\boldsymbol{A}$ with real entries, $\|\boldsymbol{A}\|_2 = \sup_{\beta\in\mathbb{R}^q,\, \beta\neq 0} \frac{\|\boldsymbol{A}\beta\|_{\ell_2}}{\|\beta\|_{\ell_2}}$ denotes the operator norm of $\boldsymbol{A}$. Recall that if $\boldsymbol{A}$ is a non negative definite matrix with $p = q$ then $\|\boldsymbol{A}\|_2 = \rho_{max}(\boldsymbol{A})$.

Let $\Psi \in \mathcal{S}_M$ and β a vector in $\mathbb{R}^M$. For a subset $J \subset \{1,\ldots,M\}$ of indices of cardinality $|J|$, then β_J is the vector in $\mathbb{R}^M$ that has the same coordinates as β on J and zeros coordinates on the complement J^c of J. The $n \times |J|$ matrix obtained by removing the columns of $\boldsymbol{G}$ whose indices are not in J is denoted by $\boldsymbol{G}_J$. The sparsity of Ψ is defined as its number of non-zero columns (and thus by symmetry non-zero rows) namely

Definition 2 *For $\Psi \in \mathcal{S}_M$, the sparsity of Ψ is*

$$\mathcal{M}(\Psi) = \#\{k : \Psi_k \neq \mathbf{0}\}.$$

Then, let us introduce the following quantities that control the minimal eigenvalues of submatrices of small size extracted from the matrix $\boldsymbol{G}^\top \boldsymbol{G}$, and the correlations between the columns of $\boldsymbol{G}$:

Definition 3 *Let $0 < s \leq M$. Then,*

$$\rho_{\min}(s) := \inf_{\substack{J \subset \{1,\ldots,M\} \\ |J| \leq s}} \left(\frac{\beta_J^\top \boldsymbol{G}^\top \boldsymbol{G} \beta_J}{\|\beta_J\|_{\ell_2}^2} \right) = \inf_{\substack{J \subset \{1,\ldots,M\} \\ |J| \leq s}} \rho_{\min}\left(\boldsymbol{G}_J^\top \boldsymbol{G}_J \right).$$

Definition 4 *The mutual coherence $\theta(\boldsymbol{G})$ of the columns $\boldsymbol{G}_k$, $k = 1,\ldots,M$ of $\boldsymbol{G}$ is defined as*

$$\theta(\boldsymbol{G}) := \max\left\{ \left| \boldsymbol{G}_{k'}^\top \boldsymbol{G}_k \right|,\ k \neq k',\ 1 \leq k, k' \leq M \right\},$$

and let

$$\boldsymbol{G}_{\max}^2 := \max\left\{ \|\boldsymbol{G}_k\|_{\ell_2}^2,\ 1 \leq k \leq M \right\}.$$

To derive oracle inequalities showing the consistency of the group Lasso estimator $\widehat{\Psi}_\lambda$ the correlations between the columns of $\boldsymbol{G}$ (measured by $\theta(\boldsymbol{G})$) should not be too large when compared to the minimal eigenvalues of small matrices extracted from $\boldsymbol{G}^\top \boldsymbol{G}$, which is formulated in the following assumption:

Assumption 1 *Let $c_0 > 0$ be some constant and $0 < s \leq M$. Then*

$$\theta(\boldsymbol{G}) < \frac{\rho_{\min}(s)^2}{c_0 \rho_{\max}(\boldsymbol{G}^\top \boldsymbol{G}) s}.$$

Assumption 1 is inspired by recent results in Bickel et al. (2009) on the consistency of Lasso estimators in the standard nonparametric regression model using a large dictionary of basis functions. In Bickel et al. (2009), a general condition called *restricted eigenvalue assumption* is introduced to control the minimal eigenvalues of the Gram matrix associated to the dictionary over sets of sparse vectors. In the setting of nonparametric regression, a condition similar to Assumption 1 is given in Bickel et al. (2009) as an example for which the restricted eigenvalue assumption holds.

Let us give some examples for which Assumption 1 is satisfied. If $M \leq n$ and the design points are chosen such that the columns of the matrix $\boldsymbol{G}$ are orthonormal vectors in $\mathbb{R}^n$, then for any $0 < s \leq M$ one has that $\rho_{\min}(s) = 1$ and $\theta(\boldsymbol{G}) = 0$ and thus Assumption 1 holds for any value of c_0 and s.

Now, suppose that the columns of $\boldsymbol{G}$ are normalized to one, that is, $\|\boldsymbol{G}_k\|_{\ell_2} = 1$, $k = 1,\ldots,M$ implying that $\boldsymbol{G}_{\max} = 1$. Let $\beta \in \mathbb{R}^M$. Then, for any $J \subset \{1,\ldots,M\}$ with $|J| \leq s \leq \min(n,M)$

$$\beta_J^\top \boldsymbol{G}^\top \boldsymbol{G} \beta_J \geq \|\beta_J\|_{\ell_2}^2 - \theta(\boldsymbol{G}) s \|\beta_J\|_{\ell_2}^2,$$

which implies that

$$\rho_{\min}(s) \geq 1 - \theta(\boldsymbol{G}) s.$$

Therefore, if $(1-\theta(\boldsymbol{G})(s-1))^2 > c_0 \theta(\boldsymbol{G}) \rho_{\max}(\boldsymbol{G}^\top \boldsymbol{G}) s$, then Assumption 1 is satisfied.

Let us now specify the law of the stochastic process X. For this, recall that for a real-valued random variable Z, the ψ_α Orlicz norm of Z is

$$\|Z\|_{\psi_\alpha} := \inf \left\{ C > 0 \, ; \, \mathbb{E} \exp\left(\frac{|Z|^\alpha}{C^\alpha} \right) \leq 2 \right\}.$$

Such Orlicz norms are useful to characterize the tail behavior of random variables. Indeed, if $\|Z\|_{\psi_\alpha} < +\infty$ then this is equivalent to assuming that there exists two constants $K_1, K_2 > 0$ such that for all $x > 0$

$$\mathbb{P}(|Z| \geq x) \leq K_1 \exp\left(-\frac{x^\alpha}{K_2^\alpha} \right),$$

(see for example Mendelson and Pajor, 2006 for more details on Orlicz norms of random variables) . Therefore, if $\|Z\|_{\psi_2} < +\infty$ then Z is said to have a sub-Gaussian behavior and if $\|Z\|_{\psi_1} < +\infty$ then Z is said to have a sub-Exponential behavior. In the next sections, oracle inequalities for the group Lasso estimator will be derived under the following assumption on X:

Assumption 2 *The random vector* $\boldsymbol{X} = (X(t_1), ..., X(t_n))^\top \in \mathbb{R}^n$ *is such that*

(A1) *There exists* $\rho(\boldsymbol{\Sigma}) > 0$ *such that, for all vector* $\beta \in \mathbb{R}^n$ *with* $\|\beta\|_{\ell_2} = 1$, *then* $\left(\mathbb{E}|\boldsymbol{X}^\top \beta|^4\right)^{1/4} < \rho(\boldsymbol{\Sigma})$.

(A2) *Set* $Z = \|\boldsymbol{X}\|_{\ell_2}$. *There exists* $\alpha \geq 1$ *such that* $\|Z\|_{\psi_\alpha} < +\infty$.

Note that **(A1)** implies that $\|\boldsymbol{\Sigma}\|_2 \leq \rho(\boldsymbol{\Sigma})^2$. Indeed, one has that

$$\begin{aligned} \|\boldsymbol{\Sigma}\|_2 = \rho_{\max}(\boldsymbol{\Sigma}) &= \sup_{\beta \in \mathbb{R}^n, \, \|\beta\|_{\ell_2}=1} \beta^\top \boldsymbol{\Sigma} \beta = \sup_{\beta \in \mathbb{R}^n, \, \|\beta\|_{\ell_2}=1} \mathbb{E} \beta^\top \boldsymbol{X} \boldsymbol{X}^\top \beta \\ &= \sup_{\beta \in \mathbb{R}^n, \, \|\beta\|_{\ell_2}=1} \mathbb{E}|\beta^\top \boldsymbol{X}|^2 \leq \sup_{\beta \in \mathbb{R}^n, \, \|\beta\|_{\ell_2}=1} \sqrt{\mathbb{E}|\beta^\top \boldsymbol{X}|^4} \leq \rho^2(\boldsymbol{\Sigma}). \end{aligned}$$

When X is a Gaussian process, it follows that for any $\beta \in \mathbb{R}^n$ with $\|\beta\|_{\ell_2} = 1$ then $\left(\mathbb{E}|\boldsymbol{X}^\top \beta|^4\right)^{1/4} = 3^{1/4} \left(\beta^\top \boldsymbol{\Sigma} \beta\right)^{1/2}$ since $\boldsymbol{X}^\top \beta \sim N(0, \beta^\top \boldsymbol{\Sigma} \beta)$. Therefore, under the assumption that X is a Gaussian process, Assumption **(A1)** holds with $\rho(\boldsymbol{\Sigma}) = 3^{1/4} \|\boldsymbol{\Sigma}\|_2^{1/2}$.

Assumption **(A2)** requires that $\|Z\|_{\psi_\alpha} < +\infty$, where $Z = \|\boldsymbol{X}\|_{\ell_2}$. The following proposition provides some examples where such an assumption holds.

Proposition 5 *Let* $Z = \|\boldsymbol{X}\|_{\ell_2} = \left(\sum_{i=1}^n |X(t_i)|^2\right)^{1/2}$. *Then*

- *If X is a Gaussian process*

$$\|Z\|_{\psi_2} < \sqrt{8/3}\sqrt{tr(\mathbf{\Sigma})}.$$

- *If the random process X is such that $\|Z\|_{\psi_2} < +\infty$, and there exists a constant C_1 such that $\|\mathbf{\Sigma}_{ii}^{-1/2}|X(t_i)|\|_{\psi_2} \leq C_1$ for all $i = 1,\dots,n$, then*

$$\|Z\|_{\psi_2} < C_1\sqrt{tr(\mathbf{\Sigma})}.$$

- *If X is a bounded process, meaning that there exists a constant $R > 0$ such that for all $t \in \mathbb{T}$, $|X(t)| \leq R$, then for any $\alpha \geq 1$,*

$$\|Z\|_{\psi_\alpha} \leq \sqrt{n}R(\log 2)^{-1/\alpha}.$$

Assumption 2 will be used to control the deviation in operator norm between the sample covariance matrix $\boldsymbol{S}$ and the true covariance matrix $\mathbf{\Sigma}$ in the sense of the following proposition whose proof follows from Theorem 2.1 in Mendelson and Pajor (2006).

Proposition 6 *Let $X_1,...,X_N$ be independent copies of the stochastic process X, let $Z = \|\boldsymbol{X}\|_{\ell_2}$ and $\boldsymbol{X}_i = (X_i(t_1),...,X_i(t_n))^\top$ for $i = 1,\dots,N$. Recall that $\boldsymbol{S} = \frac{1}{N}\sum_{i=1}^{N}\boldsymbol{X}_i\boldsymbol{X}_i^\top$ and $\mathbf{\Sigma} = \mathbb{E}\left(\boldsymbol{X}\boldsymbol{X}^\top\right)$. Suppose that X satisfies Assumption 2. Let $d = \min(n,N)$. Then, there exists a universal constant $\delta_* > 0$ such that for all $x > 0$*

$$\mathbb{P}\left(\left\|\boldsymbol{S} - \mathbf{\Sigma}\right\|_2 \geqslant \tau_{d,N,n}x\right) \leqslant \exp\left(-(\delta_*^{-1}x)^{\frac{\alpha}{2+\alpha}}\right), \tag{9}$$

where $\tau_{N,n} = \max(A_{N,n}^2, B_{N,n})$, with

$$A_{N,n} = \|Z\|_{\psi_\alpha}\frac{\sqrt{\log d}(\log N)^{1/\alpha}}{\sqrt{N}} \text{ and } B_{N,n} = \frac{\rho^2(\mathbf{\Sigma})}{\sqrt{N}} + \|\mathbf{\Sigma}\|_2^{1/2}A_{N,n}.$$

Let us briefly comment Proposition 6 in some specific cases. If X is Gaussian, then Proposition 5 implies that $A_{N,n} \leq A_{N,n,1}$, where

$$A_{N,n,1} = \sqrt{8/3}\sqrt{tr(\mathbf{\Sigma})}\frac{\sqrt{\log d}(\log N)^{1/\alpha}}{\sqrt{N}} \leq \sqrt{8/3}\,\|\mathbf{\Sigma}\|_2^{1/2}\sqrt{\frac{n}{N}}\sqrt{\log d}(\log N)^{1/\alpha}, \tag{10}$$

and in this case inequality (9) becomes

$$\mathbb{P}\left(\left\|\boldsymbol{S} - \mathbf{\Sigma}\right\|_2 \geqslant \max\left(A_{N,n,1}^2, B_{N,n,1}\right)x\right) \leqslant \exp\left(-(\delta_*^{-1}x)^{\frac{\alpha}{2+\alpha}}\right) \tag{11}$$

for all $x > 0$, where $B_{N,n,1} = \frac{\rho^2(\mathbf{\Sigma})}{\sqrt{N}} + \|\mathbf{\Sigma}\|_2^{1/2}A_{N,n,1}$.

If X is a bounded process by some constant $R > 0$, then using Proposition 5 and by letting $\alpha \to +\infty$, Proposition 6 implies that for all $x > 0$,

$$\mathbb{P}\left(\left\|\boldsymbol{S} - \mathbf{\Sigma}\right\|_2 \geqslant \max\left(A_{N,n,2}^2, B_{N,n,2}\right)x\right) \leqslant \exp\left(-\delta_*^{-1}x\right), \tag{12}$$

where

$$A_{N,n,2} = R\sqrt{\frac{n}{N}}\sqrt{\log d} \text{ and } B_{N,n,2} = \frac{\rho^2(\boldsymbol{\Sigma})}{\sqrt{N}} + \|\boldsymbol{\Sigma}\|_2^{1/2} A_{N,n,2}. \tag{13}$$

Contrary to the low-dimensional case ($n << N$), in a high-dimensional setting when $n >> N$ or when n and N are of the same magnitude ($\frac{n}{N} \to c > 0$ as $n, N \to +\infty$), inequalities (11) and (12) cannot be used to conclude that the norm $\left\|\boldsymbol{S} - \boldsymbol{\Sigma}\right\|_2$ concentrates around zero. Actually, it is well known that the sample covariance $\boldsymbol{S}$ is a bad estimator of $\boldsymbol{\Sigma}$ in a high-dimensional setting, and that without any further restriction on the structure of the covariance matrix $\boldsymbol{\Sigma}$, then $\boldsymbol{S}$ cannot be a consistent estimator. However, we would like to point out that Proposition 6 relates the quality of $\boldsymbol{S}$ to the "true dimensionality" of the vector $\boldsymbol{X} = (X(t_1), ..., X(t_n))^\top \in \mathbb{R}^n$ that is measured by the quantity $\|Z\|_{\psi_\alpha}$ with $Z = \|\boldsymbol{X}\|_{\ell_2}$. Indeed, if X is a low-dimensional Gaussian process such that $tr(\boldsymbol{\Sigma}) = 1$ then Proposition 6 and inequality (10) imply that

$$\mathbb{P}\left(\left\|\boldsymbol{S} - \boldsymbol{\Sigma}\right\|_2 \geqslant \max\left(A_N^2, B_N\right) x\right) \leqslant \exp\left(-(\delta_*^{-1} x)^{\frac{1}{2}}\right) \tag{14}$$

for all $x > 0$, where $A_N = \sqrt{8/3}\frac{\sqrt{\log N}(\log N)^{1/\alpha}}{\sqrt{N}}$ and $B_N = \frac{\rho^2(\boldsymbol{\Sigma})}{\sqrt{N}} + \|\boldsymbol{\Sigma}\|_2^{1/2} A_N$. Hence, inequality (14) shows that, under an assumption of low-dimensionality of the process X, the deviation in operator norm between $\boldsymbol{S}$ and $\boldsymbol{\Sigma}$ depends on the ratio $\frac{1}{N}$ and not on $\frac{n}{N}$, and thus the quality of $\boldsymbol{S}$ as an estimator of $\boldsymbol{\Sigma}$ is much better in such settings.

More generally, another assumption of low-dimensionality for the process X is to suppose that it has a sparse representation in a dictionary of basis functions, which may also improve the quality of $\boldsymbol{S}$ as an estimator of $\boldsymbol{\Sigma}$. To see this, consider the simplest case $X = X^0$, where the process X^0 has a sparse representation in the basis $(g_m)_{1 \leq m \leq M}$ given by

$$X^0(t) = \sum_{m \in J^*} a_m g_m(t),\ t \in \mathbb{T}, \tag{15}$$

where $J^* \subset \{1, \ldots, M\}$ is a subset of indices of cardinality $|J^*| = s_*$ and a_m, $m \in J^*$ are random coefficients (possibly correlated). Under such an assumption, the following proposition holds.

Proposition 7 *Suppose that $X = X^0$ with X^0 defined by (15) with $s_* \leq \min(n, M)$. Assume that X satisfies Assumption 2 and that the matrix $\boldsymbol{G}_{J^*}^\top \boldsymbol{G}_{J^*}$ is invertible, where $\boldsymbol{G}_{J^*}$ denotes the $n \times |J^*|$ matrix obtained by removing the columns of $\boldsymbol{G}$ whose indices are not in J^*. Then, there exists a universal constant $\delta_* > 0$ such that for all $x > 0$,*

$$\mathbb{P}\left(\left\|\boldsymbol{S} - \boldsymbol{\Sigma}\right\|_2 \geqslant \tilde{\tau}_{N,s_*} x\right) \leqslant \exp\left(-(\delta_*^{-1} x)^{\frac{\alpha}{2+\alpha}}\right),$$

where $\tilde{\tau}_{N,s_*} = \max(\tilde{A}_{N,s_*}^2, \tilde{B}_{N,s_*})$, with

$$\tilde{A}_{N,s_*} = \rho_{\max}^{1/2}\left(\boldsymbol{G}_{J^*}^\top \boldsymbol{G}_{J^*}\right) \|\tilde{Z}\|_{\psi_\alpha} \frac{\sqrt{\log d^*}(\log N)^{1/\alpha}}{\sqrt{N}},$$

and

$$\tilde{B}_{N,s_*} = \left(\frac{\rho_{\max}\left(\boldsymbol{G}_{J^*}^\top \boldsymbol{G}_{J^*}\right)}{\rho_{\min}\left(\boldsymbol{G}_{J^*}^\top \boldsymbol{G}_{J^*}\right)}\right) \frac{\rho^2(\boldsymbol{\Sigma})}{\sqrt{N}} + \left(\frac{\rho_{\max}\left(\boldsymbol{G}_{J^*}^\top \boldsymbol{G}_{J^*}\right)}{\rho_{\min}\left(\boldsymbol{G}_{J^*}^\top \boldsymbol{G}_{J^*}\right)}\right)^{1/2} \|\boldsymbol{\Sigma}\|_2^{1/2} \tilde{A}_{d^*,N,s_*},$$

with $d^* = \min(N, s_*)$ and $\tilde{Z} = \|\boldsymbol{a}_{J^*}\|_{\ell_2}$, where $\boldsymbol{a}_{J^*} = (\boldsymbol{G}_{J^*}^\top \boldsymbol{G}_{J^*})^{-1} \boldsymbol{G}_{J^*}^\top \boldsymbol{X} \in \mathbb{R}^{s_*}$.

Using Proposition 5 and Proposition 7 it follows that

- If $X = X^0$ is a Gaussian process then

$$\tilde{A}_{N,s_*} \leq \sqrt{8/3}\left(\frac{\rho_{\max}\left(\boldsymbol{G}_{J^*}^\top \boldsymbol{G}_{J^*}\right)}{\rho_{\min}\left(\boldsymbol{G}_{J^*}^\top \boldsymbol{G}_{J^*}\right)}\right)^{1/2} \|\boldsymbol{\Sigma}\|_2^{1/2}\sqrt{\frac{s_*}{N}}\sqrt{\log d^*}(\log N)^{1/\alpha} \tag{16}$$

- If $X = X^0$ is such that the random variables a_m are bounded by for some constant $R > 0$, then

$$\tilde{A}_{N,s_*} \leq R\|g\|_\infty \sqrt{\frac{s_*}{N}}\sqrt{\log d^*} \tag{17}$$

with $\|g\|_\infty = \max_{1\leq m\leq M}\|g_m\|_\infty$ where $\|g_m\|_\infty = \sup_{t\in\mathcal{T}}|g_m(t)|$.

Therefore, let us compare the bounds (16) and (17) with the inequalities (10) and (13). It follows that, in the case $X = X^0$, if the sparsity s_* of X in the dictionary is small compared to the number of time points n then the deviation between $\boldsymbol{S}$ and $\boldsymbol{\Sigma}$ is much smaller than in the general case without any assumption on the structure of $\boldsymbol{\Sigma}$. Obviously, the gain also depends on the control of the ratio $\frac{\rho_{\max}\left(\boldsymbol{G}_{J^*}^\top \boldsymbol{G}_{J^*}\right)}{\rho_{\min}\left(\boldsymbol{G}_{J^*}^\top \boldsymbol{G}_{J^*}\right)}$. Note that in the case of an orthonormal design ($M = n$ and $\boldsymbol{G}^\top\boldsymbol{G} = \boldsymbol{I}_n$) then $\rho_{\max}\left(\boldsymbol{G}_{J^*}^\top \boldsymbol{G}_{J^*}\right) = \rho_{\min}\left(\boldsymbol{G}_{J^*}^\top \boldsymbol{G}_{J^*}\right) = 1$ for any J^*, and thus the gain in operator norm between $\boldsymbol{S}$ and $\boldsymbol{\Sigma}$ clearly depends on the size of $\frac{s_*}{N}$ compared to $\frac{n}{N}$. Supposing that $X = X^0$ also implies that the operator norm of the error term $\boldsymbol{U}$ in the matrix regression model (8) is controlled by the ratio $\frac{s_*}{N}$ instead of the ratio $\frac{n}{N}$ when no assumptions are made on the structure of $\boldsymbol{\Sigma}$. This means that if X has a sparse representation in the dictionary then the error term $\boldsymbol{U}$ becomes smaller.

3.2 An Oracle Inequality for the Frobenius Norm

Consistency is first studied for the normalized Frobenius norm $\frac{1}{n}\|\boldsymbol{A}\|_F^2$ for an $n\times n$ matrix $\boldsymbol{A}$. The following theorem provides an oracle inequality for the group Lasso estimator $\widehat{\boldsymbol{\Sigma}}_\lambda = \boldsymbol{G}\widehat{\boldsymbol{\Psi}}_\lambda\boldsymbol{G}^\top$.

Theorem 8 *Assume that X satisfies Assumption 2. Let $\varepsilon > 0$ and $1 \leq s \leq \min(n, M)$. Suppose that Assumption 1 holds with $c_0 = 3 + 4/\varepsilon$, and that the covariance matrix $\boldsymbol{\Sigma}_{noise} = \mathbb{E}(\boldsymbol{W}_1)$ of the noise is positive-definite. Consider the group Lasso estimator $\widehat{\boldsymbol{\Sigma}}_\lambda$ defined by (5) with the choices*

$$\gamma_k = 2\|\boldsymbol{G}_k\|_{\ell_2}\sqrt{\rho_{\max}(\boldsymbol{G}\boldsymbol{G}^\top)},$$

and

$$\lambda = \|\boldsymbol{\Sigma}_{noise}\|_2\left(1 + \sqrt{\frac{n}{N}} + \sqrt{\frac{2\delta\log M}{N}}\right)^2 \text{ for some constant } \delta > 1.$$

Then, with probability at least $1 - M^{1-\delta}$ one has that

$$\begin{aligned}\frac{1}{n}\left\|\widehat{\boldsymbol{\Sigma}}_\lambda - \boldsymbol{\Sigma}\right\|_F^2 \quad \leq \quad & (1+\varepsilon)\inf_{\substack{\boldsymbol{\Psi}\in\mathcal{S}_M \\ \mathcal{M}(\boldsymbol{\Psi})\leq s}}\left(\frac{4}{n}\left\|\boldsymbol{G}\boldsymbol{\Psi}\boldsymbol{G}^\top - \boldsymbol{\Sigma}\right\|_F^2 + \frac{8}{n}\|\boldsymbol{S} - \boldsymbol{\Sigma}\|_F^2\right. \\ & \left. + C(\varepsilon)\frac{\boldsymbol{G}_{\max}^2\rho_{\max}(\boldsymbol{G}^\top\boldsymbol{G})}{\kappa_{s,c_0}^2}\|\boldsymbol{\Sigma}_{noise}\|_2^2\left(1 + \sqrt{\frac{n}{N}} + \sqrt{\frac{2\delta\log M}{N}}\right)^4\frac{\mathcal{M}(\boldsymbol{\Psi})}{n}\right),\end{aligned} \tag{18}$$

where $\kappa_{s,c_0}^2 = \rho_{\min}(s)^2 - c_0\theta(\boldsymbol{G})\rho_{\max}(\boldsymbol{G}^\top\boldsymbol{G})s$, *and* $C(\varepsilon) = 8\frac{\varepsilon}{1+\varepsilon}(1+2/\varepsilon)^2$.

The first term $\frac{1}{n}\left\|\boldsymbol{G}\boldsymbol{\Psi}\boldsymbol{G}^\top - \boldsymbol{\Sigma}\right\|_F^2$ in inequality (18) is the bias of the estimator $\widehat{\boldsymbol{\Sigma}}_\lambda$. It reflects the quality of the approximation of $\boldsymbol{\Sigma}$ by the set of matrices of the form $\boldsymbol{G}\boldsymbol{\Psi}\boldsymbol{G}^\top$, with $\boldsymbol{\Psi} \in \mathcal{S}_M$ and $\mathcal{M}(\boldsymbol{\Psi}) \leq s$. As an example, suppose that $X = X^0$, where the process X^0 has a sparse representation in the basis $(g_m)_{1\leq m\leq M}$ given by

$$X^0(t) = \sum_{m\in J^*} a_m g_m(t),\ t \in \mathbb{T},$$

where $J^* \subset \{1,\ldots,M\}$ is a subset of indices of cardinality $|J^*| = s_* \leq s$ and $a_m, m \in J^*$ are random coefficients. Then, in this case, since $s_* \leq s$ the bias term in (18) is equal to zero.

The second term $\frac{1}{n}\|\boldsymbol{S}-\boldsymbol{\Sigma}\|_F^2$ in (18) is a variance term as the empirical covariance matrix $\boldsymbol{S}$ is an unbiased estimator of $\boldsymbol{\Sigma}$. Using the inequality $\frac{1}{n}\|A\|_F^2 \leq \|A\|_2^2$ that holds for any $n\times n$ matrix A, it follows that $\frac{1}{n}\|\boldsymbol{S}-\boldsymbol{\Sigma}\|_F^2 \leq \|\boldsymbol{S}-\boldsymbol{\Sigma}\|_2^2$. Therefore, under the assumption that X has a sparse representation in the dictionary (for example when $X = X_0$ as above) then the variance term $\frac{1}{n}\|\boldsymbol{S}-\boldsymbol{\Sigma}\|_F^2$ is controlled by the ratio $\frac{s_*}{N} \leq \frac{s}{N}$ (see Proposition 7) instead of the ratio $\frac{n}{N}$ without any assumption on the structure of $\boldsymbol{\Sigma}$.

The third term in (18) is also a variance term due to the noise in the measurements (1). If there exists a constant $c > 0$ independent of n and N such that $\frac{n}{N} \leq c$ then the decay of this third variance term is essentially controlled by the ratio $\frac{\mathcal{M}(\boldsymbol{\Psi})}{n} \leq \frac{s}{n}$. Therefore, if $\mathcal{M}(\boldsymbol{\Psi}) \leq s$ with sparsity s much smaller than n then the variance of the group Lasso estimator $\widehat{\boldsymbol{\Sigma}}_\lambda$ is smaller than the variance of $\widetilde{\boldsymbol{S}}$. This shows some of the improvements achieved by regularization (7) of the empirical covariance matrix $\widetilde{\boldsymbol{S}}$ with a group Lasso penalty.

An important assumption of Theorem 8 is that the covariance matrix of the noise $\boldsymbol{\Sigma}_{noise} = \mathbb{E}(\boldsymbol{W}_1)$ is positive definite. This restriction is clearly necessary as illustrated by the following example: suppose that the contaminating process $\mathcal{E}(t) = \zeta g_1(t)$ with $\zeta \sim N(0,\sigma_1^2)$, implying that $\boldsymbol{\Sigma}_{noise} = \sigma_1^2\boldsymbol{g}_1\boldsymbol{g}_1^\top$ with $\boldsymbol{g}_1 = (g_1(t_1),\ldots,g_1(t_n))^\top$ has $n-1$ eigenvalues equal to zero. Now, suppose that $X(t) = a_2 g_2(t)$ with $a_2 \sim N(0,\sigma_2^2)$. If $\sigma_1 > \sigma_2$ then the group LASSO regularization alone cannot get rid of the additive error term without eliminating first the right component g_2. Hence, in such settings, group LASSO regularization does not yield to a consistent estimation of $\boldsymbol{\Sigma} = \sigma_2^2\boldsymbol{g}_2\boldsymbol{g}_2^\top$ with $\boldsymbol{g}_2 = (g_2(t_1),\ldots,g_2(t_n))^\top$.

3.3 An Oracle Inequality for the Operator Norm

The "normalized" Frobenius norm $\frac{1}{n}\left\|\widehat{\boldsymbol{\Sigma}}_\lambda - \boldsymbol{\Sigma}\right\|_F^2$, that is, the average of the eigenvalues of $\left(\widehat{\boldsymbol{\Sigma}}_\lambda - \boldsymbol{\Sigma}\right)^2$, can be viewed as a reasonable proxy for the operator norm $\left\|\widehat{\boldsymbol{\Sigma}}_\lambda - \boldsymbol{\Sigma}\right\|_2^2$ (maximum eigenvalue of $\left(\widehat{\boldsymbol{\Sigma}}_\lambda - \boldsymbol{\Sigma}\right)^2$). It is thus expected that the results of Theorem 8 imply that the group Lasso estimator $\widehat{\boldsymbol{\Sigma}}_\lambda$ is a good estimator of $\boldsymbol{\Sigma}$ in operator norm. Let us recall that controlling the operator norm enables to study the convergence of the eigenvectors and eigenvalues of $\widehat{\boldsymbol{\Sigma}}_\lambda$ by controlling of the angles between the eigenspaces of a population and a sample covariance matrix through the use of the $\sin(\theta)$ theorems in Davis and Kahan (1970).

Now, let us consider the case where X consists in noisy observations of the process X^0 (15) meaning that

$$\widetilde{X}(t_j) = X^0(t_j) + \mathcal{E}(t_j),\ j = 1,\dots,n, \tag{19}$$

where $\mathcal{E}$ is a second order Gaussian process $\mathcal{E}$ with zero mean and independent of X^0. In this case, one has that

$$\boldsymbol{\Sigma} = \boldsymbol{G}\boldsymbol{\Psi}^*\boldsymbol{G}^\top, \text{ where } \boldsymbol{\Psi}^* = \mathbb{E}\left(\boldsymbol{a}\boldsymbol{a}^\top\right),$$

where $\boldsymbol{a}$ is the random vector of $\mathbb{R}^M$ with $\boldsymbol{a}_m = a_m$ for $m \in J^*$ and $\boldsymbol{a}_m = 0$ for $m \notin J^*$. Therefore, using Theorem 8 by replacing s by $s^* = |J^*|$, since $\boldsymbol{\Psi}^* \in \{\boldsymbol{\Psi} \in \mathcal{S}_M : M(\boldsymbol{\Psi}) \leq s_*\}$, one can derive the following corrollary:

Corollary 9 *Suppose that the observations $\widetilde{X}_i(t_j)$ with $i = 1,...,N$ and $j = 1,\dots,n$ are i.i.d random variables from model (19) and that the conditions of Theorem 8 are satisfied with $1 \leq s = s_* \leq \min(n,M)$. Then, with probability at least $1 - M^{1-\delta}$ one has that*

$$\frac{1}{n}\left\|\widehat{\boldsymbol{\Sigma}}_\lambda - \boldsymbol{\Sigma}\right\|_F^2 \leq C_0\left(n,M,N,s_*,\boldsymbol{S},\boldsymbol{\Psi}^*,\boldsymbol{G},\boldsymbol{\Sigma}_{noise}\right),$$

where

$$C_0\left(n,M,N,s_*,\boldsymbol{S},\boldsymbol{\Psi}^*,\boldsymbol{G},\boldsymbol{\Sigma}_{noise}\right) = (1+\varepsilon)\left(\frac{8}{n}\left\|\boldsymbol{S} - \boldsymbol{G}\boldsymbol{\Psi}^*\boldsymbol{G}^\top\right\|_F^2 + C(\varepsilon)\frac{\boldsymbol{G}_{\max}^2\rho_{\max}(\boldsymbol{G}^\top\boldsymbol{G})}{\kappa_{s_*,c_0}^2}\lambda^2\frac{s_*}{n}\right).$$

To simplify notations, write $\widehat{\boldsymbol{\Psi}} = \widehat{\boldsymbol{\Psi}}_\lambda$, with $\widehat{\boldsymbol{\Psi}}_\lambda$ given by (7). Define $\hat{J}_\lambda \subset \{1,\dots,M\}$ as

$$\hat{J}_\lambda \equiv \hat{J} := \left\{k : \frac{\delta_k}{\sqrt{n}}\left\|\widehat{\boldsymbol{\Psi}}_k\right\|_{\ell_2} > C_1\left(n,M,N,s_*,\boldsymbol{S},\boldsymbol{\Psi}^*,\boldsymbol{G},\boldsymbol{\Sigma}_{noise}\right)\right\},\ \text{with } \delta_k = \frac{\|\boldsymbol{G}_k\|_{\ell_2}}{\boldsymbol{G}_{\max}}, \tag{20}$$

and $C_1\left(n,M,N,s_*,\boldsymbol{S},\boldsymbol{\Psi}^*,\boldsymbol{G},\boldsymbol{\Sigma}_{noise}\right) = C_1$ with

$$C_1 = \max\left(\gamma_{\max}^{-1}n^{-1/2}\frac{1+\varepsilon}{\lambda}\left\|\boldsymbol{S} - \boldsymbol{G}\boldsymbol{\Psi}^*\boldsymbol{G}^\top\right\|_F^2 ; \frac{4(1+\varepsilon)\sqrt{s_*}}{\varepsilon\kappa_{s_*,c_0}}\sqrt{C_0\left(n,M,N,s_*,\boldsymbol{S},\boldsymbol{\Psi}^*,\boldsymbol{G},\boldsymbol{\Sigma}_{noise}\right)}\right). \tag{21}$$

with $\gamma_{\max} = 2\boldsymbol{G}_{\max}\sqrt{\rho_{\max}(\boldsymbol{G}^\top\boldsymbol{G})}$. The set of indices $\hat{J}$ is an estimation of the set of active basis functions J^*. Note that such thresholding procedure (20) does not lead immediately to a practical way to choose the set $\hat{J}$. Indeed the constant C_1 in (20) depends on the a priori unknown sparsity s_* and on the amplitude of the noise in the matrix regression model (8) measured by the quantities $\frac{8}{n}\left\|\boldsymbol{S} - \boldsymbol{G}\boldsymbol{\Psi}^*\boldsymbol{G}^\top\right\|_F^2$ and $\|\boldsymbol{\Sigma}_{noise}\|_2^2$. Nevertheless, in Section 4 on numerical experiments we give a simple procedure to automatically threshold the ℓ_2-norm of the columns of the matrix $\widehat{\Psi}_\lambda$ that are two small.

Note that to estimate J^* we did not simply take $\hat{J} = \hat{J}_0 := \left\{k : \left\|\widehat{\boldsymbol{\Psi}}_k\right\|_{\ell_2} \neq 0\right\}$, but rather apply a thresholding step to discard the columns of $\widehat{\boldsymbol{\Psi}}$ whose ℓ_2-norm are too small. By doing so, we want to stress the fact that to obtain a consistent procedure with respect to the operator norm it is not sufficient to simply take $\hat{J} = \hat{J}_0$. A similar thresholding step is proposed in Lounici (2008) and Lounici et al. (2009) in the standard linear model to select a sparse set of active variables when using regularization by a Lasso or group-Lasso penalty. In the paper (Lounici, 2008), the second thresholding step used to estimate the true sparsity pattern depends on a unknown constant that is related to the amplitude of the unknown coefficients to estimate.

Then, the following theorem holds.

Theorem 10 *Under the assumptions of Corollary 9, for any solution of problem (7), we have that with probability at least* $1-M^{1-\delta}$,

$$\max_{1\leq k\leq M}\frac{\delta_k}{\sqrt{n}}\left\|\widehat{\boldsymbol{\Psi}}_k-\boldsymbol{\Psi}_k^*\right\|_{\ell_2}\leq C_1\left(n,M,N,s_*,\boldsymbol{S},\boldsymbol{\Psi}^*,\boldsymbol{G},\boldsymbol{\Sigma}_{noise}\right).$$

If in addition

$$\min_{k\in J^*}\frac{\delta_k}{\sqrt{n}}\left\|\boldsymbol{\Psi}_k^*\right\|_{\ell_2}>2C_1\left(n,M,N,s_*,\boldsymbol{S},\boldsymbol{\Psi}^*,\boldsymbol{G},\boldsymbol{\Sigma}_{noise}\right) \tag{22}$$

then with the same probability the set of indices $\hat{J}$*, defined by (20), estimates correctly the true set of active basis functions* J^**, that is* $\hat{J}=J^*$ *with probability at least* $1-M^{1-\delta}$.

The results of Theorem 10 indicate that if the ℓ_2-norm of the columns of $\boldsymbol{\Psi}_k^*$ for $k\in J^*$ are sufficiently large with respect to the level of noise in the matrix regression model (8) and the sparsity s_*, then $\hat{J}$ is a consistent estimation of the active set of variables. Indeed, if $\mathcal{M}(\boldsymbol{\Psi}^*)=s_*$, then by symmetry the columns of $\boldsymbol{\Psi}^*$ such $\boldsymbol{\Psi}_k^*\neq 0$ have exactly s_* non-zero entries. Hence, the condition (22) means that the ℓ_2-norm of $\boldsymbol{\Psi}_k^*\neq 0$ (normalized by $\frac{\delta_k}{\sqrt{n}}$) has to be larger than $\frac{4(1+\varepsilon)}{\varepsilon\kappa_{s_*,c_0}}\sqrt{s_*}\sqrt{C_0}$. A simple condition to satisfy such an assumption is that the amplitude of the s_* non-vanishing entries of $\boldsymbol{\Psi}_k^*\neq 0$ are larger than $\frac{\sqrt{n}}{\delta_k}\frac{4(1+\varepsilon)}{\varepsilon\kappa_{s_*,c_0}}\sqrt{C_0}$ which can be interpreted as a kind of measure of the noise in model (8). This suggests to take as a final estimator of $\boldsymbol{\Sigma}$ the following matrix:

$$\widehat{\boldsymbol{\Sigma}}_{\hat{J}}=\boldsymbol{G}_{\hat{J}}\widehat{\boldsymbol{\Psi}}_{\hat{J}}\boldsymbol{G}_{\hat{J}} \tag{23}$$

where $\boldsymbol{G}_{\hat{J}}$ denotes the $n\times|\hat{J}|$ matrix obtained by removing the columns of $\boldsymbol{G}$ whose indices are not in $\hat{J}$, and

$$\widehat{\boldsymbol{\Psi}}_{\hat{J}}=\underset{\boldsymbol{\Psi}\in\mathcal{S}_{|\hat{J}|}}{\operatorname{argmin}}\left\{\left\|\widetilde{\boldsymbol{S}}-\boldsymbol{G}_{\hat{J}}\boldsymbol{\Psi}\boldsymbol{G}_{\hat{J}}^\top\right\|_F^2\right\},$$

where $\mathcal{S}_{|\hat{J}|}$ denotes the set of $|\hat{J}|\times|\hat{J}|$ symmetric matrices. Note that if $\boldsymbol{G}_{\hat{J}}^\top\boldsymbol{G}_{\hat{J}}$ is invertible, then

$$\widehat{\boldsymbol{\Psi}}_{\hat{J}}=\left(\boldsymbol{G}_{\hat{J}}^\top\boldsymbol{G}_{\hat{J}}\right)^{-1}\boldsymbol{G}_{\hat{J}}^\top\widetilde{\boldsymbol{S}}\boldsymbol{G}_{\hat{J}}\left(\boldsymbol{G}_{\hat{J}}^\top\boldsymbol{G}_{\hat{J}}\right)^{-1}.$$

Let us recall that if the observations are i.i.d random variables from model (19) then

$$\boldsymbol{\Sigma}=\boldsymbol{G}\boldsymbol{\Psi}^*\boldsymbol{G}^\top,$$

where $\boldsymbol{\Psi}^*=\mathbb{E}\left(\boldsymbol{a}\boldsymbol{a}^\top\right)$, and $\boldsymbol{a}$ is the random vector of $\mathbb{R}^M$ with $\boldsymbol{a}_m=a_m$ for $m\in J^*$ and $\boldsymbol{a}_m=0$ for $m\notin J^*$. Then, define the random vector $\boldsymbol{a}_{J^*}\in\mathbb{R}^{J^*}$ whose coordinates are the random coefficients a_m for $m\in J^*$. Let $\boldsymbol{\Psi}_{J^*}=\mathbb{E}\left(\boldsymbol{a}_{J^*}\boldsymbol{a}_{J^*}^\top\right)$ and denote by $\boldsymbol{G}_{J^*}$ the $n\times|J^*|$ matrix obtained by removing the columns of $\boldsymbol{G}$ whose indices are not in J^*. Note that $\boldsymbol{\Sigma}=\boldsymbol{G}_{J^*}\boldsymbol{\Psi}_{J^*}\boldsymbol{G}_{J^*}^\top$.

Assuming that $\boldsymbol{G}_{J^*}^\top\boldsymbol{G}_{J^*}$ is invertible, define the matrix

$$\boldsymbol{\Sigma}_{J^*}=\boldsymbol{\Sigma}+\boldsymbol{G}_{J^*}(\boldsymbol{G}_{J^*}^\top\boldsymbol{G}_{J^*})^{-1}\boldsymbol{G}_{J^*}^\top\boldsymbol{\Sigma}_{noise}\boldsymbol{G}_{J^*}\left(\boldsymbol{G}_{J^*}^\top\boldsymbol{G}_{J^*}\right)^{-1}\boldsymbol{G}_{J^*}^\top. \tag{24}$$

Then, the following theorem gives a control of deviation between $\widehat{\boldsymbol{\Sigma}}_{\hat{J}}$ and $\boldsymbol{\Sigma}_{J^*}$ in operator norm.

Theorem 11 *Suppose that the observations are i.i.d random variables from model (19) and that the conditions of Theorem 8 are satisfied with* $1 \leq s = s_* \leq \min(n,M)$. *Suppose that* $\boldsymbol{G}_{J^*}^\top \boldsymbol{G}_{J^*}$ *is an invertible matrix, and that*

$$\min_{k\in J^*} \frac{\delta_k}{\sqrt{n}} \|\boldsymbol{\Psi}_k^*\|_{\ell_2} > 2C_1\left(n,M,N,s_*,\boldsymbol{S},\boldsymbol{\Psi}^*,\boldsymbol{G},\boldsymbol{\Sigma}_{noise}\right),$$

where $C_1\left(n,M,N,s_*,\boldsymbol{S},\boldsymbol{\Psi}^*,\boldsymbol{G},\boldsymbol{\Sigma}_{noise}\right)$ *is the constant defined in (21). Let*

$$\boldsymbol{Y} = \left(\boldsymbol{G}_{J^*}^\top \boldsymbol{G}_{J^*}\right)^{-1} \boldsymbol{G}_{J^*}^\top \widetilde{\boldsymbol{X}}$$

and $\tilde{Z} = \|\boldsymbol{Y}\|_{\ell_2}$. *Let* $\rho\left(\boldsymbol{\Sigma}_{noise}\right) = \left(\sup_{\beta\in\mathbb{R}^n,\|\beta\|_{\ell_2}=1} \mathbb{E}|\mathcal{E}^\top\beta|^4\right)^{1/4}$ *where* $\mathcal{E} = \left(\mathcal{E}(t_1),...,\mathcal{E}(t_n)\right)^\top$. *Then, with probability at least* $1 - M^{1-\delta} - M^{-\left(\frac{\delta_\star}{\delta_*}\right)^{\frac{\alpha}{2+\alpha}}}$, *with* $\delta > 1$ *and* $\delta_\star > \delta_*$ *one has that*

$$\left\|\widehat{\boldsymbol{\Sigma}}_{\hat{J}} - \boldsymbol{\Sigma}_{J^*}\right\|_2 \leq \rho_{\max}\left(\boldsymbol{G}_{J^*}^\top \boldsymbol{G}_{J^*}\right) \tilde{\tau}_{N,s_*} \delta_\star \left(\log(M)\right)^{\frac{2+\alpha}{\alpha}},$$

where $\tilde{\tau}_{N,s_*} = \max(\tilde{A}_{N,s_*}^2, \tilde{B}_{N,s_*})$, *with* $\tilde{A}_{N,s_*} = \|\tilde{Z}\|_{\psi_\alpha} \frac{\sqrt{\log d^*}(\log N)^{1/\alpha}}{\sqrt{N}}$ *and*

$$\tilde{B}_{N,s_*} = \frac{\tilde{\rho}^2(\boldsymbol{\Sigma},\boldsymbol{\Sigma}_{noise})\rho_{\min}^{-1}\left(\boldsymbol{G}_{J^*}^\top \boldsymbol{G}_{J^*}\right)}{\sqrt{N}} + \left(\|\boldsymbol{\Psi}_{J^*}\|_2 + \rho_{\min}^{-1}\left(\boldsymbol{G}_{J^*}^\top \boldsymbol{G}_{J^*}\right)\|\boldsymbol{\Sigma}_{noise}\|_2\right)^{1/2} \tilde{A}_{N,s_*},$$

where $d^* = \min(N, s_*)$ *and* $\tilde{\rho}(\boldsymbol{\Sigma},\boldsymbol{\Sigma}_{noise}) = 8^{1/4}\left(\rho^4(\boldsymbol{\Sigma}) + \rho^4(\boldsymbol{\Sigma}_{noise})\right)^{1/4}$.

First note that the above theorem gives a deviation in operator norm from $\widehat{\boldsymbol{\Sigma}}_{\hat{J}}$ to the matrix $\boldsymbol{\Sigma}_{J^*}$ (24) which is not equal to the true covariance $\boldsymbol{\Sigma}$ of X at the design points. Indeed, even if we know the true sparsity set J^*, the additive noise in the measurements in model (1) complicates the estimation of $\boldsymbol{\Sigma}$ in operator norm. However, although $\boldsymbol{\Sigma}_{J^*} \neq \boldsymbol{\Sigma}$, they can have the same eigenvectors if the structure of the additive noise matrix term in (24) is not too complex. As an example, consider the case of an additive white noise, for which $\boldsymbol{\Sigma}_{noise} = \sigma^2 \boldsymbol{I}_n$ where σ is the level of noise and $\boldsymbol{I}_n$ the $n \times n$ identity matrix. Under such an assumption, if we further suppose for simplicity that $(\boldsymbol{G}_{J^*}^\top \boldsymbol{G}_{J^*})^{-1} = \boldsymbol{I}_{s_*}$, then $\boldsymbol{\Sigma}_{J^*} = \boldsymbol{\Sigma} + \sigma^2 \boldsymbol{G}_{J^*}(\boldsymbol{G}_{J^*}^\top \boldsymbol{G}_{J^*})^{-1}\boldsymbol{G}_{J^*}^\top = \boldsymbol{\Sigma} + \sigma^2 \boldsymbol{I}_n$ and clearly $\boldsymbol{\Sigma}_{J^*}$ and $\boldsymbol{\Sigma}$ have the same eigenvectors. Therefore, the eigenvectors of $\widehat{\boldsymbol{\Sigma}}_{\hat{J}}$ can be used as estimators of the eigenvectors of $\boldsymbol{\Sigma}$ which is suitable for the sparse PCA application described in the next section on numerical experiments.

Let us illustrate the implications of Theorem 11 on a simple example. If X is Gaussian, the random vector $\boldsymbol{Y} = \left(\boldsymbol{G}_{J^*}^\top \boldsymbol{G}_{J^*}\right)^{-1} \boldsymbol{G}_{J^*}^\top (\boldsymbol{X} + \mathcal{E})$ is also Gaussian and Proposition 5 can be used to prove that

$$\begin{aligned}
\|\tilde{Z}\|_{\psi_2} &\leq \sqrt{8/3}\sqrt{tr\left(\left(\boldsymbol{G}_{J^*}^\top \boldsymbol{G}_{J^*}\right)^{-1} \boldsymbol{G}_{J^*}^\top \left(\boldsymbol{\Sigma} + \boldsymbol{\Sigma}_{noise}\right) \boldsymbol{G}_{J^*} \left(\boldsymbol{G}_{J^*}^\top \boldsymbol{G}_{J^*}\right)^{-1}\right)} \\
&\leq \sqrt{8/3}\|\boldsymbol{\Sigma} + \boldsymbol{\Sigma}_{noise}\|_2^{1/2} \rho_{\min}^{-1/2}\left(\boldsymbol{G}_{J^*}^\top \boldsymbol{G}_{J^*}\right)\sqrt{s_*}.
\end{aligned}$$

Then Theorem 11 implies that with high probability

$$\left\|\widehat{\boldsymbol{\Sigma}}_{\hat{J}} - \boldsymbol{\Sigma}_{J^*}\right\|_2 \leq \rho_{\max}\left(\boldsymbol{G}_{J^*}^\top \boldsymbol{G}_{J^*}\right) \tilde{\tau}_{N,s_*,1} \delta \left(\log(M)\right)^{\frac{2+\alpha}{\alpha}},$$

where $\tilde{\tau}_{N,s_*,1} = \max(\tilde{A}^2_{N,s_*,1}, \tilde{B}_{N,s_*,1})$, with

$$\tilde{A}_{N,s_*,1} = \sqrt{8/3}\|\Sigma + \Sigma_{noise}\|_2^{1/2} \rho_{\min}^{-1/2}\left(G_{J^*}^\top G_{J^*}\right)\sqrt{\log d^*}(\log N)^{1/\alpha}\sqrt{\frac{s_*}{N}}$$

and

$$\tilde{B}_{N,s_*,1} = \frac{\tilde{\rho}^2(\Sigma,\Sigma_{noise})\rho_{\min}^{-1}\left(G_{J^*}^\top G_{J^*}\right)}{\sqrt{N}} + \left(\|\Psi_{J^*}\|_2 + \rho_{\min}^{-1}\left(G_{J^*}^\top G_{J^*}\right)\|\Sigma_{noise}\|_2\right)^{1/2}\tilde{A}_{N,s_*,1}.$$

Therefore, in the Gaussian case (but also under other assumptions for X such as those in Proposition 5) the above equations show that the operator norm $\left\|\widehat{\Sigma}_{\hat{J}} - \Sigma_{J^*}\right\|_2^2$ depends on the ratio $\frac{s_*}{N}$. Recall that $\|S-\Sigma\|_2^2$ depends on the ratio $\frac{n}{N}$. Thus, using $\widehat{\Sigma}_{\hat{J}}$ clearly yields significant improvements if s_* is small compared to n.

To summarize our results let us finally consider the case of an orthogonal design. Combining Theorems 8, 10 and 11 one arrives at the following corrolary:

Corollary 12 *Suppose that the observations are i.i.d random variables from model (19). Suppose that $M = n$ and that $G^\top G = I_n$ (orthogonal design) and that X^0 satisfies Assumption 2. Let $\varepsilon > 0$ and $1 \le s_* \le \min(n,M)$. Consider the group Lasso estimator $\widehat{\Sigma}_\lambda$ defined by (5) with the choices*

$$\gamma_k = 2, k = 1,\ldots,n \text{ and } \lambda = \|\Sigma_{noise}\|_2\left(1+\sqrt{\frac{n}{N}}+\sqrt{\frac{2\delta\log M}{N}}\right)^2 \text{ for some constant } \delta > 1.$$

Suppose that $\min_{k\in J^*}\left\|\Psi_k^*\right\|_{\ell_2} > 2n^{1/2}\tilde{C}_1(\sigma,n,s_*,N,\delta)$, where

$$\tilde{C}_1(\sigma,n,s,N,\delta) = \frac{4(1+\varepsilon)\sqrt{s_*}}{\varepsilon}\sqrt{\tilde{C}_0(\sigma,n,s_*,N,\delta)}$$

and

$$\tilde{C}_0(\sigma,n,s_*,N,\delta) = (1+\varepsilon)\left(\frac{8}{n}\left\|S - G\Psi^* G^\top\right\|_F^2 + C(\varepsilon)\|\Sigma_{noise}\|_2^2\left(1+\sqrt{\frac{n}{N}}+\sqrt{\frac{2\delta\log M}{N}}\right)^4\frac{s_*}{n}\right).$$

Take $\hat{J} := \left\{k : \left\|\widehat{\Psi}_k\right\|_{\ell_2} > n^{1/2}\tilde{C}_1(\sigma,n,s,N,\delta)\right\}$. Let $Y = G_{J^*}^\top \widetilde{X}$ and $\tilde{Z} = \|Y\|_{\ell_2}$. Then, with probability at least $1 - M^{1-\delta} - M^{-\left(\frac{\delta_\star}{\delta_*}\right)^{\frac{\alpha}{2+\alpha}}}$, with $\delta > 1$ and $\delta_\star > \delta_*$ one has that

$$\left\|\widehat{\Sigma}_{\hat{J}} - \Sigma_{J^*}\right\|_2 \le \tilde{\tau}_{N,s_*}\delta_\star(\log(M))^{\frac{2+\alpha}{\alpha}},$$

where $\tilde{\tau}_{N,s_*} = \max(\tilde{A}^2_{N,s_*}, \tilde{B}_{N,s_*})$, with $\tilde{A}_{N,s_*} = \|\tilde{Z}\|_{\psi_\alpha}\frac{\sqrt{\log d^*}(\log N)^{1/\alpha}}{\sqrt{N}}$ and $\tilde{B}_{N,s_*} = \frac{\tilde{\rho}^2(\Sigma,\Sigma_{noise})}{\sqrt{N}} + \left(\|\Psi_{J^*}\|_2 + \|\Sigma_{noise}\|_2\right)^{1/2}\tilde{A}_{N,s_*}$.

3.4 Comparison with the Standard Lasso

In this work, we chose a Group Lasso estimation procedure rather than a standard Lasso. As a matter of fact, for covariance estimation in our setting, the group structure enables to impose a constraint on the number of non zero columns of the matrix Ψ and not on the single entries of the matrix Ψ. This corresponds to the natural assumption of obtaining a sparse representation of the process $X(t)$ in the basis given by the functions g_m's and replacing its dimension by its sparsity. Alternatively, the standard Lasso in our setting would be the estimator defined by

$$\widehat{\boldsymbol{\Psi}}_L = \underset{\boldsymbol{\Psi}\in\mathcal{S}_M}{\operatorname{argmin}} \left\{ \left\| \widetilde{\boldsymbol{S}} - \boldsymbol{G}\boldsymbol{\Psi}\boldsymbol{G}^\top \right\|_F^2 + 2\lambda \sum_{k=1}^{M}\sum_{m=1}^{M} \gamma_{mk}|\Psi_{mk}| \right\},$$

where $\lambda \geq 0$ is a regularization parameters and the γ_{mk}'s are positive weights. This procedure leads to the following Lasso estimator of the covariance matrix $\boldsymbol{\Sigma}$

$$\widehat{\boldsymbol{\Sigma}}_L = \boldsymbol{G}\widehat{\boldsymbol{\Psi}}_L\boldsymbol{G}^\top \in \mathbb{R}^{n\times n}.$$

In the orthogonal case (that is $M = n$ and $\boldsymbol{G}^\top\boldsymbol{G} = \boldsymbol{I}_n$), this gives rise to the estimator $\widehat{\boldsymbol{\Psi}}_L$ obtained by soft thresholding individually each entry Y_{mk} of the matrix $\boldsymbol{Y} = \boldsymbol{G}^\top\widetilde{\boldsymbol{S}}\boldsymbol{G}$ with the thresholds $\lambda\gamma_{mk}$. Proposition 13 (see below) allows a simple comparison of the statistical performances of the group Lasso estimator $\widehat{\boldsymbol{\Sigma}}_\lambda$ with those of the standard Lasso estimator $\widehat{\boldsymbol{\Sigma}}_L$ in terms of upper bounds for the Frobenius norm. To simplify the discussion, we only consider the orthogonal case and the simple model

$$\widetilde{X}(t_j) = X^0(t_j) + \mathcal{E}(t_j),\ j = 1,\dots,n, \tag{25}$$

where the process X^0 is defined in (15). The statement of the result for the group Lasso is an immediate consequence of Theorem 8, while the proof to obtain the upper bound for the standard Lasso is an immediate adaptation of the arguments in the proof of Theorem 8.

Proposition 13 *Assume that X satisfies model* (25) *and that the covariance matrix $\boldsymbol{\Sigma}_{noise} = \mathbb{E}(\boldsymbol{W}_1)$ of the noise is positive-definite. Consider the group Lasso estimator $\widehat{\boldsymbol{\Sigma}}_\lambda$ and the standard Lasso estimator $\widehat{\boldsymbol{\Sigma}}_L$ with the choices*

$$\gamma_k = 2,\ \gamma_{mk} = 2,\ \lambda = \|\boldsymbol{\Sigma}_{noise}\|_2 \left(2 + \sqrt{\frac{2\delta\log M}{N}}\right)^2 \text{ for some constant } \delta > 1.$$

Then, there exist two positive constants C_1, C_2 not depending on n, N, s_* such that with probability at least $1 - M^{1-\delta}$ one has that

$$\frac{1}{n}\left\|\widehat{\boldsymbol{\Sigma}}_\lambda - \boldsymbol{\Sigma}\right\|_F^2 \leq \frac{C_1}{n}\|\boldsymbol{S} - \boldsymbol{\Sigma}\|_F^2 + C_2\|\boldsymbol{\Sigma}_{noise}\|_2^2 \left(2 + \sqrt{\frac{2\delta\log n}{N}}\right)^4 \frac{s_*}{n},$$

and

$$\frac{1}{n}\left\|\widehat{\boldsymbol{\Sigma}}_L - \boldsymbol{\Sigma}\right\|_F^2 \leq \frac{C_1}{n}\|\boldsymbol{S} - \boldsymbol{\Sigma}\|_F^2 + C_2\|\boldsymbol{\Sigma}_{noise}\|_2^2 \left(2 + \sqrt{\frac{2\delta\log n}{N}}\right)^4 \frac{s_*^2}{n}.$$

Proposition 13 illustrates the advantages of the Group Lasso over the standard Lasso. Indeed, the second term in the upper bound for the group Lasso is much smaller (of the order $\frac{s_*}{n}$) than the second term in the upper bound for the standard Lasso (of the order $\frac{s_*^2}{n}$). This comes from the fact that the sparsity prior of the Group Lasso is on the number of vanishing columns of the matrix Ψ, while the sparsity prior of the standard Lasso only controls the number of non-zero entries of Ψ. However, to really demonstrate the benefits of our method when compared to the performances of the standard Lasso, it is required to also derive lower bounds. This issue is a difficult task which has been considered in few papers and that is beyond the scope of this paper. For recent work in this direction, we refer to Huang and Zhang (2010) for regression models or Lounici et al. (2011) and Lounici et al. (2009) for linear regression and multi-task learning.

However, the analysis in Huang and Zhang (2010); Lounici et al. (2011) of Group Lasso regularization is carried out the setting of multiple regression models where the parameters to estimate are vectors and with error terms that are centered. Therefore, the results in Huang and Zhang (2010); Lounici et al. (2011) cannot be applied to the matrix regression model (4) since, in our setting, the parameter to estimate is the matrix $\mathbf{\Sigma}$ and the error terms $\boldsymbol{U}_i + \boldsymbol{W}_i$ in (4) are not centered.

4. Numerical Experiments and an Application to Sparse PCA

In this section we present some simulated examples to illustrate the practical behaviour of the covariance matrix estimator by group Lasso regularization proposed in this paper. In particular, we show its performances with an application to sparse Principal Components Analysis (PCA). In the numerical experiments, we use the explicit estimator described in Proposition 1 in the case $M = n$ and an orthogonal design matrix $\boldsymbol{G}$, and also the estimator proposed in the more general situation when $n < M$. The programs for our simulations were implemented using the MATLAB programming environment.

4.1 Description of the Estimating Procedure and the Data

We consider a noisy stochastic processes $\widetilde{X}$ on $\mathbb{T} = [0,1]$ with values in $\mathbb{R}$ observed at fixed location points $t_1, ..., t_n$ in $[0,1]$, generated according to

$$\widetilde{X}(t_j) = X^0(t_j) + \sigma\varepsilon_j,\ j = 1, \ldots, n, \tag{26}$$

where $\sigma > 0$ is the level of noise, $\varepsilon_1, \ldots, \varepsilon_n$ are i.i.d. standard Gaussian variables, and X^0 is a random process independent of the ε_j's. For the process X^0 we consider two simple models. The first one is given by

$$X^0(t) = af(t), \tag{27}$$

where a is a Gaussian random coefficient such that $\mathbb{E}a = 0$, $\mathbb{E}a^2 = \gamma^2$, and $f : [0,1] \to \mathbb{R}$ is an unknown function. The second model for X^0 is

$$X^0(t) = a_1 f_1(t) + a_2 f_2(t), \tag{28}$$

where a_1 and a_2 are independent Gaussian variables such that $\mathbb{E}a_1 = \mathbb{E}a_2 = 0, \mathbb{E}a_1^2 = \gamma_1^2, \mathbb{E}a_2^2 = \gamma_2^2$ (with $\gamma_1 > \gamma_2$), and $f_1, f_2 : [0,1] \to \mathbb{R}$ are unknown functions. The simulated data consists in a sample of N independent observations of the process $\widetilde{X}$ at the points $t_1, ..., t_n$, which are generated according to (26). Therefore, throughout the numerical experiments, one has that

$$\mathbf{\Sigma}_{noise} = \sigma^2 \boldsymbol{I}_n.$$

In model (27), the covariance matrix $\mathbf{\Sigma}$ of the process X^0 at the locations points is given by $\mathbf{\Sigma} = \gamma^2 \boldsymbol{F}\boldsymbol{F}^\top$, where by definition $\boldsymbol{F} = (f(t_1), ..., f(t_1))^\top \in \mathbb{R}^n$. Note that the largest eigenvalue of $\mathbf{\Sigma}$ is $\gamma^2 \|\boldsymbol{F}\|_{\ell_2}^2$ with corresponding eigenvector $\boldsymbol{F}$. We suppose that the signal f has some sparse representation in a large dictionary of basis functions of size M, given by $\{g_m, m = 1, \ldots, M\}$, meaning that $f(t) = \sum_{m=1}^M \beta_m g_m(t)$, with $J^* = \{m, \beta_m \neq 0\}$ of small cardinality s_*. Then, the process X^0 can be written as $X^0(t) = \sum_{m=1}^M a\beta_m g_m(t)$, and thus $\mathbf{\Sigma} = \gamma^2 \boldsymbol{G}\mathbf{\Psi}_{J^*}\boldsymbol{G}^\top$, where $\mathbf{\Psi}_{J^*}$ is an $M \times M$ matrix with entries equal to $\beta_m \beta_{m'}$ for $1 \leq m, m' \leq M$.

Similarly, in model (28), the covariance matrix $\mathbf{\Sigma}$ of the process X^0 at the locations points is given by $\mathbf{\Sigma} = \gamma_1^2 \boldsymbol{F}_1\boldsymbol{F}_1^\top + \gamma_2^2 \boldsymbol{F}_2\boldsymbol{F}_2^\top$, where by definition

$$\boldsymbol{F}_1 = (f_1(t_1), ..., f(t_1))^\top \in \mathbb{R}^n \text{ and } \boldsymbol{F}_2 = (f_2(t_1), ..., f(t_1))^\top \in \mathbb{R}^n.$$

In the following simulations, the functions f_1 and f_2 are chosen such that $\boldsymbol{F}_1$ and $\boldsymbol{F}_2$ are orthogonal vectors in $\mathbb{R}^n$ with $\|\boldsymbol{F}_1\|_{\ell_2} = 1$ and $\|\boldsymbol{F}_2\|_{\ell_2} = 1$. Under such an assumption and since $\gamma_1 > \gamma_2$, the largest eigenvalue of $\mathbf{\Sigma}$ is γ_1^2 with corresponding eigenvector $\boldsymbol{F}_1$, and the second largest eigenvalue of $\mathbf{\Sigma}$ is γ_2^2 with corresponding eigenvector $\boldsymbol{F}_2$. We suppose that the signals f_1 and f_2 have some sparse representations in a large dictionary of basis functions of size M, given by $f_1(t) = \sum_{m=1}^M \beta_m^1 g_m(t)$, and $f_2(t) = \sum_{m=1}^M \beta_m^2 g_m(t)$. Then, the process X^0 can be written as $X^0(t) = \sum_{m=1}^M (a_1\beta_m^1 + a_2\beta_m^2) g_m(t)$ and thus $\mathbf{\Sigma} = \boldsymbol{G}(\gamma_1^2 \mathbf{\Psi}^1 + \gamma_2^2 \mathbf{\Psi}^2)\boldsymbol{G}^\top$, where $\mathbf{\Psi}^1, \mathbf{\Psi}^2$ are $M \times M$ matrix with entries equal to $\beta_m^1(\beta_m^1)'$ and $\beta_m^2(\beta_m^2)'$ for $1 \leq m, m' \leq M$ respectively.

In models (27) and (28), we aim at estimating either $\boldsymbol{F}$ or $\boldsymbol{F}_1, \boldsymbol{F}_2$ by the eigenvectors corresponding to the largest eigenvalues of the matrix $\widehat{\mathbf{\Sigma}}_f$ defined in (23), in a high-dimensional setting with $n > N$ and by using different type of dictionaries. The idea behind this is that $\widehat{\mathbf{\Sigma}}_{\hat{f}}$ is a consistent estimator of $\mathbf{\Sigma}_{J^*}$ (see its definition in 24) in operator norm. Although the matrices $\mathbf{\Sigma}_{J^*}$ and $\mathbf{\Sigma}$ may have different eigenvectors (depending on the design points and chosen dictionary), the examples below show the eigenvectors of $\widehat{\mathbf{\Sigma}}_{\hat{f}}$ can be used as estimators of the eigenvectors of $\mathbf{\Sigma}$.

The estimator $\widehat{\mathbf{\Sigma}}_{\hat{f}}$ of the covariance matrix $\mathbf{\Sigma}$ is computed as follows. Once the dictionary has been chosen, we compute the covariance group Lasso (CGL) estimator $\widehat{\mathbf{\Sigma}}_{\widehat{\lambda}} = \boldsymbol{G}\widehat{\mathbf{\Psi}}_{\widehat{\lambda}}\boldsymbol{G}^\top$, where $\widehat{\mathbf{\Psi}}_{\widehat{\lambda}}$ is defined in (7). We use a completely data-driven choice for the regularizarion parameter λ, given by $\widehat{\lambda} = \|\widehat{\mathbf{\Sigma}_{noise}}\|_2 \left(1 + \sqrt{\frac{n}{N}} + \sqrt{\frac{2\delta \log M}{N}}\right)^2$, where $\|\widehat{\mathbf{\Sigma}_{noise}}\|_2 = \widehat{\sigma}^2$ is the median absolute deviation (MAD) estimator of σ^2 used in standard wavelet denoising (see for example Antoniadis et al., 2001) and $\delta = 1.1$. Hence, the method to compute $\widehat{\mathbf{\Sigma}}_{\widehat{\lambda}}$ is fully data-driven. Furthermore, we will show in the examples below that replacing λ by $\widehat{\lambda}$ into the penalized criterion yields a very good practical performance of the covariance estimation procedure.

As a final step, one needs to compute the estimator $\widehat{\mathbf{\Sigma}}_{\hat{f}}$ of $\mathbf{\Sigma}$, as in (23). For this, we need to have an idea of the true sparsity s_*, since $\hat{f}$ defined in (20) depends on s_* and also on unknown upper bounds on the level of noise in the matrix regression model (8) . A similar problem arises in the selection of a sparse set of active variables when using regularization by a Lasso penalty in the standard linear model. As an example, recall that in Lounici (2008), a second thresholding step is aso used to estimate the true sparsity pattern. However, the suggested thresholding procedure in Lounici (2008) also depends on a priori unknown quantities (such as the amplitude of the coefficients to estimate). To overcome this drawback in our case, we can define the final covariance group Lasso (FCGL) estimator as the matrix

$$\widehat{\mathbf{\Sigma}}_{\hat{f}} = \boldsymbol{G}_{\hat{f}}\widehat{\mathbf{\Psi}}_{\hat{f}}\boldsymbol{G}_{\hat{f}}^\top, \tag{29}$$

with $\hat{J} = \hat{J}_\varepsilon = \left\{ k : \left\| \widehat{\boldsymbol{\Psi}}_k \right\|_{\ell_2} > \varepsilon \right\}$, where ε is a positive constant. To select an appropriate value of ε, one can plot the cardinality of $\hat{J}_\varepsilon$ as a function of ε, and then use an L-curve criterion to only keep in $\hat{J}$ the indices of the columns of $\widehat{\boldsymbol{\Psi}}_{\hat{\lambda}}$ with a significant value in ℓ_2-norm. This choice for $\hat{J}$ is sufficient for numerical purposes.

In the simulations, to measure the accuracy of the estimation procedure, we also use the empirical average of the Frobenius and operator norm of the estimators $\widehat{\boldsymbol{\Sigma}}_{\hat{\lambda}}$ and $\widehat{\boldsymbol{\Sigma}}_{\hat{J}}$ with respect to the true covariance matrix $\boldsymbol{\Sigma}$ defined by $EAFN = \frac{1}{P} \sum_{p=1}^{P} \left\| \widehat{\boldsymbol{\Sigma}}_{\hat{\lambda}}^p - \boldsymbol{\Sigma} \right\|_F$ and $EAON = \frac{1}{P} \sum_{p=1}^{P} \left\| \widehat{\boldsymbol{\Sigma}}_{\hat{J}}^p - \boldsymbol{\Sigma} \right\|_2$ respectively, over a number P of iterations, where $\widehat{\boldsymbol{\Sigma}}_{\hat{\lambda}}^p$ and $\widehat{\boldsymbol{\Sigma}}_{\hat{J}}^p$ are the CGL and FCGL estimators of $\boldsymbol{\Sigma}$, respectively, obtained at the p-th iteration. We also compute the empirical average of the operator norm of the estimator $\widehat{\boldsymbol{\Sigma}}_{\hat{J}}$ with respect to the matrix $\boldsymbol{\Sigma}_{J^*}$, defined by $EAON^* = \frac{1}{P} \sum_{p=1}^{P} \left\| \widehat{\boldsymbol{\Sigma}}_{\hat{J}}^p - \boldsymbol{\Sigma}_{J^*} \right\|_2$.

4.2 Model (27) - Case of an Orthonormal Design (With $n = M$)

First, the size of the dictionary M as well as the basis functions $\{g_m, m = 1, ..., M\}$ have to be specified. In model (27), we will use for the test function f the signals HeaviSine and Blocks (see for example Antoniadis et al., 2001 for a definition), and the Symmlet 8 and Haar wavelet basis for the HeaviSine and Blocks signals respectively, which are implemented in the Matlab's open-source library WaveLab (see for example Antoniadis et al., 2001 for further references on wavelet methods in nonparametric statistics). Then, we took $n = M$ and the location points $t_1, ..., t_n$ are given by the equidistant grid of points $t_j = \frac{j}{M}$, $j = 1, \ldots, M$ such that the design matrix $\boldsymbol{G}$ (using either the Symmlet 8 or the Haar basis) is orthogonal.

In Figure 1 we display the results obtained for a particular simulated sample of size $N = 25$ according to (26), with $n = M = 256$, $\sigma = 0.015$, $\gamma = 0.5$ and with f being either the function HeaviSine or the function Blocks. It can be observed in Figures 1(a) and 1(b) that, as expected in this high dimensional setting ($N < n$), the empirical eigenvector of $\widetilde{\boldsymbol{S}}$ associated to its largest empirical eigenvalue does not lead to a consistent estimator of $\boldsymbol{F}$.

The CGL estimator $\widehat{\boldsymbol{\Sigma}}_{\hat{\lambda}}$ is computed directly from Proposition 1. In Figures 1(c) and 1(d), we display the eigenvector associated to the largest eigenvalue of $\widehat{\boldsymbol{\Sigma}}_{\hat{\lambda}}$ as an estimator of $\boldsymbol{F}$. Note that this estimator behaves poorly. The estimation considerably improves by taking the FCGL estimator $\widehat{\boldsymbol{\Sigma}}_{\hat{J}}$ defined in (29). Figures 1(e) and 1(f) illustrate the very good performance of the eigenvector associated to the largest eigenvalue of the matrix $\widehat{\boldsymbol{\Sigma}}_{\hat{J}}$ as an estimator of $\boldsymbol{F}$.

It is clear that the estimators $\widehat{\boldsymbol{\Sigma}}_{\hat{\lambda}}$ and $\widehat{\boldsymbol{\Sigma}}_{\hat{J}}$ are random matrices that depend on the observed sample. Tables 1 and 2 show the values of $EAFN$, $EAON$ and $EAON^*$ corresponding to $P = 100$ simulated samples of different sizes N and different values of the level of noise σ. It can be observed that for both signals the empirical averages $EAFN$, $EAON$ and $EAON^*$ behaves similarly, being the values of $EAON$ smaller than its corresponding values of $EAFN$ as expected. Observing each table separately we can remark that, for N fixed, when the level of noise σ increases then the values of $EAFN$, $EAON$ and $EAON^*$ also increase. By simple inspection of the values of $EAFN$, $EAON$ and $EAON^*$ in the same position at Tables 1 and 2 we can check that, for σ fixed, when the number of replicates N increases then the values of $EAFN$, $EAON$ and $EAON^*$ decrease in all cases. We

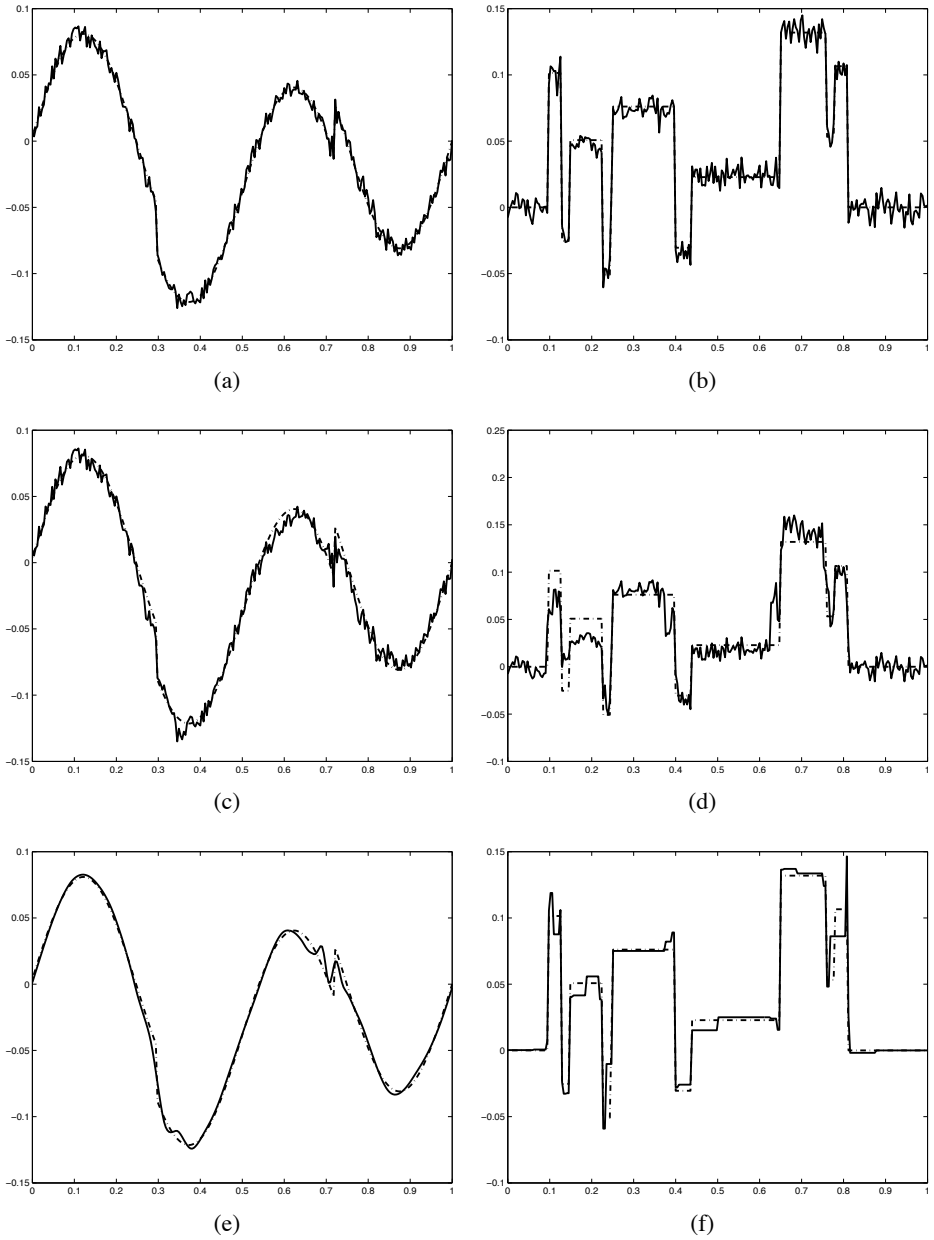


Figure 1: Orthonormal case - Model (27). Signal HeaviSine - (a) Eigenvector associated to the largest eigenvalue of $\widetilde{\boldsymbol{S}}$, (c) Eigenvector associated to the largest eigenvalue of $\widehat{\boldsymbol{\Sigma}}_{\widehat{\lambda}}$, (e) Eigenvector associated to the largest eigenvalue of $\widehat{\boldsymbol{\Sigma}}_{\widehat{f}}$. Signal Blocks - (b) Eigenvector associated to the largest eigenvalue of $\widetilde{\boldsymbol{S}}$, (d) Eigenvector associated to the largest eigenvalue of $\widehat{\boldsymbol{\Sigma}}_{\widehat{\lambda}}$, (f) Eigenvector associated to the largest eigenvalue of $\widehat{\boldsymbol{\Sigma}}_{\widehat{f}}$.

Signal	σ	0.005	0.01	0.05	0.1	0.5	1
HeaviSine	*EAFN*	0.0634	0.0634	0.2199	0.2500	0.2500	0.2500
HeaviSine	*EAON*	0.0619	0.0569	0.1932	0.2500	0.2500	0.2500
HeaviSine	*EAON**	0.0619	0.0569	0.1943	0.2600	0.5000	1.2500
Blocks	*EAFN*	0.0553	0.0681	0.2247	0.2500	0.2500	0.2500
Blocks	*EAON*	0.0531	0.0541	0.2083	0.2500	0.2500	0.2500
Blocks	*EAON**	0.0531	0.0541	0.2107	0.2600	0.5000	1.2500

Table 1: Values of *EAFN*, *EAON* and *EAON** corresponding to signals HeaviSine and Blocks for $M = n = 256, N = 25$.

Signal	σ	0.005	0.01	0.05	0.1	0.5	1
HeaviSine	*EAFN*	0.0501	0.0524	0.1849	0.2499	0.2500	0.2500
HeaviSine	*EAON*	0.0496	0.0480	0.1354	0.2496	0.2500	0.2500
HeaviSine	*EAON**	0.0496	0.0480	0.1366	0.2596	0.5000	1.2500
Blocks	*EAFN*	0.0485	0.0494	0.2014	0.2500	0.2500	0.2500
Blocks	*EAON*	0.0483	0.0429	0.1871	0.2500	0.2500	0.2500
Blocks	*EAON**	0.0483	0.0429	0.1893	0.2600	0.5000	1.2500

Table 2: Values of *EAFN*, *EAON* and *EAON** corresponding to signals HeaviSine and Blocks for $M = n = 256, N = 40$.

can also observe how the difference between *EAON* and *EAON** is bigger as the level of noise increases.

4.3 Model (28) - The Case $M = 2n$ by Mixing Two Orthonormal Basis

Consider now the setting of model (28) with $\gamma_1 = 0.5$, $\gamma_2 = 0.2$, $\sigma = 0.045$, $N = 25$ and an equidistant grid of design points $t_1, ..., t_n$ given by $t_j = \frac{j}{n}$, $j = 1, \ldots, n$ with $n = 128$. For the signals f_1 and f_2 we took the test functions displayed in Figure 2(a) and 2(b). Obviously, the signal f_1 has a sparse representation in a Haar basis while the signal f_2 has a sparse representation in a Fourier basis. Thus, this suggests to construct a dictionary by mixing two orthonormal basis. More precisely, we construct a $n \times n$ orthogonal matrix $\boldsymbol{G}^1$ using the Haar basis and a $n \times n$ orthogonal matrix $\boldsymbol{G}^2$ using a Fourier basis (cosine and sine at various frequencies) at the design points. Then, we form the $n \times M$ design matrix $\boldsymbol{G} = [\boldsymbol{G}^1\ \boldsymbol{G}^2]$ with $M = 2n$. The CGL estimator $\widehat{\boldsymbol{\Sigma}}_{\widehat{\lambda}}$ is computed by the minimization procedure (7) using the Matlab package *minConf* of Schmidt et al. (2008).

In Figures 2(c) and 2(d), we display the eigenvector associated to the largest eigenvalue of $\widehat{\boldsymbol{\Sigma}}_{\widehat{\lambda}}$ as an estimator of $\boldsymbol{F}_1$, and the eigenvector associated to the second largest eigenvalue of $\widehat{\boldsymbol{\Sigma}}_{\widehat{\lambda}}$ as an estimator of $\boldsymbol{F}_2$. Note that these estimators behaves poorly. The estimation considerably improves by taking the FCGL estimator $\widehat{\boldsymbol{\Sigma}}_{\hat{f}}$ defined in (29). Figures 2(e) and 2(f) illustrate the very good performance of the eigenvectors associated to the largest eigenvalue and second largest eigenvalue of the matrix $\widehat{\boldsymbol{\Sigma}}_f$ as estimators of $\boldsymbol{F}_1$ and $\boldsymbol{F}_2$.

Finally, to illustrate the benefits of mixing two orthonormal basis, we also display in Figure 3 and Figure 4 the estimation of $\boldsymbol{F}_1$ and $\boldsymbol{F}_2$ when computing the matrix $\widehat{\boldsymbol{\Sigma}}_f$ by using either only the Haar basis (that is $\boldsymbol{G} = \boldsymbol{G}^1$ and $M = n$) or only the Fourier basis (that is $\boldsymbol{G} = \boldsymbol{G}^1$ and $M = n$). The results are clearly much worse and not satisfactory.

4.4 Model (27) - Case of Non-Equispaced Design Points such that $n < M$

Let us now return to the setting of model (27). The test functions f are either the signal HeaviSine and or the signal Blocks. We also use the Symmlet 8 and Haar wavelet basis for the HeaviSine and Blocks functions respectively. However, we now choose to take a setting where the number of design points n is smaller than the size M of the dictionary. Taking $n < M$, the location points are given by a subset $\{t_1,...,t_n\} \subset \{\frac{k}{M} : k = 1,...,M\}$ of size n, such that the design matrix $\boldsymbol{G}$ is an $n \times M$ matrix (using either the Symmlet 8 and Haar basis). For a fixed value of n, the subset $\{t_1,...,t_n\}$ is chosen by taking the first n points obtained from a random permutation of the elements of the set $\{\frac{1}{M}, \frac{2}{M}, ..., 1\}$. In Figure 5 we present the results obtained for a particular simulated sample of size $N = 25$ according to (26), with $n = 90$, $M = 128$, $\sigma = 0.02$, $\gamma = 0.5$ and with f being either the function HeaviSine or the function Blocks. It can be observed in Figures 5(a) and 5(c) that, as expected in this high dimensional setting ($N < n$), the empirical eigenvector of $\widetilde{\boldsymbol{S}}$ associated to its largest empirical eigenvalue are noisy versions of $\boldsymbol{F}$. As explained previously, the CGL estimator $\widehat{\boldsymbol{\Sigma}}_{\widehat{\lambda}}$ is computed by the minimization procedure (7) using the Matlab package *minConf* of Schmidt et al. (2008). In Figures 5(c) and 5(d) is shown the eigenvector associated to the largest eigenvalue of $\widehat{\boldsymbol{\Sigma}}_{\widehat{\lambda}}$ as an estimator of $\boldsymbol{F}$. Note that this estimator is quite noisy. Again, the eigenvector associated to the largest eigenvalue of the matrix $\widehat{\boldsymbol{\Sigma}}_{\widehat{f}}$ defined in (29) is much a better estimator of $\boldsymbol{F}$. This is illustrated in Figures 5(e) and 5(f). To compare the accuracy of the estimators for different simulated samples, we compute the values of $EAFN$, $EAON$ and $EAON^*$ with fixed values of $\sigma = 0.05$, $M = 128$, $N = 40$, $P = 50$ for different values of the number of design points n. For all the values of n considered, the design points $t_1,...,t_n$ are selected as the first n points obtained from the same random permutation of the elements of the set $\{\frac{1}{M}, \frac{2}{M}, ..., 1\}$. The chosen subset $\{t_1,...,t_n\}$ is used for all the P iterations needed in the computation of the empirical averages (fixed design over the iterations). Figure 6 shows the values of $EAFN$, $EAON$ and $EAON^*$ obtained for each value of n for both signals HeaviSine and Blocks. It can be observed that the values of the empirical averages $EAON$ and $EAON^*$ are much smaller than its corresponding values of $EAFN$ as expected. We can remark that, when n increases, the values of $EAFN$, $EAON$ and $EAON^*$ first increase and then decrease, and the change of monotony occurs when $n > N$. Note that the case $n = M = 128$ is included in these results.

Acknowledgments

This work was supported in part by Egide, under the Program of Eiffel excellency Phd grants, as well as by the BDI CNRS grant. J. Bigot would like to thank the Center for Mathematical Modeling

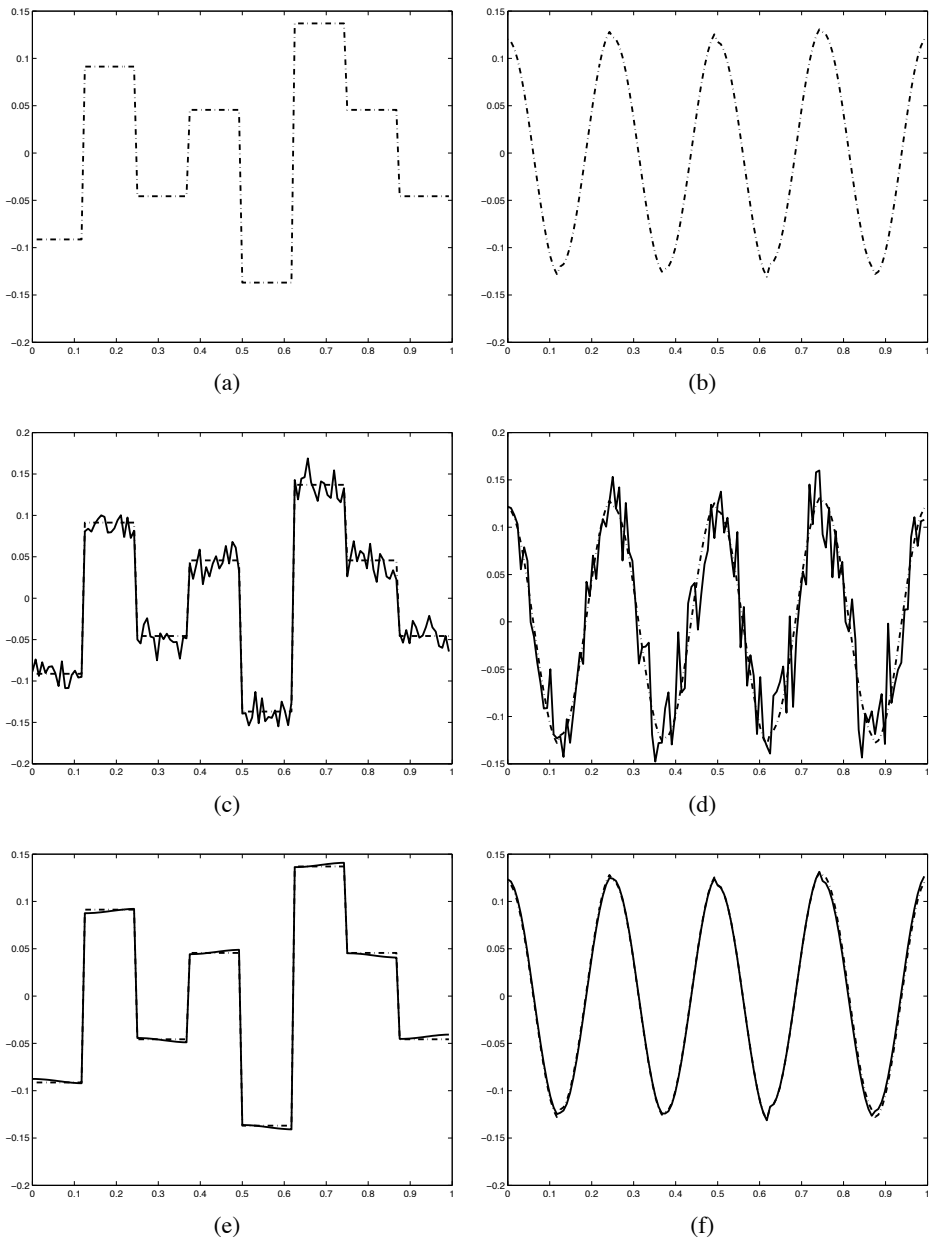
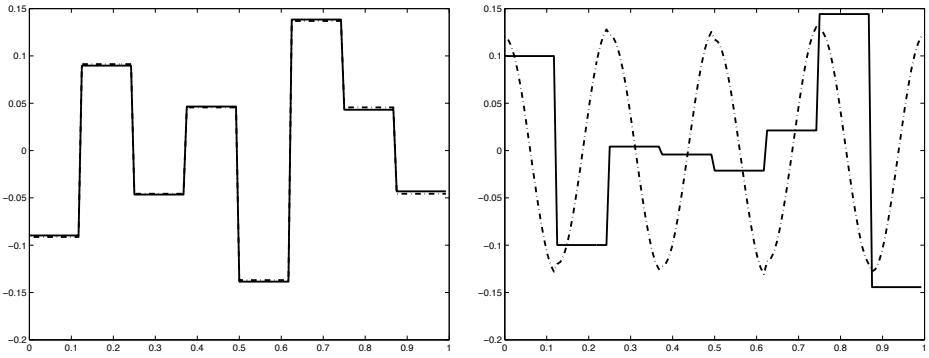
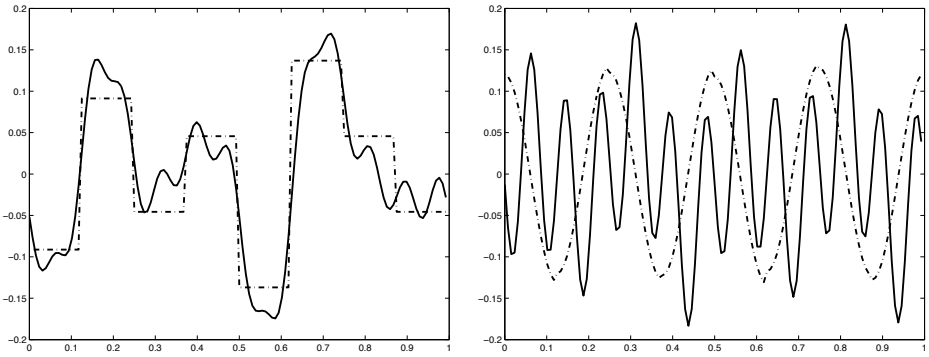


Figure 2: Case $M = 2n$ (Haar + Fourier basis). (a) Signal F_1, (c) Signal F_1 and eigenvector associated to the largest eigenvalue of $\widehat{\mathbf{\Sigma}}_{\widehat{\lambda}}$, (e) Signal F_1 and eigenvector associated to the largest eigenvalue of $\widehat{\mathbf{\Sigma}}_{\widehat{f}}$ with $\boldsymbol{G} = [\boldsymbol{G}^1\ \boldsymbol{G}^2]$. (b) Signal F_2, (d) Signal F_2 and eigenvector associated to the second largest eigenvalue of $\widehat{\mathbf{\Sigma}}_{\widehat{\lambda}}$, (f) Signal F_2 and eigenvector associated to the second largest eigenvalue of $\widehat{\mathbf{\Sigma}}_{\widehat{f}}$ with $\boldsymbol{G} = [\boldsymbol{G}^1\ \boldsymbol{G}^2]$.



(a) (b)

Figure 3: Orthonormal case $M = n$ (Haar). (a) Signal F_1 and Eigenvector associated to the largest eigenvalue of $\widehat{\mathbf{\Sigma}}_{\hat{f}}$ with $\boldsymbol{G} = \boldsymbol{G}^1$, (b) Signal F_2 and Eigenvector associated to the second largest eigenvalue of $\widehat{\mathbf{\Sigma}}_{\hat{f}}$ with $\boldsymbol{G} = \boldsymbol{G}^1$.



(a) (b)

Figure 4: Orthonormal case $M = n$ (Fourier). (a) Signal F_1 and Eigenvector associated to the largest eigenvalue of $\widehat{\mathbf{\Sigma}}_{\hat{f}}$ with $\boldsymbol{G} = \boldsymbol{G}^2$, (b) Signal F_2 and Eigenvector associated to the second largest eigenvalue of $\widehat{\mathbf{\Sigma}}_{\hat{f}}$ with $\boldsymbol{G} = \boldsymbol{G}^2$.

and the CNRS for financial support and excellent hospitality while visiting Santiago where part of this work was carried out.

Appendix A.

This appendix contains the proof of the main results of the paper.

A.1 Notations

First let us introduce some notations and properties that will be used throughout this Appendix. The vectorization of a $p \times q$ matrix $\boldsymbol{A} = (a_{ij})_{1 \leq i \leq p, 1 \leq j \leq q}$ is the $pq \times 1$ column vector

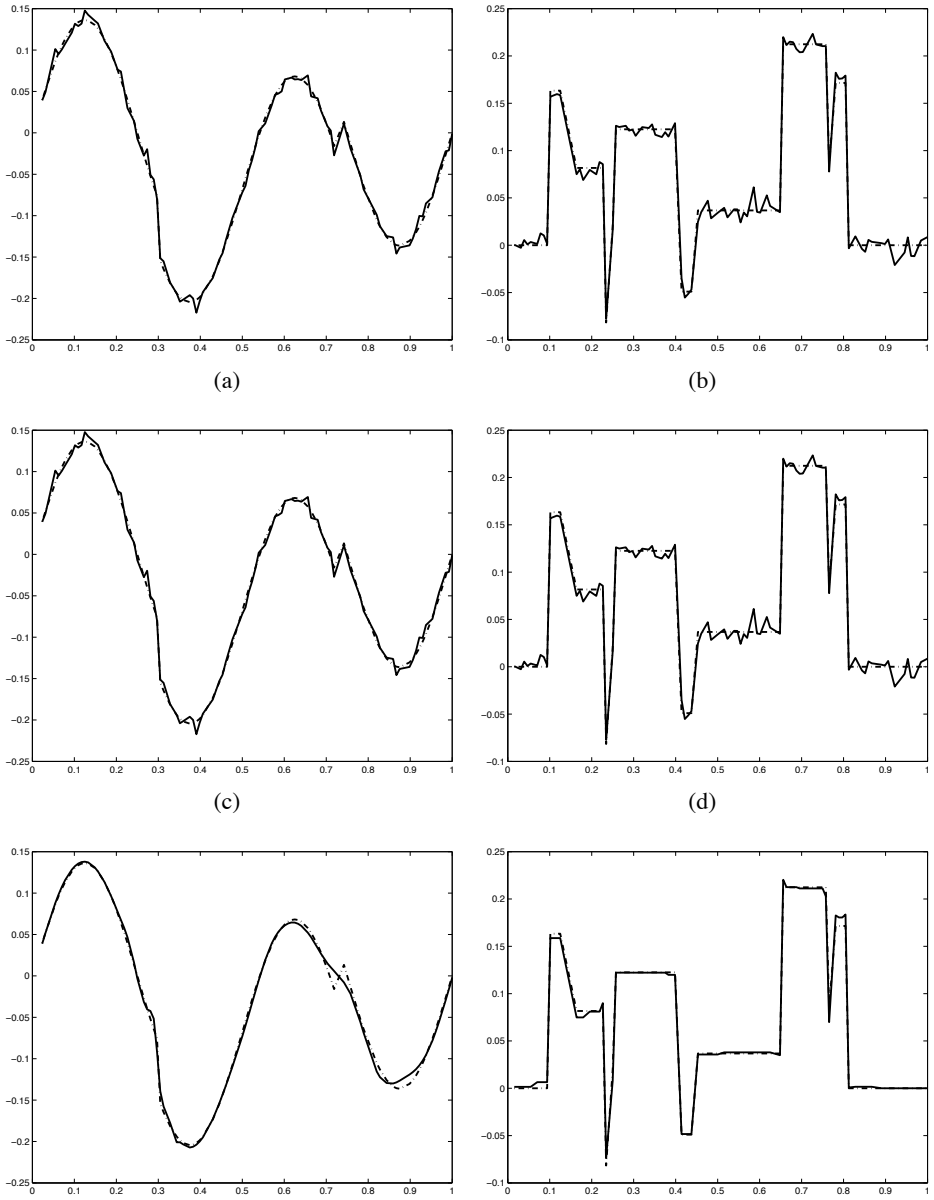
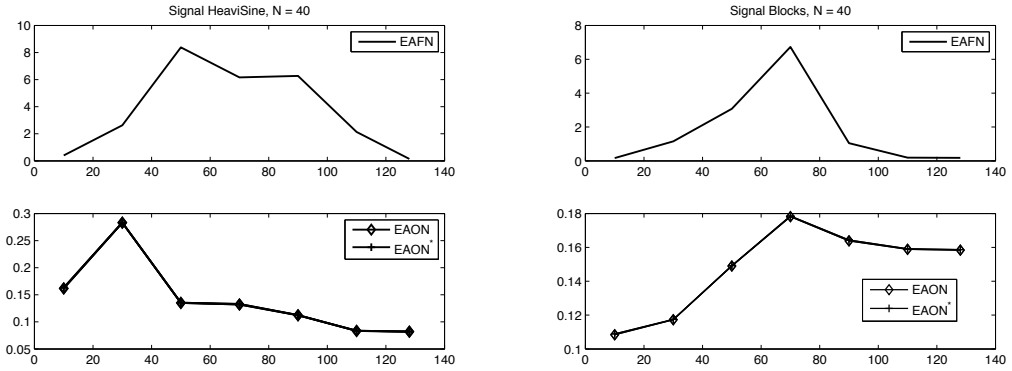


Figure 5: Non equi-spaced points with $n < M$. Signal HeaviSine - (a) Eigenvector associated to the largest eigenvalue of $\widetilde{\boldsymbol{S}}$, (c) Eigenvector associated to the largest eigenvalue of $\widehat{\boldsymbol{\Sigma}}_{\widehat{\lambda}}$, (e) Eigenvector associated to the largest eigenvalue of $\widehat{\boldsymbol{\Sigma}}_{\widehat{f}}$. Signal Blocks - (b) Eigenvector associated to the largest eigenvalue of $\widetilde{\boldsymbol{S}}$, (d) Eigenvector associated to the largest eigenvalue of $\widehat{\boldsymbol{\Sigma}}_{\widehat{\lambda}}$, (f) Eigenvector associated to the largest eigenvalue of $\widehat{\boldsymbol{\Sigma}}_{\widehat{f}}$.



(a) (b)

Figure 6: (a) Values of $EAFN$, $EAON$ and $EAON^*$ for Signal HeaviSine as a function of n, (b) Values of $EAFN$, $EAON$ and $EAON^*$ for Signal Blocks as a function of n.

denoted by $vec(\boldsymbol{A})$, obtain by stacking the columns of the matrix $\boldsymbol{A}$ on top of one another. That is $vec(\boldsymbol{A}) = [a_{11},...,a_{p1},a_{12},...,a_{p2},...,a_{1q},...,a_{pq}]^\top$. If $\boldsymbol{A} = (a_{ij})_{1\leq i\leq k, 1\leq j\leq n}$ is a $k\times n$ matrix and $\boldsymbol{B} = (b_{ij})_{1\leq i\leq p, 1\leq j\leq q}$ is a $p\times q$ matrix, then the Kronecker product of the two matrices, denoted by $\boldsymbol{A}\otimes\boldsymbol{B}$, is the $kp\times nq$ block matrix

$$\boldsymbol{A}\otimes\boldsymbol{B} = \begin{bmatrix} a_{11}\boldsymbol{B} & . & . & . & a_{1n}\boldsymbol{B} \\ . & . & & & . \\ . & & . & & . \\ . & & & . & . \\ a_{k1}\boldsymbol{B} & . & . & . & a_{kn}\boldsymbol{B} \end{bmatrix}.$$

In what follows, we repeatedly use the fact that the Frobenius norm is invariant by the *vec* operation meaning that

$$\|\boldsymbol{A}\|_F^2 = \|vec(\boldsymbol{A})\|_{\ell_2}^2, \tag{30}$$

and the properties that

$$vec(\boldsymbol{ABC}) = \left(\boldsymbol{C}^\top\otimes\boldsymbol{A}\right) vec(\boldsymbol{B}), \tag{31}$$

and

$$(\boldsymbol{A}\otimes\boldsymbol{B})(\boldsymbol{C}\otimes\boldsymbol{D}) = \boldsymbol{AC}\otimes\boldsymbol{BD}, \tag{32}$$

provided the above matrix products are compatible.

A.2 Proof of Proposition 1

Lemma 14 *Let $\widehat{\boldsymbol{\Psi}} = \widehat{\boldsymbol{\Psi}}_\lambda$ denotes the solution of* (7)*. Then, for $k = 1,\ldots,M$*

$$\begin{aligned} \left[(\boldsymbol{G}\otimes\boldsymbol{G})^\top\left(vec(\widetilde{\boldsymbol{S}}) - (\boldsymbol{G}\otimes\boldsymbol{G})vec(\widehat{\boldsymbol{\Psi}})\right)\right]^k &= \lambda\gamma_k\frac{\widehat{\boldsymbol{\Psi}}_k}{\|\widehat{\boldsymbol{\Psi}}_k\|_{\ell_2}} \quad \text{if} \quad \boldsymbol{\Psi}_k\neq 0 \\ \left\|\left[(\boldsymbol{G}\otimes\boldsymbol{G})^\top\left(vec(\widetilde{\boldsymbol{S}}) - (\boldsymbol{G}\otimes\boldsymbol{G})vec(\widehat{\boldsymbol{\Psi}})\right)\right]^k\right\|_{\ell_2} &\leq \lambda\gamma_k \quad \text{if} \quad \widehat{\boldsymbol{\Psi}}_k = 0 \end{aligned}$$

where $\widehat{\boldsymbol{\Psi}}_k$ *denotes the k-th column of the matrix* $\widehat{\boldsymbol{\Psi}}$ *and the notation* $[\beta]^k$ *denotes the vector* $(\beta_{k,m})_{m=1,\ldots,M}$ *in* $\mathbb{R}^M$ *for a vector* $\beta = (\beta_{k,m})_{k,m=1,\ldots,M} \in \mathbb{R}^{M^2}$.

Proof of Lemma 14 For $\boldsymbol{\Psi} \in \mathbb{R}^{M\times M}$ define

$$L(\boldsymbol{\Psi}) = \left\| \widetilde{\boldsymbol{S}} - \boldsymbol{G}\boldsymbol{\Psi}\boldsymbol{G}^\top \right\|_F^2 = \left\| vec(\widetilde{\boldsymbol{S}}) - (\boldsymbol{G}\otimes\boldsymbol{G}) vec(\boldsymbol{\Psi}) \right\|_{\ell_2}^2,$$

and remark that $\widehat{\boldsymbol{\Psi}}$ is the solution of the convex optimization problem

$$\widehat{\boldsymbol{\Psi}} = \underset{\boldsymbol{\Psi}\in\mathcal{S}_M}{\operatorname{argmin}} \left\{ L(\boldsymbol{\Psi}) + 2\lambda \sum_{k=1}^M \gamma_k \sqrt{\sum_{m=1}^M \Psi_{mk}^2} \right\}.$$

It follows from standard arguments in convex analysis (see for example Boyd and Vandenberghe, 2004), that $\widehat{\boldsymbol{\Psi}}$ is a solution of the above minimization problem if and only if

$$-\nabla L(\widehat{\boldsymbol{\Psi}}) \in 2\lambda\partial \left(\sum_{k=1}^M \gamma_k \sqrt{\sum_{m=1}^M \hat{\Psi}_{mk}^2} \right)$$

where $\nabla L(\widehat{\boldsymbol{\Psi}})$ denotes the gradient of L at $\widehat{\boldsymbol{\Psi}}$ and ∂ denotes the subdifferential given by

$$\partial \left(\sum_{k=1}^M \gamma_k \sqrt{\sum_{m=1}^M \Psi_{mk}^2} \right) = \left\{ \boldsymbol{\Theta} \in \mathbb{R}^{M\times M} : \boldsymbol{\Theta}_k = \gamma_k \frac{\boldsymbol{\Psi}_k}{\|\boldsymbol{\Psi}_k\|_{\ell_2}} \text{ if } \boldsymbol{\Psi}_k \neq 0, \|\boldsymbol{\Theta}_k\|_{\ell_2} \leq \gamma_k \text{ if } \boldsymbol{\Psi}_k = 0 \right\}$$

where $\boldsymbol{\Theta}_k$ denotes the k-th column of $\boldsymbol{\Theta} \in \mathbb{R}^{M\times M}$ which completes the proof. □

Now, let $\boldsymbol{\Psi} \in \mathcal{S}_M$ with $M = n$ and suppose that $\boldsymbol{G}^\top\boldsymbol{G} = \boldsymbol{I}_n$. Let $\boldsymbol{Y} = (\boldsymbol{Y}_{mk})_{1\leq m,k\leq M} = \boldsymbol{G}^\top\widetilde{\boldsymbol{S}}\boldsymbol{G}$ and remark that $vec(\boldsymbol{Y}) = (\boldsymbol{G}\otimes\boldsymbol{G})^\top vec(\widetilde{\boldsymbol{S}})$. Then, by using Lemma 14 and the fact that $\boldsymbol{G}^\top\boldsymbol{G} = \boldsymbol{I}_n$ implies that $(\boldsymbol{G}\otimes\boldsymbol{G})^\top(\boldsymbol{G}\otimes\boldsymbol{G}) = \boldsymbol{I}_{n^2}$, it follows that $\widehat{\boldsymbol{\Psi}} = \widehat{\boldsymbol{\Psi}}_\lambda$ satisfies for $k = 1,\ldots,M$ the following equations

$$\widehat{\boldsymbol{\Psi}}_k \left(1 + \frac{\lambda\gamma_k}{\sqrt{\sum_{m=1}^M \widehat{\Psi}_{mk}^2}} \right) = \boldsymbol{Y}_k \text{ for all } \widehat{\boldsymbol{\Psi}}_k \neq 0,$$

and

$$\sqrt{\sum_{m=1}^M \boldsymbol{Y}_{mk}^2} \leq \lambda\gamma_k \text{ for all } \widehat{\boldsymbol{\Psi}}_k = 0.$$

where $\widehat{\boldsymbol{\Psi}}_k = (\widehat{\Psi}_{mk})_{1\leq m\leq M} \in \mathbb{R}^M$ and $\boldsymbol{Y}_k = (\boldsymbol{Y}_{mk})_{1\leq m\leq M} \in \mathbb{R}^M$, which implies that the solution is given by

$$\widehat{\Psi}_{mk} = \begin{cases} 0 & \text{if} \quad \sqrt{\sum_{m=1}^M \boldsymbol{Y}_{mk}^2} \leq \lambda\gamma_k \\ Y_{mk}\left(1 - \frac{\lambda\gamma_k}{\sqrt{\sum_{j=1}^M \boldsymbol{Y}_{jk}^2}} \right) & \text{if} \quad \sqrt{\sum_{m=1}^M \boldsymbol{Y}_{mk}^2} > \lambda\gamma_k \end{cases}$$

which completes the proof of Proposition 1. □

A.3 Proof of Proposition 5

First suppose that X is Gaussian. Then, remark that for $Z = \|\boldsymbol{X}\|_{\ell_2}$, one has that $\|Z\|_{\psi_2} < +\infty$ which implies that $\|Z\|_{\psi_2} = \|Z^2\|_{\psi_1}^{1/2}$. Since $Z^2 = \sum_{i=1}^n |X(t_i)|^2$ it follows that

$$\|Z^2\|_{\psi_1} \leq \sum_{i=1}^n \|Z_i^2\|_{\psi_1} = \sum_{i=1}^n \|Z_i\|_{\psi_2}^2 = \sum_{i=1}^n \boldsymbol{\Sigma}_{ii} \|\boldsymbol{\Sigma}_{ii}^{-1/2} Z_i\|_{\psi_2}^2,$$

where $Z_i = X(t_i), i = 1, \ldots, n$ and $\boldsymbol{\Sigma}_{ii}$ denotes the ith diagonal element of $\boldsymbol{\Sigma}$. Then, the result follows by noticing that $\|Y\|_{\psi_2} \leq \sqrt{8/3}$ if $Y \sim N(0,1)$. The proof for the case where X is such that $\|Z\|_{\psi_2} < +\infty$ and there exists a constant C_1 such that $\|\boldsymbol{\Sigma}_{ii}^{-1/2} Z_i\|_{\psi_2} \leq C_1$ for all $i = 1, \ldots, n$ follows from the same arguments.

Now, consider the case where X is a bounded process. Since there exists a constant $R > 0$ such that for all $t \in \mathbb{T}$, $|X(t)| \leq R$, it follows that for $Z = \|\boldsymbol{X}\|_{\ell_2}$ then $Z \leq \sqrt{n}R$ which implies that for any $\alpha \geq 1$, $\|Z\|_{\psi_\alpha} \leq \sqrt{n}R(\log 2)^{-1/\alpha}$, (by definition of the norm $\|Z\|_{\psi_\alpha}$) which completes the proof of Proposition 5. □

A.4 Proof of Proposition 7

Under the assumption that $X = X^0$, it follows that $\boldsymbol{\Sigma} = \boldsymbol{G}\boldsymbol{\Psi}^*\boldsymbol{G}^\top$ with $\boldsymbol{\Psi}^* = \mathbb{E}\left(\boldsymbol{a}\boldsymbol{a}^\top\right)$, where $\boldsymbol{a}$ is the random vector of $\mathbb{R}^M$ with $\boldsymbol{a}_m = a_m$ for $m \in J^*$ and $\boldsymbol{a}_m = 0$ for $m \notin J^*$. Then, define the random vector $\boldsymbol{a}_{J^*} \in \mathbb{R}^{J^*}$ whose coordinates are the random coefficients a_m for $m \in J^*$. Let $\boldsymbol{\Psi}_{J^*} = \mathbb{E}\left(\boldsymbol{a}_{J^*}\boldsymbol{a}_{J^*}^\top\right)$. Note that $\boldsymbol{\Sigma} = \boldsymbol{G}_{J^*}\boldsymbol{\Psi}_{J^*}\boldsymbol{G}_{J^*}^\top$ and $\boldsymbol{S} = \boldsymbol{G}_{J^*}\widehat{\boldsymbol{\Psi}}_{J^*}\boldsymbol{G}_{J^*}^\top$, with $\widehat{\boldsymbol{\Psi}}_{J^*} = \frac{1}{N}\sum_{i=1}^N \boldsymbol{a}_{J^*}^i(\boldsymbol{a}_{J^*}^i)^\top$, where $\boldsymbol{a}_{J^*}^i \in \mathbb{R}^{J^*}$ denotes the random vector whose coordinates are the random coefficients a_m^i for $m \in J^*$ such that $X_i(t) = \sum_{m \in J^*} a_m^i g_m(t)$, $t \in \mathbb{T}$.

Therefore, $\widehat{\boldsymbol{\Psi}}_{J^*}$ is a sample covariance matrix of size $s_* \times s_*$ and we can control its deviation in operator norm from $\widehat{\boldsymbol{\Psi}}_{J^*}$ by using Proposition 6. For this we simply have to verify conditions similar to **(A1)** and **(A2)** in Assumption 2 for the random vector $\boldsymbol{a}_{J^*} = (\boldsymbol{G}_{J^*}^\top\boldsymbol{G}_{J^*})^{-1}\boldsymbol{G}_{J^*}^\top\boldsymbol{X} \in \mathbb{R}^{s_*}$. First, let $\beta \in \mathbb{R}^{s_*}$ with $\|\beta\|_{\ell_2} = 1$. Then, remark that $\boldsymbol{a}_{J^*}^\top\beta = \boldsymbol{X}^\top\tilde{\beta}$ with $\tilde{\beta} = \boldsymbol{G}_{J^*}\left(\boldsymbol{G}_{J^*}^\top\boldsymbol{G}_{J^*}\right)^{-1}\beta$. Since $\|\tilde{\beta}\|_{\ell_2} \leq \left(\rho_{\min}\left(\boldsymbol{G}_{J^*}^\top\boldsymbol{G}_{J^*}\right)\right)^{-1/2}$ and using that X satisfies Assumption 2 it follows that

$$\left(\mathbb{E}|\boldsymbol{a}_{J^*}^\top\beta|^4\right)^{1/4} \leq \rho\left(\boldsymbol{\Sigma}\right)\rho_{\min}^{-1/2}\left(\boldsymbol{G}_{J^*}^\top\boldsymbol{G}_{J^*}\right). \tag{33}$$

Now let $\tilde{Z} = \|\boldsymbol{a}_{J^*}\|_{\ell_2} \leq \rho_{\min}^{-1/2}\left(\boldsymbol{G}_{J^*}^\top\boldsymbol{G}_{J^*}\right)\|\boldsymbol{X}\|_{\ell_2}$. Given our assumptions on X it follows that there exists $\alpha \geq 1$ such that

$$\|\tilde{Z}\|_{\psi_\alpha} \leq \rho_{\min}^{-1/2}\left(\boldsymbol{G}_{J^*}^\top\boldsymbol{G}_{J^*}\right)\|Z\|_{\psi_\alpha} < +\infty, \tag{34}$$

where $Z = \|\boldsymbol{X}\|_{\ell_2}$. Hence, using the relations (33) and (34), and Proposition 6 (with $\boldsymbol{a}_{J^*}$ instead of $\boldsymbol{X}$), it follows that there exists a universal constant $\delta_* > 0$ such that for all $x > 0$,

$$\mathbb{P}\left(\left\|\widehat{\boldsymbol{\Psi}}_{J^*} - \boldsymbol{\Psi}_{J^*}\right\|_2 \geqslant \tilde{\tau}_{d^*,N,s_*,1}x\right) \leqslant \exp\left(-(\delta_*^{-1}x)^{\frac{\alpha}{2+\alpha}}\right),$$

where $\tilde{\tau}_{d^*,N,s_*,1} = \max(\tilde{A}_{d^*,N,s_*,1}^2, \tilde{B}_{d^*,N,s_*,1})$, with $\tilde{A}_{d^*,N,s_*,1} = \|\tilde{Z}\|_{\psi_\alpha}\frac{\sqrt{\log d^*}(\log N)^{1/\alpha}}{\sqrt{N}}$, $\tilde{B}_{d^*,N,s_*,1} = \frac{\rho^2(\boldsymbol{\Sigma})\rho_{\min}^{-1}\left(\boldsymbol{G}_{J^*}^\top\boldsymbol{G}_{J^*}\right)}{\sqrt{N}} + \|\boldsymbol{\Psi}_{J^*}\|_2^{1/2}\tilde{A}_{d^*,N,s_*,1}$ and $d^* = \min(N, s_*)$. Then, using the inequality $\|\boldsymbol{S} - \boldsymbol{\Sigma}\|_2 \leq$

$\rho_{\max}\left(\boldsymbol{G}_{J^*}^\top \boldsymbol{G}_{J^*}\right) \|\widehat{\boldsymbol{\Psi}}_{J^*} - \boldsymbol{\Psi}_{J^*}\|_2$, it follows that

$$\begin{aligned}
& \mathbb{P}\left(\|\boldsymbol{S}-\boldsymbol{\Sigma}\|_2 \geq \rho_{\max}\left(\boldsymbol{G}_{J^*}^\top \boldsymbol{G}_{J^*}\right) \tilde{\tau}_{d^*,N,s_*,1} x\right) \\
\leq\ & \mathbb{P}\left(\rho_{\max}\left(\boldsymbol{G}_{J^*}^\top \boldsymbol{G}_{J^*}\right) \left\|\widehat{\boldsymbol{\Psi}}_{J^*} - \boldsymbol{\Psi}_{J^*}\right\|_2 \geqslant \rho_{\max}\left(\boldsymbol{G}_{J^*}^\top \boldsymbol{G}_{J^*}\right) \tilde{\tau}_{d^*,N,s_*,1} x\right) \\
=\ & \mathbb{P}\left(\left\|\widehat{\boldsymbol{\Psi}}_{J^*} - \boldsymbol{\Psi}_{J^*}\right\|_2 \geqslant \tilde{\tau}_{d^*,N,s_*,1} x\right) \\
\leqslant\ & \exp\left(-(\delta_*^{-1} x)^{\frac{\alpha}{2+\alpha}}\right).
\end{aligned}$$

Hence, the result follows with

$$\begin{aligned}
\tilde{\tau}_{N,s_*} &= \rho_{\max}\left(\boldsymbol{G}_{J^*}^\top \boldsymbol{G}_{J^*}\right) \tilde{\tau}_{d^*,N,s_*,1} \\
&= \max(\rho_{\max}\left(\boldsymbol{G}_{J^*}^\top \boldsymbol{G}_{J^*}\right) \tilde{A}^2_{d^*,N,s_*,1}, \rho_{\max}\left(\boldsymbol{G}_{J^*}^\top \boldsymbol{G}_{J^*}\right) \tilde{B}_{d^*,N,s_*,1}) \\
&= \max(\tilde{A}^2_{d^*,N,s_*}, \tilde{B}_{d^*,N,s_*}),
\end{aligned}$$

where $\tilde{A}_{d^*,N,s_*} = \rho^{1/2}_{\max}\left(\boldsymbol{G}_{J^*}^\top \boldsymbol{G}_{J^*}\right) \|\tilde{Z}\|_{\psi_\alpha} \frac{\sqrt{\log d^*}(\log N)^{1/\alpha}}{\sqrt{N}}$ and, using the inequality

$$\|\boldsymbol{\Psi}_{J^*}\|_2 = \left\|\left(\boldsymbol{G}_{J^*}^\top \boldsymbol{G}_{J^*}\right)^{-1} \boldsymbol{G}_{J^*}^\top \boldsymbol{\Sigma} \boldsymbol{G}_{J^*} \left(\boldsymbol{G}_{J^*}^\top \boldsymbol{G}_{J^*}\right)^{-1}\right\|_2 \leq \rho^{-1}_{\min}\left(\boldsymbol{G}_{J^*}^\top \boldsymbol{G}_{J^*}\right) \|\boldsymbol{\Sigma}\|_2,$$

$$\tilde{B}_{d^*,N,s_*} = \left(\frac{\rho_{\max}\left(\boldsymbol{G}_{J^*}^\top \boldsymbol{G}_{J^*}\right)}{\rho_{\min}\left(\boldsymbol{G}_{J^*}^\top \boldsymbol{G}_{J^*}\right)}\right) \frac{\rho^2(\boldsymbol{\Sigma})}{\sqrt{N}} + \left(\frac{\rho_{\max}\left(\boldsymbol{G}_{J^*}^\top \boldsymbol{G}_{J^*}\right)}{\rho_{\min}\left(\boldsymbol{G}_{J^*}^\top \boldsymbol{G}_{J^*}\right)}\right)^{1/2} \|\boldsymbol{\Sigma}\|_2^{1/2} \tilde{A}_{d^*,N,s_*}.$$

A.5 Proof of Theorem 8

Let us first prove the following lemmas.

Lemma 15 *Let $\mathcal{E}_1, ..., \mathcal{E}_N$ be independent copies of a second order Gaussian process $\mathcal{E}$ with zero mean. Let $\boldsymbol{W} = \frac{1}{N}\sum_{i=1}^{N} \boldsymbol{W}_i$ with*

$$\boldsymbol{W}_i = \mathcal{E}_i \mathcal{E}_i^\top \in \mathbb{R}^{n\times n} \text{ and } \mathcal{E}_i = (\mathcal{E}_i(t_1), ..., \mathcal{E}_i(t_n))^\top,\ i = 1, \ldots, N.$$

Suppose that $\boldsymbol{\Sigma}_{noise} = \mathbb{E}(\boldsymbol{W}_1)$ is positive-definite. For $1 \leq k \leq M$, let $\boldsymbol{\eta}_k$ be the k-th column of the matrix $\boldsymbol{G}^\top \boldsymbol{W} \boldsymbol{G}$. Then, for any $x > 0$,

$$\mathbb{P}\left(\|\boldsymbol{\eta}_k\|_{\ell_2} \geq \|\boldsymbol{G}_k\|_{\ell_2} \sqrt{\rho_{\max}(\boldsymbol{G}\boldsymbol{G}^\top)} \|\boldsymbol{\Sigma}_{noise}\|_2 \left(1 + \sqrt{\frac{n}{N}} + \sqrt{\frac{2x}{N}}\right)^2\right) \leq \exp(-x).$$

Proof of Lemma 15: by definition one has that $\|\boldsymbol{\eta}_k\|^2_{\ell_2} = \boldsymbol{G}_k^\top \boldsymbol{W} \boldsymbol{G} \boldsymbol{G}^\top \boldsymbol{W} \boldsymbol{G}_k$ where $\boldsymbol{G}_k$ denotes the k-th column of $\boldsymbol{G}$. Hence

$$\|\boldsymbol{\eta}_k\|^2_{\ell_2} \leq \|\boldsymbol{G}_k\|^2_{\ell_2} \rho_{\max}(\boldsymbol{G}\boldsymbol{G}^\top) \|\boldsymbol{W}\|_2^2. \tag{35}$$

Using the assumption that $\mathbf{\Sigma}_{noise}$ is positive-definite define the random vectors $Z_i = \mathbf{\Sigma}_{noise}^{-1/2}\mathcal{E}_i, i = 1,\ldots,n$. Note that the Z_i's are i.i.d. Gaussian vectors in $\mathbb{R}^n$ with zero mean and covariance matrix the identity. Then, define the $N \times n$ matrix

$$\Gamma = \frac{1}{\sqrt{N}} \begin{pmatrix} Z_1^\top \\ \vdots \\ Z_N^\top \end{pmatrix}.$$

Since Γ is a matrix with i.i.d. entries following a Gaussian distribution with zero mean and variance $1/N$, it follows from the arguments in the proof of Theorem II.13 in Davidson and Szarek (2001) that for any $x > 0$

$$\mathbb{P}\left(\|\mathbf{\Gamma}^\top \mathbf{\Gamma}\|_2 \geq \left(1 + \sqrt{\frac{n}{N}} + \sqrt{\frac{2x}{N}}\right)^2 \right) \leq \exp(-x). \tag{36}$$

Now, since $\boldsymbol{W} = \mathbf{\Sigma}_{noise}^{1/2}\Gamma^\top\Gamma\mathbf{\Sigma}_{noise}^{1/2}$ it follows that $\|\boldsymbol{W}\|_2 \leq \|\mathbf{\Sigma}_{noise}\|_2\|\Gamma^\top\Gamma\|_2$. Hence, inequality (36) implies that for any $x > 0$

$$\mathbb{P}\left(\|\boldsymbol{W}\|_2 \geq \|\mathbf{\Sigma}_{noise}\|_2 \left(1 + \sqrt{\frac{n}{N}} + \sqrt{\frac{2x}{N}}\right)^2 \right) \leq \exp(-x),$$

and the result finally follows from inequality (35). □

Lemma 16 *Let $1 \leq s \leq \min(n,M)$ and suppose that Assumption 1 holds for some $c_0 > 0$. Let $J \subset \{1,\ldots,M\}$ be a subset of indices of cardinality $|J| \leq s$. Let $\mathbf{\Delta} \in \mathcal{S}_M$ and suppose that*

$$\sum_{k \in J^c} \|\mathbf{\Delta}_k\|_{\ell_2} \leq c_0 \sum_{k \in J} \|\mathbf{\Delta}_k\|_{\ell_2},$$

where $\mathbf{\Delta}_k$ denotes the k-th column of $\mathbf{\Delta}$. Let

$$\kappa_{s,c_0} = \left(\rho_{\min}(s)^2 - c_0\theta(\boldsymbol{G})\rho_{\max}(\boldsymbol{G}^\top\boldsymbol{G})s\right)^{1/2}.$$

Then,

$$\left\|\boldsymbol{G}\mathbf{\Delta}\boldsymbol{G}^\top\right\|_F^2 \geq \kappa_{s,c_0}^2 \left\|\mathbf{\Delta}_J\right\|_F^2,$$

where $\mathbf{\Delta}_J$ denotes the $M \times M$ matrix obtained by setting to zero the rows and columns of $\mathbf{\Delta}$ whose indices are not in J.

Proof of Lemma 16: first let us introduce some notations. For $\mathbf{\Delta} \in \mathcal{S}_M$ and $J \subset \{1,\ldots,M\}$, then $\mathbf{\Delta}_{J^c}$ denotes the $M \times M$ matrix obtained by setting to zero the rows and columns of $\mathbf{\Delta}$ whose indices are not in the complementary J^c of J. Now, remark that

$$\begin{aligned} \left\|\boldsymbol{G}\mathbf{\Delta}\boldsymbol{G}^\top\right\|_F^2 &= \left\|\boldsymbol{G}\mathbf{\Delta}_J\boldsymbol{G}^\top\right\|_F^2 + \left\|\boldsymbol{G}\mathbf{\Delta}_{J^c}\boldsymbol{G}^\top\right\|_F^2 + 2tr\left(\boldsymbol{G}\mathbf{\Delta}_J\boldsymbol{G}^\top\boldsymbol{G}\mathbf{\Delta}_{J^c}\boldsymbol{G}^\top\right) \\ &\geq \left\|\boldsymbol{G}\mathbf{\Delta}_J\boldsymbol{G}^\top\right\|_F^2 + 2tr\left(\boldsymbol{G}\mathbf{\Delta}_J\boldsymbol{G}^\top\boldsymbol{G}\mathbf{\Delta}_{J^c}\boldsymbol{G}^\top\right). \end{aligned} \tag{37}$$

Let $\boldsymbol{A} = \boldsymbol{G}\boldsymbol{\Delta}_J\boldsymbol{G}^\top$ and $\boldsymbol{B} = \boldsymbol{G}\boldsymbol{\Delta}_{J^c}\boldsymbol{G}^\top$. Using that $tr\left(\boldsymbol{A}^\top\boldsymbol{B}\right) = vec(\boldsymbol{A})^\top vec(\boldsymbol{B})$ and the properties (30) and (32) it follows that

$$tr\left(\boldsymbol{G}\boldsymbol{\Delta}_J\boldsymbol{G}^\top\boldsymbol{G}\boldsymbol{\Delta}_{J^c}\boldsymbol{G}^\top\right) = vec(\boldsymbol{\Delta}_J)^\top\left(\boldsymbol{G}^\top\boldsymbol{G}\otimes\boldsymbol{G}^\top\boldsymbol{G}\right)vec(\boldsymbol{\Delta}_{J^c}). \tag{38}$$

Let $\boldsymbol{C} = \boldsymbol{G}^\top\boldsymbol{G}\otimes\boldsymbol{G}^\top\boldsymbol{G}$ and note that $\boldsymbol{C}$ is a $M^2\times M^2$ matrix whose elements can be written in the form of $M\times M$ block matrices given by

$$\boldsymbol{C}_{ij} = (\boldsymbol{G}^\top\boldsymbol{G})_{ij}\boldsymbol{G}^\top\boldsymbol{G}, \text{ for } 1\le i,j\le M.$$

Now, write the $M^2\times 1$ vectors $vec(\boldsymbol{\Delta}_J)$ and $vec(\boldsymbol{\Delta}_{J^c})$ in the form of block vectors as $vec(\boldsymbol{\Delta}_J) = [(\boldsymbol{\Delta}_J)_i^\top]_{1\le i\le M}^\top$ and $vec(\boldsymbol{\Delta}_{J^c}) = [(\boldsymbol{\Delta}_{J^c})_j^\top]_{1\le j\le M}^\top$, where $(\boldsymbol{\Delta}_J)_i\in\mathbb{R}^M$ $(\boldsymbol{\Delta}_{J^c})_j\in\mathbb{R}^M$ for $1\le i,j\le M$. Using (38) it follows that

$$\begin{aligned} tr\left(\boldsymbol{G}\boldsymbol{\Delta}_J\boldsymbol{G}^\top\boldsymbol{G}\boldsymbol{\Delta}_{J^c}\boldsymbol{G}^\top\right) &= \sum_{1\le i,j\le M}(\boldsymbol{\Delta}_J)_i^\top\boldsymbol{C}_{ij}(\boldsymbol{\Delta}_{J^c})_j \\ &= \sum_{i\in J}\sum_{j\in J^c}(\boldsymbol{G}^\top\boldsymbol{G})_{ij}(\boldsymbol{\Delta}_J)_i^\top\boldsymbol{G}^\top\boldsymbol{G}(\boldsymbol{\Delta}_{J^c})_j. \end{aligned}$$

Now, using that $\left|(\boldsymbol{G}^\top\boldsymbol{G})_{ij}\right|\le\theta(\boldsymbol{G})$ for $i\neq j$ and that

$$\left|(\boldsymbol{\Delta}_J)_i^\top\boldsymbol{G}^\top\boldsymbol{G}(\boldsymbol{\Delta}_{J^c})_j\right| \le \|\boldsymbol{G}(\boldsymbol{\Delta}_J)_i\|_{\ell_2}\|\boldsymbol{G}(\boldsymbol{\Delta}_{J^c})_j\|_{\ell_2} \le \rho_{\max}(\boldsymbol{G}^\top\boldsymbol{G})\|(\boldsymbol{\Delta}_J)_i\|_{\ell_2}\|(\boldsymbol{\Delta}_{J^c})_j\|_{\ell_2},$$

it follows that

$$tr\left(\boldsymbol{G}\boldsymbol{\Delta}_J\boldsymbol{G}^\top\boldsymbol{G}\boldsymbol{\Delta}_{J^c}\boldsymbol{G}^\top\right) \ge -\theta(\boldsymbol{G})\rho_{\max}(\boldsymbol{G}^\top\boldsymbol{G})\left(\sum_{i\in J}\|(\boldsymbol{\Delta}_J)_i\|_{\ell_2}\right)\left(\sum_{j\in J^c}\|(\boldsymbol{\Delta}_{J^c})_j\|_{\ell_2}\right).$$

Now, using the assumption that $\sum_{k\in J^c}\|\boldsymbol{\Delta}_k\|_{\ell_2}\le c_0\sum_{k\in J}\|\boldsymbol{\Delta}_k\|_{\ell_2}$ it follows that

$$\begin{aligned} tr\left(\boldsymbol{G}\boldsymbol{\Delta}_J\boldsymbol{G}^\top\boldsymbol{G}\boldsymbol{\Delta}_{J^c}\boldsymbol{G}^\top\right) &\ge -c_0\theta(\boldsymbol{G})\rho_{\max}(\boldsymbol{G}^\top\boldsymbol{G})\left(\sum_{i\in J}\|(\boldsymbol{\Delta}_J)_i\|_{\ell_2}\right)^2 \\ &\ge -c_0\theta(\boldsymbol{G})\rho_{\max}(\boldsymbol{G}^\top\boldsymbol{G})s\|\boldsymbol{\Delta}_J\|_F^2, \end{aligned} \tag{39}$$

where, for the inequality, we have used the properties that for the positive reals $c_i = \|(\boldsymbol{\Delta}_J)_i\|_{\ell_2}$, $i\in J$ then $\left(\sum_{i\in J}c_i\right)^2\le|J|\sum_{i\in J}c_i^2\le s\sum_{i\in J}c_i^2$ and that $\sum_{i\in J}\|(\boldsymbol{\Delta}_J)_i\|_{\ell_2}^2 = \|\boldsymbol{\Delta}_J\|_F^2$.

Using the properties (30) and (31) remark that

$$\begin{aligned} \left\|\boldsymbol{G}\boldsymbol{\Delta}_J\boldsymbol{G}^\top\right\|_F^2 &= \|\boldsymbol{G}_J\otimes\boldsymbol{G}_J\,vec(\tilde{\boldsymbol{\Delta}}_J)\|_{\ell_2}^2 \\ &\ge \rho_{\min}\left(\boldsymbol{G}_J\otimes\boldsymbol{G}_J\right)\|vec(\tilde{\boldsymbol{\Delta}}_J)\|_{\ell_2}^2 \\ &\ge \rho_{\min}(s)^2\,\|\boldsymbol{\Delta}_J\|_F^2, \end{aligned} \tag{40}$$

where $vec(\tilde{\boldsymbol{\Delta}}_J) = [(\boldsymbol{\Delta}_J)_i^\top]_{i\in J}^\top$. Therefore, combining inequalities (37), (39) and (40) it follows that

$$\left\|\boldsymbol{G}\boldsymbol{\Delta}\boldsymbol{G}^\top\right\|_F^2 \ge \left(\rho_{\min}(s)^2 - c_0\theta(\boldsymbol{G})\rho_{\max}(\boldsymbol{G}^\top\boldsymbol{G})s\right)\|\boldsymbol{\Delta}_J\|_F^2,$$

which completes the proof of Lemma 16. □

Let us now proceed to the proof of Theorem 8. Part of the proof is inspired by results in Bickel et al. (2009). Let $s \leq \min(n,M)$ and $\boldsymbol{\Psi} \in \mathcal{S}_M$ with $\mathcal{M}(\boldsymbol{\Psi}) \leq s$. Let $J = \{k\,; \boldsymbol{\Psi}_k \neq 0\}$. To simplify the notations, write $\widehat{\boldsymbol{\Psi}} = \widehat{\boldsymbol{\Psi}}_\lambda$. By definition of $\widehat{\boldsymbol{\Sigma}}_\lambda = \boldsymbol{G}\widehat{\boldsymbol{\Psi}}\boldsymbol{G}^\top$ one has that

$$\left\|\widetilde{\boldsymbol{S}} - \boldsymbol{G}\widehat{\boldsymbol{\Psi}}\boldsymbol{G}^\top\right\|_F^2 + 2\lambda\sum_{k=1}^M \gamma_k \|\widehat{\boldsymbol{\Psi}}_k\|_{\ell_2} \leq \left\|\widetilde{\boldsymbol{S}} - \boldsymbol{G}\boldsymbol{\Psi}\boldsymbol{G}^\top\right\|_F^2 + 2\lambda\sum_{k=1}^M \gamma_k \|\boldsymbol{\Psi}_k\|_{\ell_2}. \tag{41}$$

Using the scalar product associated to the Frobenius norm $\langle A,B\rangle_F = tr\left(A^\top B\right)$ then

$$\begin{aligned}\left\|\widetilde{\boldsymbol{S}} - \boldsymbol{G}\widehat{\boldsymbol{\Psi}}\boldsymbol{G}^\top\right\|_F^2 &= \left\|\boldsymbol{S} + \boldsymbol{W} - \boldsymbol{G}\widehat{\boldsymbol{\Psi}}\boldsymbol{G}^\top\right\|_F^2 \\ &= \|\boldsymbol{W}\|_F^2 + \left\|\boldsymbol{S} - \boldsymbol{G}\widehat{\boldsymbol{\Psi}}\boldsymbol{G}^\top\right\|_F^2 + 2\left\langle \boldsymbol{W}, \boldsymbol{S} - \boldsymbol{G}\widehat{\boldsymbol{\Psi}}\boldsymbol{G}^\top\right\rangle_F. \end{aligned} \tag{42}$$

Putting (42) in (41) we get

$$\begin{aligned}\left\|\boldsymbol{S} - \boldsymbol{G}\widehat{\boldsymbol{\Psi}}\boldsymbol{G}^\top\right\|_F^2 + 2\lambda\sum_{k=1}^M \gamma_k \|\widehat{\boldsymbol{\Psi}}_k\|_{\ell_2} &\leq \left\|\boldsymbol{S} - \boldsymbol{G}\boldsymbol{\Psi}\boldsymbol{G}^\top\right\|_F^2 + 2\left\langle \boldsymbol{W}, \boldsymbol{G}\left(\widehat{\boldsymbol{\Psi}} - \boldsymbol{\Psi}\right)\boldsymbol{G}^\top\right\rangle_F \\ &\quad + 2\lambda\sum_{k=1}^M \gamma_k \|\boldsymbol{\Psi}_k\|_{\ell_2}.\end{aligned}$$

For $k = 1,\ldots,M$ define the $M \times M$ matrix $\boldsymbol{A}_k$ with all columns equal to zero except the k-th which is equal to $\widehat{\boldsymbol{\Psi}}_k - \boldsymbol{\Psi}_k$. Then, remark that

$$\begin{aligned}\left\langle \boldsymbol{W}, \boldsymbol{G}\left(\widehat{\boldsymbol{\Psi}} - \boldsymbol{\Psi}\right)\boldsymbol{G}^\top\right\rangle_F &= \sum_{k=1}^M \left\langle \boldsymbol{W}, \boldsymbol{G}\boldsymbol{A}_k\boldsymbol{G}^\top\right\rangle_F = \sum_{k=1}^M \left\langle \boldsymbol{G}^\top\boldsymbol{W}\boldsymbol{G}, \boldsymbol{A}_k\right\rangle_F = \sum_{k=1}^M \boldsymbol{\eta}_k^\top(\widehat{\boldsymbol{\Psi}}_k - \boldsymbol{\Psi}_k) \\ &\leq \sum_{k=1}^M \|\boldsymbol{\eta}_k\|_{\ell_2}\|\widehat{\boldsymbol{\Psi}}_k - \boldsymbol{\Psi}_k\|_{\ell_2},\end{aligned}$$

where $\boldsymbol{\eta}_k$ is the k-th column of the matrix $\boldsymbol{G}^\top\boldsymbol{W}\boldsymbol{G}$. Define the event

$$\mathcal{A} = \bigcap_{k=1}^M \{2\|\boldsymbol{\eta}_k\|_{\ell_2} \leq \lambda\gamma_k\}. \tag{43}$$

Then, the choices

$$\gamma_k = 2\|\boldsymbol{G}_k\|_{\ell_2}\sqrt{\rho_{\max}(\boldsymbol{G}\boldsymbol{G}^\top)},\ \lambda = \|\boldsymbol{\Sigma}_{noise}\|_2\left(1 + \sqrt{\frac{n}{N}} + \sqrt{\frac{2\delta\log M}{N}}\right)^2,$$

and Lemma 15 imply that the probability of the complementary event $\mathcal{A}^c$ satisfies

$$\mathbb{P}(\mathcal{A}^c) \leq \sum_{k=1}^M \mathbb{P}(2\|\boldsymbol{\eta}_k\|_{\ell_2} > \lambda\gamma_k) \leq M^{1-\delta}.$$

Then, on the event $\mathcal{A}$ one has that

$$\begin{aligned}\left\|\boldsymbol{S}-\boldsymbol{G}\widehat{\boldsymbol{\Psi}}\boldsymbol{G}^{\top}\right\|_F^2 &\leq \left\|\boldsymbol{S}-\boldsymbol{G}\boldsymbol{\Psi}\boldsymbol{G}^{\top}\right\|_F^2+\lambda\sum_{k=1}^{M}\gamma_k\|\widehat{\boldsymbol{\Psi}}_k-\boldsymbol{\Psi}_k\|_{\ell_2}\\ &\quad+2\lambda\sum_{k=1}^{M}\gamma_k\left(\|\boldsymbol{\Psi}_k\|_{\ell_2}-\|\widehat{\boldsymbol{\Psi}}_k\|_{\ell_2}\right).\end{aligned}$$

Adding the term $\lambda\sum_{k=1}^{M}\gamma_k\|\widehat{\boldsymbol{\Psi}}_k-\boldsymbol{\Psi}_k\|_{\ell_2}$ to both sides of the above inequality yields on the event $\mathcal{A}$

$$\begin{aligned}\left\|\boldsymbol{S}-\boldsymbol{G}\widehat{\boldsymbol{\Psi}}\boldsymbol{G}^{\top}\right\|_F^2+\lambda\sum_{k=1}^{M}\gamma_k\|\widehat{\boldsymbol{\Psi}}_k-\boldsymbol{\Psi}_k\|_{\ell_2} &\leq \left\|\boldsymbol{S}-\boldsymbol{G}\boldsymbol{\Psi}\boldsymbol{G}^{\top}\right\|_F^2\\ &\quad+2\lambda\sum_{k=1}^{M}\gamma_k\left(\|\widehat{\boldsymbol{\Psi}}_k-\boldsymbol{\Psi}_k\|_{\ell_2}+\|\boldsymbol{\Psi}_k\|_{\ell_2}-\|\widehat{\boldsymbol{\Psi}}_k\|_{\ell_2}\right).\end{aligned}$$

Now, remark that for all $k\notin J$, then $\|\widehat{\boldsymbol{\Psi}}_k-\boldsymbol{\Psi}_k\|_{\ell_2}+\|\boldsymbol{\Psi}_k\|_{\ell_2}-\|\widehat{\boldsymbol{\Psi}}_k\|_{\ell_2}=0$, which implies that on the event $\mathcal{A}$

$$\begin{aligned}\left\|\boldsymbol{S}-\boldsymbol{G}\widehat{\boldsymbol{\Psi}}\boldsymbol{G}^{\top}\right\|_F^2+\lambda\sum_{k=1}^{M}\gamma_k\|\widehat{\boldsymbol{\Psi}}_k-\boldsymbol{\Psi}_k\|_{\ell_2} &\leq \left\|\boldsymbol{S}-\boldsymbol{G}\boldsymbol{\Psi}\boldsymbol{G}^{\top}\right\|_F^2 \qquad (44)\\ &\quad+4\lambda\sum_{k\in J}\gamma_k\|\widehat{\boldsymbol{\Psi}}_k-\boldsymbol{\Psi}_k\|_{\ell_2}\\ &\leq \left\|\boldsymbol{S}-\boldsymbol{G}\boldsymbol{\Psi}\boldsymbol{G}^{\top}\right\|_F^2 \qquad (45)\\ &\quad+4\lambda\sqrt{\mathcal{M}(\boldsymbol{\Psi})}\sqrt{\sum_{k\in J}\gamma_k^2\|\widehat{\boldsymbol{\Psi}}_k-\boldsymbol{\Psi}_k\|_{\ell_2}^2}.\end{aligned}$$

where for the last inequality we have used the property that for the positive reals $c_k=\gamma_k\|\widehat{\boldsymbol{\Psi}}_k-\boldsymbol{\Psi}_k\|_{\ell_2}$, $k\in J$ then $\left(\sum_{k\in J}c_k\right)^2\leq\mathcal{M}(\boldsymbol{\Psi})\sum_{k\in J}c_k^2$.

Let $\varepsilon>0$ and define the event

$$\mathcal{A}_1=\left\{4\lambda\sum_{k\in J}\gamma_k\|\widehat{\boldsymbol{\Psi}}_k-\boldsymbol{\Psi}_k\|_{\ell_2}>\varepsilon\left\|\boldsymbol{S}-\boldsymbol{G}\boldsymbol{\Psi}\boldsymbol{G}^{\top}\right\|_F^2\right\}. \qquad (46)$$

Note that on the event $\mathcal{A}\cap\mathcal{A}_1^c$ then the result of the theorem trivially follows from inequality (44). Now consider the event $\mathcal{A}\cap\mathcal{A}_1$ (all the following inequalities hold on this event). Using (44) one has that

$$\lambda\sum_{k=1}^{M}\gamma_k\|\widehat{\boldsymbol{\Psi}}_k-\boldsymbol{\Psi}_k\|_{\ell_2}\leq 4(1+1/\varepsilon)\lambda\sum_{k\in J}\gamma_k\|\widehat{\boldsymbol{\Psi}}_k-\boldsymbol{\Psi}_k\|_{\ell_2}. \qquad (47)$$

Therefore, on $\mathcal{A}\cap\mathcal{A}_1$

$$\sum_{k\notin J}\gamma_k\|\widehat{\boldsymbol{\Psi}}_k-\boldsymbol{\Psi}_k\|_{\ell_2}\leq(3+4/\varepsilon)\sum_{k\in J}\gamma_k\|\widehat{\boldsymbol{\Psi}}_k-\boldsymbol{\Psi}_k\|_{\ell_2}.$$

Let $\boldsymbol{\Delta}$ be the $M\times M$ symmetric matrix with columns equal to $\boldsymbol{\Delta}_k=\gamma_k\left(\widehat{\boldsymbol{\Psi}}_k-\boldsymbol{\Psi}_k\right),k=1,\ldots,M$, and $c_0=3+4/\varepsilon$. Then, the above inequality means that $\sum_{k\in J^c}\|\boldsymbol{\Delta}_k\|_{\ell_2}\leq c_0\sum_{k\in J}\|\boldsymbol{\Delta}_k\|_{\ell_2}$ and thus

Assumption 1 and Lemma 16 imply that

$$\kappa_{s,c_0}^2 \sum_{k\in J} \gamma_k^2 \|\widehat{\Psi}_k - \Psi_k\|_{\ell_2}^2 \leq \left\|G\Delta G^\top\right\|_F^2 \leq 4G_{\max}^2 \rho_{\max}(G^\top G) \left\|G(\widehat{\Psi}-\Psi)G^\top\right\|_F^2. \qquad (48)$$

Let $\gamma_{\max}^2 = 4G_{\max}^2\rho_{\max}(G^\top G)$. Combining the above inequality with (45) yields

$$\begin{aligned}\left\|S - G\widehat{\Psi}G^\top\right\|_F^2 &\leq \left\|S - G\Psi G^\top\right\|_F^2 + 4\lambda\kappa_{s,c_0}^{-1}\gamma_{\max}\sqrt{\mathcal{M}(\Psi)}\left\|G(\widehat{\Psi}-\Psi)G^\top\right\|_F \\ &\leq \left\|S - G\Psi G^\top\right\|_F^2 + 4\lambda\kappa_{s,c_0}^{-1}\gamma_{\max}\sqrt{\mathcal{M}(\Psi)}\left(\left\|G\widehat{\Psi}G^\top - S\right\|_F \right. \\ &\quad \left. + \left\|G\Psi G^\top - S\right\|_F\right)\end{aligned}$$

Now, arguing as in Bickel et al. (2009), a decoupling argument using the inequality $2xy \leq bx^2 + b^{-1}y^2$ with $b>1, x = 2\lambda\kappa_{s,c_0}^{-1}\gamma_{\max}\sqrt{\mathcal{M}(\Psi)}$ and y being either $\left\|G\widehat{\Psi}G^\top - S\right\|_F$ or $\left\|G\Psi G^\top - S\right\|_F$ yields the inequality

$$\left\|S - G\widehat{\Psi}G^\top\right\|_F^2 \leq \left(\frac{b+1}{b-1}\right)\left\|S - G\Psi G^\top\right\|_F^2 + \frac{8b^2\gamma_{\max}^2}{(b-1)\kappa_{s,c_0}^2}\lambda^2\mathcal{M}(\Psi).$$

Then, taking $b = 1 + 2/\varepsilon$ and using the inequalities $\left\|\Sigma - G\widehat{\Psi}G^\top\right\|_F^2 \leq 2\|S-\Sigma\|_F^2 + 2\left\|S - G\widehat{\Psi}G^\top\right\|_F^2$ and $\left\|S - G\Psi G^\top\right\|_F^2 \leq 2\|S-\Sigma\|_F^2 + 2\left\|\Sigma - G\Psi G^\top\right\|_F^2$ completes the proof of Theorem 8. □

A.6 Proof of Theorem 10

Part of the proof is inspired by the approach followed in Lounici (2008) and Lounici et al. (2009). Note first that

$$\max_{1\leq k\leq M} \gamma_k \left\|\widehat{\Psi}_k - \Psi_k^*\right\|_{\ell_2} \leq \sum_{k=1}^M \gamma_k \left\|\widehat{\Psi}_k - \Psi_k^*\right\|_{\ell_2}.$$

Since $\Psi^* \in \{\Psi \in \mathcal{S}_M : M(\Psi) \leq s_*\}$, we can use some results from the proof of Theorem (8). On the event $\mathcal{A}\cap\mathcal{A}_1$, with $\mathcal{A}$ defined by (43) and $\mathcal{A}_1$ defined by (46), inequality (47) implies that

$$\begin{aligned}\sum_{k=1}^M \gamma_k \left\|\widehat{\Psi}_k - \Psi_k^*\right\|_{\ell_2} &\leq 4\left(1+\frac{1}{\varepsilon}\right)\sum_{k\in J^*}\gamma_k\left\|\widehat{\Psi}_k - \Psi_k^*\right\|_{\ell_2} \\ &\leq 4\left(1+\frac{1}{\varepsilon}\right)\sqrt{s_*}\sqrt{\sum_{k\in J^*}\gamma_k^2\left\|\widehat{\Psi}_k - \Psi_k^*\right\|_{\ell_2}^2}.\end{aligned}$$

Let Δ^* be the $M\times M$ symmetric matrix with columns equal to $\Delta_k^* = \gamma_k\left(\widehat{\Psi}_k - \Psi_k^*\right)$, $k=1,\ldots,M$, let $\gamma_{\max} = 2G_{\max}\sqrt{\rho_{\max}(G^\top G)}$ and $c_0 = 3+4/\varepsilon$. Then, the above inequality and (48) imply that on the event $\mathcal{A}\cap\mathcal{A}_1$

$$\begin{aligned}\sum_{k=1}^{M}\gamma_k\left\|\widehat{\boldsymbol{\Psi}}_k-\boldsymbol{\Psi}_k^*\right\|_{\ell_2} &\leq \frac{4\left(1+\frac{1}{\varepsilon}\right)\sqrt{s_*}}{\kappa_{s_*,c_0}}\left\|\boldsymbol{G}\boldsymbol{\Delta}^*\boldsymbol{G}^\top\right\|_F \leq \frac{4\left(1+\frac{1}{\varepsilon}\right)\sqrt{s_*}}{\kappa_{s_*,c_0}}\gamma_{\max}\left\|\boldsymbol{G}\left(\widehat{\boldsymbol{\Psi}}-\boldsymbol{\Psi}^*\right)\boldsymbol{G}^\top\right\|_F \\ &= \frac{4\left(1+\varepsilon\right)\sqrt{s_*}}{\varepsilon\kappa_{s_*,c_0}}\gamma_{\max}\left\|\widehat{\boldsymbol{\Sigma}}_\lambda-\boldsymbol{\Sigma}\right\|_F \\ &\leq \frac{4\left(1+\varepsilon\right)\sqrt{s_*}}{\varepsilon\kappa_{s_*,c_0}}\gamma_{\max}\sqrt{n}\sqrt{C_0\left(n,M,N,s_*,\boldsymbol{S},\boldsymbol{\Psi}^*,\boldsymbol{G},\boldsymbol{\Sigma}_{noise}\right)},\end{aligned}$$

Then, using (44) one has that on the event $\mathcal{A}\cap\mathcal{A}_1^c$

$$\sum_{k=1}^{M}\gamma_k\left\|\widehat{\boldsymbol{\Psi}}_k-\boldsymbol{\Psi}_k^*\right\|_{\ell_2}\leq\frac{1+\varepsilon}{\lambda}\left\|\boldsymbol{S}-\boldsymbol{G}\boldsymbol{\Psi}^*\boldsymbol{G}^\top\right\|_F^2.$$

Therefore, by definition of C_1, the previous inequalities imply that on the event $\mathcal{A}$ (of probability $1-M^{1-\delta}$)

$$\sum_{k=1}^{M}\frac{\|\boldsymbol{G}_k\|_{\ell_2}}{\sqrt{n}\boldsymbol{G}_{\max}}\left\|\widehat{\boldsymbol{\Psi}}_k-\boldsymbol{\Psi}_k^*\right\|_{\ell_2}\leq C_1\left(n,M,N,s_*,\boldsymbol{S},\boldsymbol{\Psi}^*,\boldsymbol{G},\boldsymbol{\Sigma}_{noise}\right). \tag{49}$$

Hence $\max_{1\leq k\leq M}\frac{\delta_k}{\sqrt{n}}\left\|\widehat{\boldsymbol{\Psi}}_k-\boldsymbol{\Psi}_k^*\right\|_{\ell_2}\leq C_1\left(\sigma,n,M,N,s_*,\boldsymbol{G},\boldsymbol{\Sigma}_{noise}\right)$ with probability at least $1-M^{1-\delta}$, which proves the first assertion of Theorem 10.

Then, to prove that $\hat{J}=J^*$ we use that $\frac{\delta_k}{\sqrt{n}}\left|\left\|\widehat{\boldsymbol{\Psi}}_k\right\|_{\ell_2}-\left\|\boldsymbol{\Psi}_k^*\right\|_{\ell_2}\right|\leq\frac{\delta_k}{\sqrt{n}}\left\|\widehat{\boldsymbol{\Psi}}_k-\boldsymbol{\Psi}_k^*\right\|_{\ell_2}$ for all $k=1,\ldots,M$. Then, by (49)

$$\left|\frac{\delta_k}{\sqrt{n}}\left\|\widehat{\boldsymbol{\Psi}}_k\right\|_{\ell_2}-\frac{\delta_k}{\sqrt{n}}\left\|\boldsymbol{\Psi}_k^*\right\|_{\ell_2}\right|\leq C_1\left(n,M,N,s_*,\boldsymbol{S},\boldsymbol{\Psi}^*,\boldsymbol{G},\boldsymbol{\Sigma}_{noise}\right),$$

which is equivalent to

$$-C_1\left(n,M,N,s_*,\boldsymbol{S},\boldsymbol{\Psi}^*,\boldsymbol{G},\boldsymbol{\Sigma}_{noise}\right)\leq\frac{\delta_k}{\sqrt{n}}\left\|\widehat{\boldsymbol{\Psi}}_k\right\|_{\ell_2}-\frac{\delta_k}{\sqrt{n}}\left\|\boldsymbol{\Psi}_k^*\right\|_{\ell_2}\leq C_1\left(n,M,N,s_*,\boldsymbol{S},\boldsymbol{\Psi}^*,\boldsymbol{G},\boldsymbol{\Sigma}_{noise}\right). \tag{50}$$

If $k\in\hat{J}$ then $\frac{\delta_k}{\sqrt{n}}\left\|\widehat{\boldsymbol{\Psi}}_k\right\|_{\ell_2}>C_1\left(n,M,N,s_*,\boldsymbol{S},\boldsymbol{\Psi}^*,\boldsymbol{G},\boldsymbol{\Sigma}_{noise}\right)$. Inequality $\frac{\delta_k}{\sqrt{n}}\left\|\widehat{\boldsymbol{\Psi}}_k\right\|_{\ell_2}-\frac{\delta_k}{\sqrt{n}}\left\|\boldsymbol{\Psi}_k^*\right\|_{\ell_2}\leq C_1\left(n,M,N,s_*,\boldsymbol{S},\boldsymbol{\Psi}^*,\boldsymbol{G},\boldsymbol{\Sigma}_{noise}\right)$ from (50) imply that $\frac{\delta_k}{\sqrt{n}}\left\|\boldsymbol{\Psi}_k^*\right\|_{\ell_2}\geq\frac{\delta_k}{\sqrt{n}}\left\|\widehat{\boldsymbol{\Psi}}_k\right\|_{\ell_2}-C_1\left(n,M,N,s_*,\boldsymbol{S},\boldsymbol{\Psi}^*,\boldsymbol{G},\boldsymbol{\Sigma}_{noise}\right)>0$, where the last inequality is obtained using that $k\in\hat{J}$. Hence $\left\|\boldsymbol{\Psi}_k^*\right\|_{\ell_2}>0$ and therefore $k\in J^*$. If $k\in J^*$ then $\left\|\boldsymbol{\Psi}_k^*\right\|_{\ell_2}\neq 0$. Inequality $-C_1\left(n,M,N,s_*,\boldsymbol{S},\boldsymbol{\Psi}^*,\boldsymbol{G},\boldsymbol{\Sigma}_{noise}\right)\leq\frac{\delta_k}{\sqrt{n}}\left\|\widehat{\boldsymbol{\Psi}}_k\right\|_{\ell_2}-\frac{\delta_k}{\sqrt{n}}\left\|\boldsymbol{\Psi}_k^*\right\|_{\ell_2}$ from (50) imply that $\frac{\delta_k}{\sqrt{n}}\left\|\widehat{\boldsymbol{\Psi}}_k\right\|_{\ell_2}+C_1\left(n,M,N,s_*,\boldsymbol{S},\boldsymbol{\Psi}^*,\boldsymbol{G},\boldsymbol{\Sigma}_{noise}\right)\geq\frac{\delta_k}{\sqrt{n}}\left\|\boldsymbol{\Psi}_k^*\right\|_{\ell_2}>2C_1\left(n,M,N,s_*,\boldsymbol{S},\boldsymbol{\Psi}^*,\boldsymbol{G},\boldsymbol{\Sigma}_{noise}\right)$, where the last inequality is obtained using Assumption (22) on $\frac{\delta_k}{\sqrt{n}}\left\|\boldsymbol{\Psi}_k^*\right\|_{\ell_2}$. Hence $\frac{\delta_k}{\sqrt{n}}\left\|\widehat{\boldsymbol{\Psi}}_k\right\|_{\ell_2}>2C_1\left(n,M,N,s_*,\boldsymbol{S},\boldsymbol{\Psi}^*,\boldsymbol{G},\boldsymbol{\Sigma}_{noise}\right)-C_1\left(n,M,N,s_*,\boldsymbol{S},\boldsymbol{\Psi}^*,\boldsymbol{G},\boldsymbol{\Sigma}_{noise}\right)=C_1\left(n,M,N,s_*,\boldsymbol{S},\boldsymbol{\Psi}^*,\boldsymbol{G},\boldsymbol{\Sigma}_{noise}\right)$ and therefore $k\in\hat{J}$. This completes the proof of Theorem 10. □

A.7 Proof of Theorem 11

Under the assumptions of Theorem 11, we have shown in the proof of Theorem 10 that $\hat{J} = J^*$ on the event $\mathcal{A}$ defined by (43). Therefore, under the assumptions of Theorem 11 it can be checked that on the event $\mathcal{A}$ (of probability $1 - M^{1-\delta}$)

$$\widehat{\boldsymbol{\Sigma}}_{\hat{J}} = \widehat{\boldsymbol{\Sigma}}_{J^*} = \boldsymbol{G}_{J^*} \widehat{\boldsymbol{\Psi}}_{J^*} \boldsymbol{G}_{J^*}^\top,$$

with

$$\widehat{\boldsymbol{\Psi}}_{J^*} = \left(\boldsymbol{G}_{J^*}^\top \boldsymbol{G}_{J^*}\right)^{-1} \boldsymbol{G}_{J^*}^\top \widetilde{\boldsymbol{S}} \boldsymbol{G}_{J^*} \left(\boldsymbol{G}_{J^*}^\top \boldsymbol{G}_{J^*}\right)^{-1}.$$

Now, from the definition (24) of $\boldsymbol{\Sigma}_{J^*}$ it follows that on the event $\mathcal{A}$

$$\left\| \widehat{\boldsymbol{\Sigma}}_{\hat{J}} - \boldsymbol{\Sigma}_{J^*} \right\|_2 \leq \rho_{\max}\left(\boldsymbol{G}_{J^*}^\top \boldsymbol{G}_{J^*}\right) \left\| \widehat{\boldsymbol{\Psi}}_{J^*} - \Lambda_{J^*} \right\|_2 \tag{51}$$

where $\Lambda_{J^*} = \boldsymbol{\Psi}_{J^*} + (\boldsymbol{G}_{J^*}^\top \boldsymbol{G}_{J^*})^{-1} \boldsymbol{G}_{J^*}^\top \boldsymbol{\Sigma}_{noise} \boldsymbol{G}_{J^*} \left(\boldsymbol{G}_{J^*}^\top \boldsymbol{G}_{J^*}\right)^{-1}$. Let $\boldsymbol{Y}_i = \left(\boldsymbol{G}_{J^*}^\top \boldsymbol{G}_{J^*}\right)^{-1} \boldsymbol{G}_{J^*}^\top \widetilde{\boldsymbol{X}_i}$ for $i = 1, \ldots, N$ and remark that

$$\widehat{\boldsymbol{\Psi}}_{J^*} = \frac{1}{N} \sum_{i=1}^N \boldsymbol{Y}_i \boldsymbol{Y}_i^\top \text{ with } \mathbb{E}\widehat{\boldsymbol{\Psi}}_{J^*} = \Lambda_{J^*}.$$

Therefore, $\widehat{\boldsymbol{\Psi}}_{J^*}$ is a sample covariance matrix of size $s_* \times s_*$ and we can control its deviation in operator norm from Λ_{J^*} by using Proposition 6. For this we simply have to verify conditions similar to **(A1)** and **(A2)** in Assumption 2 for the random vector $\boldsymbol{Y} = \left(\boldsymbol{G}_{J^*}^\top \boldsymbol{G}_{J^*}\right)^{-1} \boldsymbol{G}_{J^*}^\top \widetilde{\boldsymbol{X}} \in \mathbb{R}^{s_*}$. First, let $\beta \in \mathbb{R}^{s_*}$ with $\|\beta\|_{\ell_2} = 1$. Then, remark that $\boldsymbol{Y}^\top \beta = \widetilde{\boldsymbol{X}}^\top \tilde{\beta}$ with $\tilde{\beta} = \boldsymbol{G}_{J^*} \left(\boldsymbol{G}_{J^*}^\top \boldsymbol{G}_{J^*}\right)^{-1} \beta$. Since $\|\tilde{\beta}\|_{\ell_2} \leq \left(\rho_{\min}\left(\boldsymbol{G}_{J^*}^\top \boldsymbol{G}_{J^*}\right)\right)^{-1/2}$ it follows that

$$\left(\mathbb{E}|\boldsymbol{Y}^\top \beta|^4\right)^{1/4} \leq \tilde{\rho}(\boldsymbol{\Sigma}, \boldsymbol{\Sigma}_{noise}) \rho_{\min}^{-1/2} \left(\boldsymbol{G}_{J^*}^\top \boldsymbol{G}_{J^*}\right), \tag{52}$$

where $\tilde{\rho}(\boldsymbol{\Sigma}, \boldsymbol{\Sigma}_{noise}) = 8^{1/4} \left(\rho^4(\boldsymbol{\Sigma}) + \rho^4(\boldsymbol{\Sigma}_{noise})\right)^{1/4}$. Now let

$$\tilde{Z} = \|\boldsymbol{Y}\|_{\ell_2} \leq \rho_{\min}^{-1/2} \left(\boldsymbol{G}_{J^*}^\top \boldsymbol{G}_{J^*}\right) \|\widetilde{\boldsymbol{X}}\|_{\ell_2}.$$

Given our assumptions on the process $\widetilde{X} = X + \mathcal{E}$ it follows that there exists $\alpha \geq 1$ such that

$$\|\tilde{Z}\|_{\psi_\alpha} \leq \rho_{\min}^{-1/2} \left(\boldsymbol{G}_{J^*}^\top \boldsymbol{G}_{J^*}\right) \left(\|Z\|_{\psi_\alpha} + \|W\|_{\psi_\alpha}\right) < +\infty, \tag{53}$$

where $Z = \|\boldsymbol{X}\|_{\ell_2}$ and $W = \|\mathcal{E}\|_{\ell_2}$, with $\boldsymbol{X} = (X(t_1), ..., X(t_n))^\top$ and $\mathcal{E} = (\mathcal{E}(t_1), ..., \mathcal{E}(t_n))^\top$. Finally, remark that

$$\|\Lambda_{J^*}\|_2 \leq \|\boldsymbol{\Psi}_{J^*}\|_2 + \rho_{\min}^{-1}\left(\boldsymbol{G}_{J^*}^\top \boldsymbol{G}_{J^*}\right) \|\boldsymbol{\Sigma}_{noise}\|_2. \tag{54}$$

Hence, using the relations (52) and (53), the bound (54) and Proposition 6 (with $\boldsymbol{Y}$ instead of $\boldsymbol{X}$), it follows that there exists a universal constant $\delta_* > 0$ such that for all $x > 0$,

$$\mathbb{P}\left(\left\| \widehat{\boldsymbol{\Psi}}_{J^*} - \Lambda_{J^*} \right\|_2 \geqslant \tilde{\tau}_{N,s_*} x\right) \leqslant \exp\left(-(\delta_*^{-1} x)^{\frac{\alpha}{2+\alpha}}\right), \tag{55}$$

where $\tilde{\tau}_{N,s_*} = \max(\tilde{A}^2_{N,s_*}, \tilde{B}_{N,s_*})$, with $\tilde{A}_{N,s_*} = \|\tilde{Z}\|_{\psi_\alpha} \frac{\sqrt{\log d^*}(\log N)^{1/\alpha}}{\sqrt{N}}$ and

$$\tilde{B}_{N,s_*} = \frac{\tilde{\rho}^2(\boldsymbol{\Sigma}, \boldsymbol{\Sigma}_{noise}) \rho^{-1}_{\min}\left(\boldsymbol{G}^\top_{J^*}\boldsymbol{G}_{J^*}\right)}{\sqrt{N}} + \left(\|\boldsymbol{\Psi}_{J^*}\|_2 + \rho^{-1}_{\min}\left(\boldsymbol{G}^\top_{J^*}\boldsymbol{G}_{J^*}\right)\|\boldsymbol{\Sigma}_{noise}\|_2\right)^{1/2} \tilde{A}_{N,s_*},$$

with $d^* = \min(N, s_*)$. Then, define the event

$$\mathcal{B} = \left\| \widehat{\boldsymbol{\Psi}}_{J^*} - \Lambda_{J^*} \right\|_2 \leqslant \tilde{\tau}_{N,s_*} \delta_\star \left(\log(M)\right)^{\frac{2+\alpha}{\alpha}},$$

and note that, for $x = \delta_\star \left(\log(M)\right)^{\frac{2+\alpha}{\alpha}}$ with $\delta_\star > \delta_*$, inequality (55) implies that $\mathbb{P}(\mathcal{B}) \geq 1 - M^{-\left(\frac{\delta_\star}{\delta_*}\right)^{\frac{\alpha}{2+\alpha}}}$. Therefore, on the event $\mathcal{A} \cap \mathcal{B}$ (of probability at least $1 - M^{1-\delta} - M^{-\left(\frac{\delta_\star}{\delta_*}\right)^{\frac{\alpha}{2+\alpha}}}$), using inequality (51) and the fact that $\hat{J} = J^*$ one obtains

$$\left\| \widehat{\boldsymbol{\Sigma}}_{\hat{J}} - \boldsymbol{\Sigma}_{J^*} \right\|_2 \leq \rho_{\max}\left(\boldsymbol{G}^\top_{J^*}\boldsymbol{G}_{J^*}\right) \tilde{\tau}_{N,s_*} \delta_\star \left(\log(M)\right)^{\frac{2+\alpha}{\alpha}},$$

which completes the proof of Theorem 11. □

References

Anestis Antoniadis, Jeremie Bigot, and Theofanis Sapatinas. Wavelet estimators in nonparametric regression: A comparative simulation study. *Journal of Statistical Software*, 6(6):1–83, 6 2001.

Francis R. Bach. Consistency of the group lasso and multiple kernel learning. *J. Mach. Learn. Res.*, 9:1179–1225, 2008.

Peter J. Bickel and Elizaveta Levina. Regularized estimation of large covariance matrices. *Ann. Statist.*, 36(1):199–227, 2008a.

Peter J. Bickel and Elizaveta Levina. Covariance regularization by thresholding. *Ann. Statist.*, 36 (6):2577–2604, 2008b.

Peter J. Bickel, Ya'acov Ritov, and Alexandre B. Tsybakov. Simultaneous analysis of lasso and Dantzig selector. *Ann. Statist.*, 37(4):1705–1732, 2009.

Jérémie Bigot, Rolando J. Biscay, Jean-Michel Loubes, and Lilian Muñiz Alvarez. Nonparametric estimation of covariance functions by model selection. *Electronic Journal of Statistics*, 4:822–855, 2010.

Stephen Boyd and Lieven Vandenberghe. *Convex Optimization*. Cambridge University Press, Cambridge, 2004. ISBN 0-521-83378-7.

Noel A. C. Cressie. *Statistics for Spatial Data*. Wiley Series in Probability and Mathematical Statistics: Applied Probability and Statistics. John Wiley & Sons Inc., New York, 1993.

Alexandre d'Aspremont, Francis Bach, and Laurent El Ghaoui. Optimal solutions for sparse principal component analysis. *J. Mach. Learn. Res.*, 9:1269–1294, 2008.

Kenneth R. Davidson and Stanislaw J. Szarek. Local operator theory, random matrices and Banach spaces. In *Handbook of the Geometry of Banach Spaces, Vol. I*, pages 317–366. North-Holland, Amsterdam, 2001.

Chandler Davis and W. M. Kahan. The rotation of eigenvectors by a perturbation. III. *SIAM J. Numer. Anal.*, 7:1–46, 1970.

Noureddine El Karoui. Operator norm consistent estimation of large-dimensional sparse covariance matrices. *Ann. Statist.*, 36(6):2717–2756, 2008.

Jianquin Fan, Yingying Fan, and Jinchi Lv. High dimensional covariance matrix estimation using a factor model. *Journal of Econometrics*, 147:186–197, 2008.

Junzhou Huang and Tong Zhang. The benefit of group sparsity. *Ann. Statist.*, 38(4):1978–2004, 2010. ISSN 0090-5364.

Iain M. Johnstone. On the distribution of the largest eigenvalue in principal components analysis. *Ann. Statist.*, 29(2):295–327, 2001.

Iain M. Johnstone and Arthur Y. Lu. On consistency and sparsity for principal components analysis in high dimensions. *Journal of the American Statistical Association*, 104(486):682–693, June 2009.

Andre G. Journel. Kriging in terms of projections. *J. Internat. Assoc. Mathematical Geol.*, 9(6): 563–586, 1977.

Clifford Lam and Jianqing Fan. Sparsistency and rates of convergence in large covariance matrix estimation. *Ann. Statist.*, 37(6B):4254–4278, 2009.

Elizaveta Levina, Adam Rothman, and Ji Zhu. Sparse estimation of large covariance matrices via a nested Lasso penalty. *Ann. Appl. Stat.*, 2(1):245–263, 2008.

Karim Lounici. Sup-norm convergence rate and sign concentration property of Lasso and Dantzig estimators. *Electron. J. Stat.*, 2:90–102, 2008.

Karim Lounici, Massimiliano Pontil, Alexandre B. Tsybakov, and Sara van de Geer. Taking advantage of sparsity in multi-task learning. *COLT*, 2009.

Karim Lounici, Massimiliano Pontil, Alexandre B. Tsybakov, and Sara van de Geer. Oracle Inequalities and Optimal Inference under Group Sparsity. *Ann. Statist.*, to be published., 2011.

Shahar Mendelson and Alain Pajor. On singular values of matrices with independent rows. *Bernoulli*, 12(5):761–773, 2006.

Yuval Nardi and Alessandro Rinaldo. On the asymptotic properties of the group lasso estimator for linear models. *Electron. J. Stat.*, 2:605–633, 2008.

Adam J. Rothman, Peter J. Bickel, Elizaveta Levina, and Ji Zhu. Sparse permutation invariant covariance estimation. *Electron. J. Stat.*, 2:494–515, 2008.

Mark Schmidt, Kevin Murphy, Glenn Fung, and Rómer Rosales. Structure learning in random fields for heart motion abnormality detection (addendum). *CVPR08*, 2008.

Michael L. Stein. *Interpolation of Spatial Data. Some Theory for Kriging*. Springer Series in Statistics. New York, NY: Springer. xvii, 247 p., 1999.

Christopher K. Wikle and Noel Cressie. A dimension-reduced approach to space-time Kalman filtering. *Biometrika*, 86(4):815–829, 1999. ISSN 0006-3444.

Hui Zou, Trevor Hastie, and Robert Tibshirani. Sparse principal component analysis. *J. Comput. Graph. Statist.*, 15(2):265–286, 2006.

Journal of Machine Learning Research 12 (2011) 3227-3257 Submitted 6/11; Revised 10/11; Published 11/11

Robust Gaussian Process Regression with a Student-*t* Likelihood

Pasi Jylänki PASI.JYLANKI@AALTO.FI
Department of Biomedical Engineering and Computational Science
Aalto University School of Science
P.O. Box 12200
FI-00076 Aalto
Finland

Jarno Vanhatalo JARNO.VANHATALO@HELSINKI.FI
Department of Environmental Sciences
University of Helsinki
P.O. Box 65
FI-00014 Helsinki
Finland

Aki Vehtari AKI.VEHTARI@AALTO.FI
Department of Biomedical Engineering and Computational Science
Aalto University School of Science
P.O. Box 12200
FI-00076 Aalto
Finland

Editor: Neil Lawrence

Abstract

This paper considers the robust and efficient implementation of Gaussian process regression with a Student-*t* observation model, which has a non-log-concave likelihood. The challenge with the Student-*t* model is the analytically intractable inference which is why several approximative methods have been proposed. Expectation propagation (EP) has been found to be a very accurate method in many empirical studies but the convergence of EP is known to be problematic with models containing non-log-concave site functions. In this paper we illustrate the situations where standard EP fails to converge and review different modifications and alternative algorithms for improving the convergence. We demonstrate that convergence problems may occur during the type-II maximum a posteriori (MAP) estimation of the hyperparameters and show that standard EP may not converge in the MAP values with some difficult data sets. We present a robust implementation which relies primarily on parallel EP updates and uses a moment-matching-based double-loop algorithm with adaptively selected step size in difficult cases. The predictive performance of EP is compared with Laplace, variational Bayes, and Markov chain Monte Carlo approximations.

Keywords: Gaussian process, robust regression, Student-*t* distribution, approximate inference, expectation propagation

1. Introduction

In many regression problems observations may include outliers which deviate strongly from the other members of the sample. Such outliers may occur, for example, because of failures in the

measurement process or absence of certain relevant explanatory variables in the model. In such cases, a robust observation model is required.

Robust inference has been studied extensively. De Finetti (1961) described how Bayesian inference on the mean of a random sample, assuming a suitable observation model, naturally leads to giving less weight to outlying observations. However, in contrast to simple rejection of outliers, the posterior depends on all data but in the limit, as the separation between the outliers and the rest of the data increases, the effect of outliers becomes negligible. More theoretical results on this kind of outlier rejection were presented by Dawid (1973) who gave sufficient conditions on the observation model $p(y|\theta)$ and the prior distribution $p(\theta)$ of an unknown location parameter θ, which ensure that the posterior expectation of a given function $m(\theta)$ tends to the prior as $y \to \infty$. He also stated that the Student-t distribution combined with a normal prior has this property.

A more formal definition of robustness was given by O'Hagan (1979) in terms of an outlier-prone observation model. The observation model is defined to be outlier-prone of order n, if $p(\theta|y_1,...,y_{n+1}) \to p(\theta|y_1,...,y_n)$ as $y_{n+1} \to \infty$. That is, the effect of a single conflicting observation to the posterior becomes asymptotically negligible as the observation approaches infinity. O'Hagan (1979) showed that the Student-t distribution is outlier prone of order 1, and that it can reject up to m outliers if there are at least $2m$ observations altogether. This contrasts heavily with the commonly used Gaussian observation model in which each observation influences the posterior no matter how far it is from the others.

In nonlinear Gaussian process (GP) regression context the outlier rejection is more complicated and one may consider the posterior distribution of the unknown function values $f_i = f(x_i)$ locally near some input locations x_i. Depending on the smoothness properties defined through the prior on f_i, m observations can be rejected locally if there are at least $2m$ data points nearby. However, already two conflicting data points can render the posterior distribution multimodal making the posterior inference challenging (these issues will be illustrated in the upcoming sections).

In this work, we adopt the Student-t observation model for GP regression because of its good robustness properties which can be altered continuously from a very heavy tailed distribution to the Gaussian model with the degrees of freedom parameter. This allows the extent of robustness to be determined from the data through hyperparameter inference. The Student-t observation model was studied in linear regression by West (1984) and Geweke (1993), and Neal (1997) introduced it for GP regression. Other robust observation models which have been used in GP regression include, for example, mixtures of Gaussians (Kuss, 2006; Stegle et al., 2008), the Laplace distribution (Kuss, 2006), and input dependent observation models (Goldberg et al., 1998; Naish-Guzman and Holden, 2008).

The challenge with the Student-t model is the analytically intractable inference. A common approach has been to use the scale-mixture representation of the Student-t distribution (Geweke, 1993), which enables Gibbs sampling (Geweke, 1993; Neal, 1997), and a factorizing variational approximation (fVB) for the posterior inference (Tipping and Lawrence, 2005; Kuss, 2006). Recently Vanhatalo et al. (2009) compared fVB with the Laplace approximation (see, e.g., Rasmussen and Williams, 2006) and showed that Laplace's method provided slightly better predictive performance with less computational burden. They also showed that fVB tends to underestimate the posterior uncertainties of the predictions because it assumes the scales and the unknown function values a posteriori independent. Another variational approach called variational bounds (VB) is available in the GPML software package (Rasmussen and Nickisch, 2010). The method is based on forming an un-normalized Gaussian lower bound for each non-Gaussian likelihood term independently

(see Nickisch and Rasmussen, 2008, for details and comparisons in GP classification). Yet another related variational approach is described by Opper and Archambeau (2009) who studied the Cauchy observation model (Student-t with degrees of freedom 1). This method is similar to the KL-divergence minimization approach (KL) described by Nickisch and Rasmussen (2008) and the VB approach can be regarded as a special case of KL. The extensive comparisons by Nickisch and Rasmussen (2008) in GP classification suggest that VB provides better predictive performance than the Laplace approximation but worse marginal likelihood estimates than KL or expectation propagation (EP) (Minka, 2001a). According to the comparisons of Nickisch and Rasmussen (2008), EP is the method of choice since it is much faster than KL, at least in GP classification. The problem with EP, however, is that the Student-t likelihood is not log-concave which may lead to convergence problems (Seeger, 2008).

In this paper, we focus on establishing a robust EP implementation for the Student-t observation model. We illustrate the convergence problems of standard EP with simple one-dimensional regression examples and discuss how damping, fractional EP updates (or power EP) (Minka, 2004; Seeger, 2005), and double-loop algorithms (Heskes and Zoeter, 2002) can be used to improve the convergence. We present a robust implementation which relies primarily on parallel EP updates (see, e.g., van Gerven et al., 2009) and uses a moment-matching-based double-loop algorithm with adaptively selected step size to find stationary solutions in difficult cases. We show that the implementation enables a robust type-II maximum a posteriori (MAP) estimation of the hyperparameters based on the approximative marginal likelihood. The proposed implementation is general so that it could be applied also to other models having non-log-concave likelihoods. The predictive performance of EP is compared to the Laplace approximation, fVB, VB, and Markov chain Monte Carlo (MCMC) using one simulated and three real-world data sets.

2. Gaussian Process Regression with the Student-t Observation Model

We will consider a regression problem, with scalar observations $y_i = f(\mathbf{x}_i) + \varepsilon_i, i = 1,...,n$ at input locations $\mathbf{X} = \{\mathbf{x}_i\}_{i=1}^n$, and where the observation errors $\varepsilon_1,...,\varepsilon_n$ are zero-mean exchangeable random variables. The object of inference is the latent function $f(\mathbf{x}) : \Re^d \to \Re$, which is given a Gaussian process prior

$$f(\mathbf{x})|\theta \sim \mathcal{GP}\left(m(\mathbf{x}), k(\mathbf{x},\mathbf{x}'|\theta)\right), \tag{1}$$

where $m(\mathbf{x})$ and $k(\mathbf{x},\mathbf{x}'|\theta)$ are the mean and covariance functions of the process controlled by hyperparameters θ. For notational simplicity we will assume a zero mean GP. By definition, a Gaussian process prior implies that any finite subset of latent variables, $\mathbf{f} = \{f(\mathbf{x}_i)\}_{i=1}^n$, has a multivariate Gaussian distribution. In particular, at the observed input locations $\mathbf{X}$ the latent variables are distributed as $p(\mathbf{f}|\mathbf{X},\theta) = \mathcal{N}(\mathbf{f}|\mathbf{0},\mathbf{K})$, where $\mathbf{K}$ is the covariance matrix with entries $\mathbf{K}_{ij} = k(\mathbf{x}_i,\mathbf{x}_j|\theta)$. The covariance function encodes the prior assumptions on the latent function, such as the smoothness and scale of the variation, and can be chosen freely as long as the covariance matrices which it produces are symmetric and positive semi-definite. An example of a stationary covariance function is the squared exponential

$$k_{\mathrm{se}}(\mathbf{x}_i,\mathbf{x}_j|\theta) = \sigma_{\mathrm{se}}^2 \exp\left(-\sum_{k=1}^{d} \frac{(x_{i,k}-x_{j,k})^2}{2l_k^2}\right), \tag{2}$$

where $\theta = \{\sigma^2_{se}, l_1, ..., l_d\}$, σ^2_{se} is a magnitude parameter which scales the overall variation of the unknown function, and l_k is a length-scale parameter which governs how fast the correlation decreases as the distance increases in the input dimension k.

The traditional assumption is that given $\mathbf{f}$ the error terms ε_i are i.i.d. Gaussian: $\varepsilon_i \sim \mathcal{N}(0, \sigma^2)$. In this case, the marginal likelihood $p(\mathbf{y}|\mathbf{X}, \theta, \sigma^2)$ and the conditional posterior of the latent variables $p(\mathbf{f}|\mathcal{D}, \theta, \sigma^2)$, where $\mathcal{D} = \{\mathbf{y}, \mathbf{X}\}$, have an analytical solution. This is computationally convenient since approximate methods are needed only for the inference on the hyperparameters θ and σ^2. The robust Student-t observation model

$$p(y_i|f_i, \sigma^2, \nu) = \frac{\Gamma((\nu+1)/2)}{\Gamma(\nu/2)\sqrt{\nu\pi}\sigma}\left(1 + \frac{(y_i - f_i)^2}{\nu\sigma^2}\right)^{-(\nu+1)/2},$$

where $f_i = f(\mathbf{x}_i)$, ν is the degrees of freedom and σ the scale parameter (Gelman et al., 2004), is computationally challenging. The marginal likelihood and the conditional posterior $p(\mathbf{f}|\mathcal{D}, \theta, \sigma^2, \nu)$ are not anymore analytically tractable but require some method for approximate inference.

3. Approximate Inference

In this section, we review the approximate inference methods considered in this paper. First we give a short description of MCMC and the Laplace approximation, as well as two variational methods, fVB and VB. Then we give a more detailed description of the EP algorithm and review ways to improve the convergence in more difficult problems.

3.1 Markov Chain Monte Carlo

The MCMC approach is based on drawing samples from $p(\mathbf{f}, \theta, \sigma^2, \nu|\mathcal{D})$ and using these samples to represent the posterior distribution and to numerically approximate integrals over the latent variables and the hyperparameters. Instead of implementing a Markov chain sampler directly for the Student-t model, a more common approach is to use the Gibbs sampling based on the following scale mixture representation of the Student-t distribution

$$\begin{aligned} &y_i|f_i, V_i \sim \mathcal{N}(f_i, V_i), \\ &V_i|\nu, \sigma^2 \sim \text{Inv-}\chi^2(\nu, \sigma^2), \end{aligned} \tag{3}$$

where each observation has its own Inv-χ^2-distributed noise variance V_i (Neal, 1997; Gelman et al., 2004). Sampling of the hyperparameters θ can be done with any general sampling algorithm, such as the Slice sampling or the hybrid Monte Carlo (HMC) (see, e.g., Gelman et al., 2004). The Gibbs sampler on the scale mixture (3) converges often slowly and may get stuck for long times in small values of σ^2 because of the dependence between V_i and σ^2. This can be avoided by reparameterization $V_i = \alpha^2 U_i$, where $U_i \sim \text{Inv-}\chi^2(\nu, \tau^2)$, $p(\tau^2) \propto 1/\tau^2$, and $p(\alpha^2) \propto 1/\alpha^2$ (Gelman et al., 2004). This improves mixing of the chains and reduces the autocorrelations but introduces an implicit prior for the scale parameter $\sigma^2 = \alpha^2\tau^2$ of the Student-t model. An alternative parameterization proposed by Liu and Rubin (1995), where $V_i = \sigma^2/\gamma_i$ and $\gamma_i \sim \text{Gamma}(\nu/2, \nu/2)$, also decouples σ^2 and V_i but does not introduce the additional scale parameter τ. It could also lead to better mixing without the implicit scale prior but in the experiments we used the decomposition of Gelman et al. (2004) because the results were not sensitive to the choice of prior on σ^2.

3.2 Laplace Approximation (LA)

The Laplace approximation for the conditional posterior of the latent function is constructed from the second order Taylor expansion of $\log p(\mathbf{f}|\mathcal{D},\theta,\sigma^2,\nu)$ around the mode $\hat{\mathbf{f}}$, which gives a Gaussian approximation to the conditional posterior

$$p(\mathbf{f}|\mathcal{D},\theta,\sigma^2,\nu) \approx q(\mathbf{f}|\mathcal{D},\theta,\sigma^2,\nu) = \mathcal{N}(\mathbf{f}|\hat{\mathbf{f}},\boldsymbol{\Sigma}_{\mathrm{LA}}),$$

where $\hat{\mathbf{f}} = \arg\max_{\mathbf{f}} p(\mathbf{f}|\mathcal{D},\theta,\sigma^2,\nu)$ (Rasmussen and Williams, 2006). $\boldsymbol{\Sigma}_{\mathrm{LA}}^{-1}$ is the Hessian of the negative log conditional posterior at the mode, that is,

$$\boldsymbol{\Sigma}_{\mathrm{LA}}^{-1} = -\nabla\nabla \log p(\mathbf{f}|\mathcal{D},\theta,\sigma^2,\nu)|_{\mathbf{f}=\hat{\mathbf{f}}} = \mathbf{K}^{-1} + \mathbf{W}, \tag{4}$$

where $\mathbf{W}$ is a diagonal matrix with entries $\mathbf{W}_{ii} = \nabla_{f_i}\nabla_{f_i} \log p(y|f_i,\sigma^2,\nu)|_{f_i=\hat{f}_i}$.

The inference in the hyperparameters is done by approximating the conditional marginal likelihood $p(\mathbf{y}|\mathbf{X},\theta,\sigma^2,\nu)$ with Laplace's method and searching for the approximate maximum a posterior estimate for the hyperparameters

$$\{\hat{\theta},\hat{\sigma}^2,\hat{\nu}\} = \arg\max_{\theta,\sigma^2,\nu} \left[\log q(\theta,\sigma^2,\nu|\mathcal{D})\right] = \arg\max_{\theta,\sigma^2,\nu} \left[\log q(\mathbf{y}|\mathbf{X},\theta,\sigma^2,\nu) + \log p(\theta,\sigma^2,\nu)\right],$$

where $p(\theta,\sigma^2,\nu)$ is the prior of the hyperparameters. The gradients of the approximate log marginal likelihood can be solved analytically, which enables the MAP estimation of the hyperparameters with gradient based optimization methods. Following Williams and Barber (1998) the approximation scheme is called the Laplace method, but essentially the same approach is named Gaussian approximation by Rue et al. (2009) in their Integrated nested Laplace approximation (INLA) software package for Gaussian Markov random field models (Vanhatalo et al., 2009), (see also Tierney and Kadane, 1986).

The implementation of the Laplace algorithm for this particular model requires care since the Student-t likelihood is not log-concave and thus $p(\mathbf{f}|\mathcal{D},\theta,\sigma^2,\nu)$ may be multimodal and some of the $\mathbf{W}_{ii}$ negative. It follows that the standard implementation presented by Rasmussen and Williams (2006) requires some modifications in determining the mode $\hat{\mathbf{f}}$ and the covariance $\boldsymbol{\Sigma}_{\mathrm{LA}}$ which are discussed in detail by Vanhatalo et al. (2009). Later on Hannes Nickisch proposed a slightly different implementation (personal communication) where the stabilized Newton algorithm is used for finding $\hat{\mathbf{f}}$ instead of the EM algorithm and LU decomposition for determining $\boldsymbol{\Sigma}_{\mathrm{LA}}$ instead of rank-1 Cholesky updates (see also Section 4.1). This alternative approach is used at the moment in the GPML software package (Rasmussen and Nickisch, 2010).

3.3 Factorizing Variational Approximation (fVB)

The scale-mixture decomposition (3) enables a computationally convenient variational approximation if the latent values $\mathbf{f}$ and the residual variance terms $\mathbf{V} = [V_1,...,V_n]$ are assumed a posteriori independent:

$$q(\mathbf{f},\mathbf{V}) = q(\mathbf{f})\prod_{i=1}^{n} q(V_i). \tag{5}$$

This kind of factorizing variational approximation was introduced by Tipping and Lawrence (2003) to form a robust observation model for linear models within the relevance vector machine framework. For robust Gaussian process regression with the Student-t model it was applied by Kuss

(2006) and essentially the same variational approach has also been used for approximate inference on linear models with the automatic relevance determination prior (see, e.g., Tipping and Lawrence, 2005). Assuming the factorizing posterior (5) and minimizing the KL-divergence from $q(\mathbf{f},\mathbf{V})$ to the true posterior $p(\mathbf{f},\mathbf{V}|\mathcal{D},\theta,\sigma^2,\nu)$ results in a Gaussian approximation for the latent values, and inverse-χ^2 (or equivalently inverse gamma) approximations for the residual variances V_i. The parameters of $q(\mathbf{f})$ and $q(V_i)$ can be estimated by maximizing a variational lower bound for the marginal likelihood $p(\mathbf{y}|X,\theta,\sigma^2,\nu)$ with an expectation maximization (EM) algorithm. In the E-step of the algorithm the lower bound is maximized with respect to $q(\mathbf{f})$ and $q(V_i)$ given the current point estimate of the hyperparameters and in the M-step a new estimate of the hyperparameters is determined with fixed $q(\mathbf{f})$ and $q(V_i)$.

The drawback with a factorizing approximation determined by minimizing the reverse KL-divergence is that it tends to underestimate the posterior uncertainties (see, e.g., Bishop, 2006). Vanhatalo et al. (2009) compared fVB with the previously described Laplace and MCMC approximations, and found that fVB provided worse predictive variance estimates compared to the Laplace approximation. In addition, the estimation of ν based on maximizing the variational lower bound was found less robust with fVB.

3.4 Variational Bounds (VB)

This variational bounding method was introduced for binary GP classification by Gibbs and MacKay (2000) and comparisons to other approximative methods for GP classification were done by Nickisch and Rasmussen (2008). The method is based on forming a Gaussian lower bound for each likelihood term independently:

$$p(y_i|f_i) \geq \exp(-f_i^2/(2\gamma_i) + b_i f_i - h(\gamma_i)/2),$$

which can be used to construct a lower bound on the marginal likelihood: $p(\mathbf{y}|\mathbf{X},\theta,\nu,\sigma) \geq Z_{\text{VB}}$. With fixed hyperparameters, γ_i and b_i can be determined by maximizing Z_{VB} to obtain a Gaussian approximation for $p(\mathbf{f}|\mathcal{D},\theta,\nu,\sigma^2)$ and an approximation for the marginal likelihood. With the Student-t observation model only the scale parameters γ_i need to be optimized because the location parameter is determined by the corresponding observations: $b_i = y_i/\gamma_i$. Similarly to the Laplace approximation and EP, MAP-estimation of the hyperparameters can be done by optimizing Z_{VB} with gradient-based methods. In our experiments we used the implementation available in the GPML-package (Rasmussen and Nickisch, 2010) augmented with the same hyperprior definitions as with the other approximative methods.

3.5 Expectation Propagation

The EP algorithm is a general method for approximating integrals over functions that factor into simple terms (Minka, 2001a). It approximates the conditional posterior with

$$q(\mathbf{f}|\mathcal{D},\theta,\sigma^2,\nu) = \frac{1}{Z_{\text{EP}}} p(\mathbf{f}|\theta) \prod_{i=1}^{n} \tilde{t}_i(f_i|\tilde{Z}_i,\tilde{\mu}_i,\tilde{\sigma}_i^2) = \mathcal{N}(\boldsymbol{\mu},\boldsymbol{\Sigma}), \tag{6}$$

where $Z_{\text{EP}} \approx p(\mathbf{y}|\mathbf{X},\theta,\sigma^2,\nu)$, and the parameters of the approximate conditional posterior distribution are given by $\boldsymbol{\Sigma} = (\mathbf{K}^{-1} + \tilde{\boldsymbol{\Sigma}}^{-1})^{-1}$, $\boldsymbol{\mu} = \boldsymbol{\Sigma}\tilde{\boldsymbol{\Sigma}}^{-1}\tilde{\boldsymbol{\mu}}$, $\tilde{\boldsymbol{\Sigma}} = \text{diag}[\tilde{\sigma}_1^2,...,\tilde{\sigma}_n^2]$, and $\tilde{\boldsymbol{\mu}} = [\tilde{\mu}_1,...,\tilde{\mu}_n]^{\text{T}}$. In Equation (6) the likelihood terms $p(y_i|f_i,\sigma^2,\nu)$ are approximated by un-normalized Gaussian site functions $\tilde{t}_i(f_i|\tilde{Z}_i,\tilde{\mu}_i,\tilde{\sigma}_i^2) = \tilde{Z}_i\mathcal{N}(f_i|\tilde{\mu}_i,\tilde{\sigma}_i^2)$.

The EP algorithm updates the site parameters $\tilde{Z}_i, \tilde{\mu}_i$ and $\tilde{\sigma}_i^2$ and the posterior approximation (6) sequentially. At each iteration (i), first the i'th site is removed from the i'th marginal posterior to obtain a cavity distribution

$$q_{-i}(f_i) \propto q(f_i|\mathcal{D},\theta,\sigma^2,\nu)\tilde{t}_i(f_i)^{-1}.$$

Then the i'th site is replaced with the exact likelihood term to form a tilted distribution $\hat{p}_i(f_i) = \hat{Z}_i^{-1} q_{-i}(f_i)p(y_i|f_i)$ which is a more refined non-Gaussian approximation to the true i'th marginal distribution. Next the algorithm attempts to match the approximative posterior marginal $q(f_i) = q(f_i|\mathcal{D},\theta,\sigma^2,\nu)$ with $\hat{p}_i(f_i)$ by finding first a Gaussian $\hat{q}_i(f_i)$ satisfying

$$\hat{q}_i(f_i) = \mathcal{N}(f_i|\hat{\mu}_i,\hat{\sigma}_i^2) = \arg\min_{q_i} \mathrm{KL}\left(\hat{p}_i(f_i)||q_i(f_i)\right),$$

which is equivalent to matching $\hat{\mu}_i$ and $\hat{\sigma}_i^2$ with the mean and variance of $\hat{p}_i(f_i)$. Then the parameters of the local approximation $\tilde{t}_i$ are updated so that the moments of $q(f_i)$ match with $\hat{q}_i(f_i)$:

$$q(f_i|\mathcal{D},\theta,\sigma^2,\nu) \propto q_{-i}(f_i)\tilde{t}_i(f_i) \equiv \hat{Z}_i\mathcal{N}(f_i|\hat{\mu}_i,\hat{\sigma}_i^2). \tag{7}$$

Finally, the parameters $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$ of the approximate posterior (6) are updated according to the changes in site $\tilde{t}_i$. These steps are repeated for all the sites at some order until convergence. Since only the means and variances are needed in the Gaussian moment matching only $\tilde{\mu}_i$ and $\tilde{\sigma}_i^2$ need to be updated during the iterations. The normalization terms $\tilde{Z}_i$ are required for the marginal likelihood approximation $Z_{\mathrm{EP}} \approx p(\mathbf{y}|\mathbf{X},\theta,\sigma^2,\nu)$ which is computed after convergence of the algorithm, and they can be determined by integrating over f_i in Equation (7) which gives $\tilde{Z}_i = \hat{Z}_i(\int q_{-i}(f_i)\mathcal{N}(f_i|\tilde{\mu}_i,\tilde{\sigma}_i^2)df_i)^{-1}$.

In the traditional EP algorithm (from now on referred to as sequential EP), the posterior approximation (6) is updated sequentially after each moment matching (7). Recently an alternative parallel update scheme has been used especially in models with a very large number of unknowns (see, e.g., van Gerven et al., 2009). In parallel EP the site updates are calculated with fixed posterior marginals $\boldsymbol{\mu}$ and $\mathrm{diag}(\boldsymbol{\Sigma})$ for all $\tilde{t}_i$, $i = 1,...,n$, in parallel, and the posterior approximation is refreshed only after all the sites have been updated. Although the theoretical cost for one sweep over the sites is the same ($O(n^3)$) for both sequential and parallel EP, in practice one re-computation of $\boldsymbol{\Sigma}$ using the Cholesky decomposition is much more efficient than n sequential rank-one updates. In our experiments, the number of sweeps required for convergence was roughly the same for both schemes in easier cases where standard EP converges.

The marginal likelihood approximation is given by

$$\log Z_{\mathrm{EP}} = -\frac{1}{2}\log|\mathbf{K}+\tilde{\boldsymbol{\Sigma}}| - \frac{1}{2}\tilde{\boldsymbol{\mu}}^{\mathrm{T}}\left(\mathbf{K}+\tilde{\boldsymbol{\Sigma}}\right)^{-1}\tilde{\boldsymbol{\mu}} + \sum_{i=1}^{n}\log\hat{Z}_i(\sigma^2,\nu) + C_{\mathrm{EP}}, \tag{8}$$

where $C_{\mathrm{EP}} = -\frac{n}{2}\log(2\pi) - \sum_i \log\int q_{-i}(f_i)\mathcal{N}(f_i|\tilde{\mu}_i,\tilde{\sigma}_i^2)df_i$ collects terms that are not explicit functions of θ, σ^2 or ν. If the algorithm has converged, that is, $\hat{p}_i(f_i)$ is consistent (has the same means and variances) with $q(f_i)$ for all sites, C_{EP}, $\tilde{\boldsymbol{\Sigma}}$ and $\tilde{\boldsymbol{\mu}}$ can be considered constants when differentiating (8) with respect to the hyperparameters (Seeger, 2005; Opper and Winther, 2005). This enables efficient MAP estimation with gradient based optimization methods.

There is no guarantee of convergence for either sequential or parallel EP. When the likelihood terms are log-concave and the approximation is initialized to the prior, the algorithm converges

fine in many cases (see, e.g., Nickisch and Rasmussen, 2008). However, in case of a non-log-concave likelihood such as the Student-t likelihood, convergence problems may arise and these will be discussed in Section 5. The convergence can be improved either by damping the EP updates (Minka and Lafferty, 2002) or by using a robust but slower double-loop algorithm (Heskes and Zoeter, 2002). In damping, the site parameters in their natural exponential forms, $\tilde{\tau}_i = \tilde{\sigma}_i^{-2}$ and $\tilde{\nu}_i = \tilde{\sigma}_i^{-2}\tilde{\mu}_i$, are updated to a convex combination of the old and proposed new values, which results in the following update rules:

$$\Delta\tilde{\tau}_i = \delta(\hat{\sigma}_i^{-2} - \sigma_i^{-2}) \quad \text{and} \quad \Delta\tilde{\nu}_i = \delta(\hat{\sigma}_i^{-2}\hat{\mu}_i - \sigma_i^{-2}\mu_i), \tag{9}$$

where μ_i and σ_i^2 are the mean and variance of $q(f_i|\mathcal{D},\theta,\sigma^2,\nu)$, and $\delta \in (0,1]$ is a step size parameter controlling the amount of damping. Damping can be viewed as using a smaller step size within a gradient-based search for saddle points of the same objective function as is used in the double-loop algorithm (Heskes and Zoeter, 2002).

3.6 Expectation Propagation, the Double-Loop Algorithm

When either sequential or parallel EP does not converge one may still find approximations satisfying the moment matching conditions (7) by a double loop algorithm. For example, Heskes and Zoeter (2002) present simulation results with linear dynamical systems where the double loop algorithm is able to find useful approximations when damped EP fails to converge. For the model under consideration, the fixed points of the EP algorithm correspond to the stationary points of the following objective function (Minka, 2001b; Opper and Winther, 2005)

$$\begin{aligned}\min_{\boldsymbol{\lambda}_s}\max_{\boldsymbol{\lambda}_-} &-\sum_{i=1}^{n}\log\int p(y_i|f_i)\exp\left(\nu_{-i}f_i-\tau_{-i}\frac{f_i^2}{2}\right)df_i-\log\int p(\mathbf{f})\prod_{i=1}^{n}\exp\left(\tilde{\nu}_i f_i-\tilde{\tau}_i\frac{f_i^2}{2}\right)d\mathbf{f}\\ &+\sum_{i=1}^{n}\log\int\exp\left(\nu_{s_i}f_i-\tau_{s_i}\frac{f_i^2}{2}\right)df_i\end{aligned} \tag{10}$$

where $\boldsymbol{\lambda}_- = \{\nu_{-i},\tau_{-i}\}$, $\tilde{\boldsymbol{\lambda}} = \{\tilde{\nu}_i,\tilde{\tau}_i\}$, and $\boldsymbol{\lambda}_s = \{\nu_{s_i},\tau_{s_i}\}$ are the natural parameters of the cavity distributions $q_{-i}(f_i)$, the site approximations $\tilde{t}_i(f_i)$, and approximate marginal distributions $q_{s_i}(f_i) = \mathcal{N}(\tau_{s_i}^{-1}\nu_{s_i},\tau_{s_i}^{-1})$ respectively. The min-max problem needs to be solved subject to the constraints $\tilde{\nu}_i = \nu_{s_i} - \nu_{-i}$ and $\tilde{\tau}_i = \tau_{s_i} - \tau_{-i}$, which resemble the moment matching conditions in (7). The objective function in (10) is equal to $-\log Z_{\text{EP}}$ defined in (6) and is also equivalent to the expectation consistent (EC) free energy approximation presented by Opper and Winther (2005). A unifying view of the EC and EP approximations as well as the connection to the Bethe free energies is presented by Heskes et al. (2005).

Equation (10) suggests a double-loop algorithm where the inner loop consist of maximization with respect to $\boldsymbol{\lambda}_-$ with fixed $\boldsymbol{\lambda}_s$ and the outer loop of minimization with respect to $\boldsymbol{\lambda}_s$. The inner maximization affects only the first two terms and ensures that the marginal moments of the current posterior approximation $q(\mathbf{f})$ are equal to the moments of the tilted distributions $\hat{p}_i(f_i)$ for fixed $\boldsymbol{\lambda}_s$. The outer minimization ensures that the moments $q_{s_i}(f_i)$ are equal to marginal moments of $q(\mathbf{f})$. At the convergence, $q(f_i)$, $\hat{p}_i(f_i)$, and $q_{s_i}(f_i)$ share the same moments up to the second order. If $p(y_i|f_i)$ are bounded, the objective is bounded from below and consequently there exists stationary points satisfying these expectation consistency constraints (Minka, 2001b; Opper and Winther, 2005). In the case of multiple stationary points the solution with the smallest free energy can be chosen.

Since the first two terms in (10) are concave functions of $\boldsymbol{\lambda}_-$ and $\tilde{\boldsymbol{\lambda}}$ the inner maximization problem is concave with respect to $\boldsymbol{\lambda}_-$ (or equivalently $\tilde{\boldsymbol{\lambda}}$) after substitution of the constraints $\tilde{\boldsymbol{\lambda}} = \boldsymbol{\lambda}_{s_i} - \boldsymbol{\lambda}_-$ (Opper and Winther, 2005). The Hessian of the first term with respect to $\boldsymbol{\lambda}_-$ is well defined (and negative semi-definite) only if the tilted distributions $\hat{p}_i(f_i) \propto p(y_i|f_i)q_{-i}(f_i)$ are proper probability distributions with finite moments up to the fourth order. Therefore, to ensure that the product of $q_{-i}(f_i)$ and the Student-t site $p(y_i|f_i)$ has finite moments and that the inner-loop moment matching remains meaningful, the cavity precisions τ_{-i} have to be kept positive. Furthermore, since the cavity distributions can be regarded as estimates for the leave-one-out (LOO) distributions of the latent values, $\tau_{-i} = 0$ would correspond to a situation where $q(f_i|\mathbf{y}_{-i}, \mathbf{X})$ has infinite variance, which does not make sense given the Gaussian prior assumption (1). On the other hand, $\tilde{\tau}_i$ may become negative for example when the corresponding observation y_i is an outlier (see Section 5).

3.7 Fractional EP Updates

Fractional EP (or power EP, Minka, 2004) is an extension of EP which can be used to reduce the computational complexity of the algorithm by simplifying the tilted moment evaluations and to improve the robustness of the algorithm when the approximation family is not flexible enough (Minka, 2005) or when the propagation of information is difficult due to vague prior information (Seeger, 2008). In fractional EP the cavity distributions are defined as $q_{-i}(f_i) \propto q(f_i|\mathcal{D}, \theta, \nu, \sigma^2)/\tilde{t}_i(f_i)^\eta$ and the tilted distribution as $\hat{p}_i(f_i) \propto q_{-i}(f_i)p(y_i|f_i)^\eta$ for a fraction parameter $\eta \in (0, 1]$. The site parameters are updated so that the moments of $q_{-i}(f_i)\tilde{t}_i(f_i)^\eta \propto q(f_i)$ match with $q_{-i}(f_i)p(y_i|f_i)^\eta$. Otherwise the procedure is similar and standard EP can be recovered by setting $\eta = 1$. In fractional EP the natural parameters of the cavity distribution are given by

$$\tau_{-i} = \sigma_i^{-2} - \eta\tilde{\tau}_i \quad \text{and} \quad \nu_{-i} = \sigma_i^{-2}\mu_i - \eta\tilde{\nu}_i, \tag{11}$$

and the site updates (with damping factor δ) by

$$\Delta\tilde{\tau}_i = \delta\eta^{-1}(\hat{\sigma}_i^{-2} - \sigma_i^{-2}) \quad \text{and} \quad \Delta\tilde{\nu}_i = \delta\eta^{-1}(\hat{\sigma}_i^{-2}\hat{\mu}_i - \sigma_i^{-2}\mu_i). \tag{12}$$

The fractional update step $\min_q \mathrm{KL}(\hat{p}_i(f_i)||q(f_i))$ can be viewed as minimization of the α-divergence with $\alpha = \eta$ (Minka, 2005). Compared to the KL-divergence, minimizing the α-divergence with $0 < \alpha < 1$ does not force $q(f_i)$ to cover as much of the probability mass of $\hat{p}_i(f_i)$ whenever $\hat{p}_i(f_i) > 0$. As a consequence, fractional EP tends to underestimate the variance and normalization constant of $q_{-i}(f_i)p(y_i|f_i)^\eta$, and also the approximate marginal likelihood Z_{EP}. On the other hand, we also found that minimizing the KL-divergence in standard EP may overestimate the marginal likelihood with some data sets. In case of multiple modes, the approximation tries to represent the overall uncertainty in $\hat{p}_i(f_i)$ the more exactly the closer α is to 1. In the limit $\alpha \to 0$ the reverse KL-divergence is obtained which is used in some form, for example, in the fVB and KL approximations (Nickisch and Rasmussen, 2008). Also the double-loop objective function (10) can be modified according to the different divergence measure of fractional EP (Cseke and Heskes, 2011; Seeger and Nickisch, 2011).

Fractional EP has some benefits over standard EP with the non-log-concave Student-t sites. First, when evaluating the moments of $q_{-i}(f_i)p(y_i|f_i)^\eta$, setting $\eta < 1$ flattens the likelihood term which alleviates the possible converge problems related to multimodality. This is related to the approximating family being too inflexible and the benefits of different divergence measures in these

cases are considered by Minka (2005). Second, the fractional updates help to avoid the cavity precisions becoming too small, or even negative. Equation (11) shows that by choosing $\eta < 1$, a fraction $(1-\eta)$ of the precision $\tilde{\tau}_i$ of the i:th site is left in the cavity. This decreases the cavity variances which in turn makes the tilted moment integrations and the subsequent EP updates (12) more robust. Problems related to cavity precision becoming too small can be present also with log-concave sites when the prior information is vague. For example, Seeger (2008) reports that with an underdetermined linear model combined with a log-concave Laplace prior the cavity precisions remain positive but they may become very small which induces numerical inaccuracies in the analytical moment evaluations. These inaccuracies may accumulate and even cause convergence problems. Seeger (2008) reports that fractional updates improve numerical robustness and convergence in such cases.

4. Robust Implementation of the Parallel EP Algorithm

The sequential EP updates are shown to be stable for models in which the exact site terms (in our case the likelihood functions $p(y_i|f_i)$) are log-concave (Seeger, 2008). In this case, all site variances, if initialized to non-negative values, remain non-negative during the updates. It follows that the variances of the cavity distributions $q_{-i}(f_i)$ are positive and thus also the subsequent moment evaluations of $q_{-i}(f_i)p(y_i|f_i)$ are numerically robust. The non-log-concave Student-t likelihood is problematic because both the conditional posterior $p(\mathbf{f}|\mathcal{D},\theta,\nu,\sigma)$ as well as the tilted distributions $\hat{p}_i(f_i)$ may become multimodal. Therefore extra care is needed in the implementation and these issues are discussed in this section.

The double-loop algorithm is a rigorous approach that is guaranteed to converge to a stationary point of the objective function (10) when the site terms $p(y_i|f_i)$ are bounded from below. The downside is that the double-loop algorithm can be much slower than for example parallel EP because it spends much computational effort during the inner loop iterations, especially in the early stages when $q_{s_i}(f_i)$ are poor approximations for the true marginals. An obvious improvement would be to start with damped parallel updates and to continue with the double-loop method if necessary. Since in our experiments parallel EP has proven quite efficient with many easier data sets, we adopt this approach and propose few modifications to improve the convergence in difficult cases. A parallel EP initialization and a double-loop backup is also used by Seeger and Nickisch (2011) in their fast EP algorithm.

Parallel EP can also be interpreted as a variant of the double-loop algorithm where only one inner-loop optimization step is done by moment matching (7) and each such update is followed by an outer-loop refinement of the marginal approximations $q_{s_i}(f_i)$. The inner-loop step consists of evaluating the tilted moments $\{\hat{\mu}_i, \hat{\sigma}_i^2 | i = 1,\ldots,n\}$ with $q_{s_i}(f_i) = q(f_i) = \mathcal{N}(\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_{ii})$, updating the sites (9), and updating the posterior (6). The outer-loop step consists of setting $q_{s_i}(f_i)$ equal to the new marginal distributions $q(f_i)$. Connections between the message passing updates and the double-loop methods together with considerations of different search directions for the inner-loop optimization can be found in the extended version of Heskes and Zoeter (2002). The robustness of parallel EP can be improved by the following modifications.

1. After each moment matching step check that the objective (10) increases. If the objective does not increase, decrease the damping coefficient δ until increase is obtained. The downside is that this requires one additional evaluation of the tilted moments for every site per iteration,

but if these one-dimensional integrals are implemented efficiently this is a reasonable price for stability.

2. Before updating the sites (9) check that the new cavity variances $\tau_{-i} = \tau_{s_i} - (\tilde{\tau}_i + \Delta\tilde{\tau}_i)$ are positive. If they are negative, choose a smaller damping factor δ so that $\tau_{-i} > 0$. This computationally cheap precaution ensures that the increase of the objective (10) can be verified according to modification 1.

3. With modifications 1 and 2 the site parameters can still oscillate (see Section 5 for an illustration) but according to our experiments the convergence is obtained with all hyperparameters values eventually. The oscillations can be reduced by updating $q_{s_i}(f_i)$ only after the moments of $\hat{p}_i(f_i)$ and $q(f_i)$ are consistent for all $i = 1,...,n$ with some small tolerance, for example 10^{-4}. At each update, check also that the new cavity precisions are positive, and if not, continue the inner-loop iterations with the previous $q_{s_i}(f_i)$ until better moment consistency is achieved or switch to fractional updates. Actually, this modification corresponds to the maximization in (10) and it results in a double-loop algorithm where the inner-loop optimization is done by moment matching (7). If no parallel initialization is done, often during the first 5-10 iterations when the step size δ is limited according to modification 2, the consistency between $\hat{p}_i(f_i)$ and $q(f_i)$ cannot be achieved. This is an indication of $q(\mathbf{f})$ being a too inflexible approximation for the tilted distributions with the current $q_{s_i}(f_i)$. An outer-loop update $q_{s_i}(f_i) = q(f_i)$ usually helps in these cases.

4. If sufficient increase of the objective is not achieved after an inner-loop update (modification 1), use the gradient information to obtain a better step size δ. The gradients of (10) with respect to the site parameters $\tilde{\nu}_i$ and $\tilde{\tau}_i$ can be calculated without additional evaluations of the objective function for fixed $\boldsymbol{\lambda}_s$. With these gradients, it is possible to determine $g(\delta)$, the gradient of the inner-loop objective function with respect to δ in the current search direction. For parallel EP the search direction is defined by (9) with fixed site updates $\Delta\tilde{\tau}_i = \hat{\sigma}_i^{-2} - \sigma_i^{-2}$ and $\Delta\tilde{\nu}_i = \hat{\sigma}_i^{-2}\hat{\mu}_i - \sigma_i^{-2}\mu_i$ for $i = 1,...,n$. In case of a too large step, $g(\delta)$ becomes negative. Then, for example, spline interpolation with derivative constraints at the end points can be used to approximate the objective as a function of δ. From this approximation a better estimate for the step size δ can be determined efficiently. In case of a too short step, $g(\delta)$ becomes positive and a better step size can be obtained by extrapolating with constraints based on approximate second order derivatives. This modification corresponds to an approximative line search in the concave inner-loop maximization.

In the comparisons of Section 6 we start with 10 damped ($\delta = 0.8$) parallel iterations because with a sensible hyperparameter initialization this is enough to achieve convergence in most hyperparameter optimization steps with the empirical data sets. If no convergence is achieved this parallel initialization also speeds up the convergence of the subsequent double-loop iterations (see Section 5.3). If after any of the initial parallel updates the posterior covariance $\boldsymbol{\Sigma}$ becomes ill-conditioned, that is, many of the $\tilde{\tau}_i$ are too negative, or any of the cavity variances become negative we reject the new site configuration and proceed with more robust updates using the previously described modifications. To reduce the computational costs we limited the maximum number of inner loop iterations (modification 3) to two with two possible additional step size adjustment iterations (modification 4). This may not be enough to suppress all oscillations of the site parameters but in practice more

frequent outer loop refinements of $q_{s_i}(f_i)$ were found to require fewer computationally expensive objective evaluations for convergence.

In some rare cases, for example, when the noise level σ is very small, the outer-loop update of $q_{s_i}(f_i)$ may result in negative values for some of the cavity variances even though the inner-loop optimality is satisfied. In practise this means that $[\mathbf{\Sigma}_{ii}]^{-1}$ is smaller than $\tilde{\tau}_i$ for some i. This may be a numerical problem or an indication of a too inflexible approximating family but switching to fractional updates helps. However, in our experiments, this happened only when the noise level was set to too small values and with a sensible hyperparameter initialization such problems did not emerge.

4.1 Other Implementation Details

The EP updates require evaluation of moments $m_k = \int f_i^k g_i(f_i) df_i$ for $k = 0, 1, 2$, where we have defined $g_i(f_i) = q_{-i}(f_i) p(y_i|f_i)^\eta$. With the Student-$t$ likelihood and an arbitrary $\eta \in (0, 1]$ numerical integration is required. Instead of the standard Gauss quadrature we used the adaptive Gauss-Kronrod quadrature described by Shampine (2008) because it can save function evaluations by reusing the existing nodes during the adaptive interval subdivisions. For further computational savings all the required moments were calculated simultaneously using the same function evaluations. The integrand $g_i(f_i)$ may have one or two modes between the cavity mean μ_{-i} and the observation y_i. In the two-modal case the first mode is near μ_{-i} and the other near $\mu_\infty = \sigma_\infty^2(\sigma_{-i}^{-2}\mu_{-i} + \eta_i\sigma^{-2}y_i)$, where μ_∞ and $\sigma_\infty^2 = (\sigma_{-i}^{-2} + \eta_i\sigma^{-2})^{-1}$ correspond to the mean and variance of the limiting Gaussian tilted distribution as $\nu \to \infty$. The integration limits were set to $\min(\mu_{-i} - 6\sigma_{-i}, \mu_\infty - 10\sigma_\infty)$ and $\max(\mu_{-i} + 6\sigma_{-i}, \mu_\infty + 10\sigma_\infty)$ to cover all the relevant mass around the both possible modes.

Both the hyperparameter estimation and monitoring the convergence of EP requires that the marginal likelihood $q(\mathbf{y}|\mathbf{X}, \theta, \sigma^2, \nu)$ can be evaluated in a numerically robust manner. Assuming a fraction parameter η the marginal likelihood is given by

$$\log Z_{\text{EP}} = \frac{1}{\eta}\sum_{i=1}^{n}\left(\log \hat{Z}_i + \frac{1}{2}\log \tau_{s_i}\tau_{-i}^{-1} + \frac{1}{2}\tau_{-i}^{-1}\nu_{-i}^2 - \frac{1}{2}\tau_{s_i}^{-1}\nu_{s_i}^2\right) - \frac{1}{2}\log|\mathbf{I} + \mathbf{K}\tilde{\mathbf{\Sigma}}^{-1}| - \frac{1}{2}\tilde{\boldsymbol{\nu}}^{\mathrm{T}}\boldsymbol{\mu},$$

where $\nu_{s_i} = \nu_{-i} + \eta\tilde{\nu}_i$ and $\tau_{s_i} = \tau_{-i} + \eta\tilde{\tau}_i$. The first sum term can be evaluated safely if the cavity precisions τ_{-i} and the tilted variances $\hat{\sigma}_i^2$ remain positive during the EP updates because at convergence $\tau_{s_i} = \hat{\sigma}_i^{-2}$.

Evaluation of $|\mathbf{I} + \mathbf{K}\tilde{\mathbf{\Sigma}}^{-1}|$ and $\mathbf{\Sigma} = (\mathbf{K}^{-1} + \tilde{\mathbf{\Sigma}}^{-1})^{-1}$ needs some care because many of the diagonal entries of $\tilde{\mathbf{\Sigma}}^{-1} = \text{diag}[\tilde{\tau}_1, ..., \tilde{\tau}_n]$ may become negative due to outliers and thus the standard approach presented by Rasmussen and Williams (2006) is not suitable. One option is to use the rank one Cholesky updates as described by Vanhatalo et al. (2009) or the LU decomposition as is done in the GPML implementation of the Laplace approximation (Rasmussen and Nickisch, 2010). In our parallel EP implementation we process the positive and negative sites separately. We define $\mathbf{W}_1 = \text{diag}(\tilde{\tau}_i^{1/2})$ for $\tilde{\tau}_i \geq 0$ and $\mathbf{W}_2 = \text{diag}(|\tilde{\tau}_i|^{1/2})$ for $\tilde{\tau}_i < 0$, and divide $\mathbf{K}$ into corresponding blocks $\mathbf{K}_{11}$, $\mathbf{K}_{22}$, and $\mathbf{K}_{12} = \mathbf{K}_{21}^{\mathrm{T}}$. We compute the Cholesky decompositions of two symmetric matrices

$$\mathbf{L}_1\mathbf{L}_1^{\mathrm{T}} = \mathbf{I} + \mathbf{W}_1\mathbf{K}_{11}\mathbf{W}_1 \quad \text{and} \quad \mathbf{L}_2\mathbf{L}_2^{\mathrm{T}} = \mathbf{I} - \mathbf{W}_2(\mathbf{K}_{22} - \mathbf{U}_2\mathbf{U}_2^{\mathrm{T}})\mathbf{W}_2,$$

where $\mathbf{U}_2 = \mathbf{K}_{21}\mathbf{W}_1\mathbf{L}_1^{-\mathrm{T}}$. The required determinant is given by $|\mathbf{I} + \mathbf{K}\tilde{\mathbf{\Sigma}}^{-1}| = |\mathbf{L}_1|^2|\mathbf{L}_2|^2$. The dimension of $\mathbf{L}_1$ is typically much larger than that of $\mathbf{L}_2$ and it is always positive definite. $\mathbf{L}_2$ may not

be positive definite if the site precisions have too small negative values, and therefore if the second Cholesky decomposition fails after a parallel EP update we reject the proposed site parameters and reduce the step size. The posterior covariance can be evaluated as $\mathbf{\Sigma} = \mathbf{K} - \mathbf{U}\mathbf{U}^{\mathrm{T}} + \mathbf{V}\mathbf{V}^{\mathrm{T}}$, where $\mathbf{U} = [\mathbf{K}_{11}, \mathbf{K}_{12}]^{\mathrm{T}}\mathbf{W}_1\mathbf{L}_1^{-\mathrm{T}}$ and $\mathbf{V} = [\mathbf{K}_{21}, \mathbf{K}_{22}]^{\mathrm{T}}\mathbf{W}_2\mathbf{L}_2^{-\mathrm{T}} - \mathbf{U}\mathbf{U}_2^{\mathrm{T}}\mathbf{W}_2\mathbf{L}_2^{-\mathrm{T}}$. The regular observations reduce the posterior uncertainty through $\mathbf{U}$ and the outliers increase uncertainty through $\mathbf{V}$.

5. Properties of EP with a Student-t Likelihood

In GP regression the outlier rejection property of the Student-*t* model depends heavily on the data and the hyperparameters. If the hyperparameters and the resulting unimodal approximation (6) are suitable for the data there are usually only a few outliers and there is enough information to handle them given the smoothness assumptions of the GP prior and the regular observations. This is usually the case during the MAP estimation if the hyperparameters are initialized sensibly. On the other hand, unsuitable hyperparameters may produce a very large number of outliers and also considerable uncertainty on whether certain data points are outliers or not. For example, a small ν combined with a too small σ and a too large lengthscale (i.e., a too inflexible model) can result into a very large number of outliers because the model is unable to explain large quantity of the observations. Unsuitable hyperparameters may not necessarily induce convergence problems for EP if there exists only one plausible posterior hypothesis capable of handling the outliers. However, if the conditional posterior distribution has multiple modes, convergence problems may occur unless sufficient amount of damping is used. In some difficult cases either fractional updates or double-loop iterations may be needed to achieve convergence. In this section we discuss the convergence properties of EP with the Student-*t* likelihood, demonstrate the effects of the different EP modifications described in the sections 3 and 4, and also compare the quality of the EP approximation to the other methods described in Section 3 with the help of simple regression examples.

An outlying observation y_i increases the posterior uncertainty on the unknown function at the input space regions a priori correlated with $\mathbf{x}_i$. The amount of increase depends on how far the posterior mean estimate of the unknown function value, $\mathrm{E}(f_i|\mathcal{D})$, is from the observation y_i. Some insight into this behavior is obtained by considering the negative Hessian of $\log p(y_i|f_i, \nu, \sigma^2)$, that is, $W_i = -\nabla^2_{f_i} \log p(y_i|f_i)$, as a function of f_i (compare to the Laplace approximation in Section 3.2). W_i is positive when $y_i - \sigma\sqrt{\nu} < f_i < y_i + \sigma\sqrt{\nu}$, attains its negative minimum when $f_i = y_i \pm \sigma\sqrt{3\nu}$ and approaches zero as $|f_i| \to \infty$. Thus, with the Laplace approximation, y_i satisfying $\hat{f}_i - \sigma\sqrt{\nu} < y_i < \hat{f}_i + \sigma\sqrt{\nu}$ can be interpreted as regular observations because they decrease the posterior covariance $\mathbf{\Sigma}_{\mathrm{LA}}^{-1}$ in Equation (4). The rest of the observations increase the posterior uncertainty and can therefore be interpreted as outliers. Observations that are far from the mode $\hat{f}_i$ are clear outliers in the sense that they have very little effect on the posterior uncertainty. Observations that are close to $\hat{f}_i \pm \sigma\sqrt{3\nu}$ are not clearly outlying because they increase the posterior uncertainty the most. The most problematic situations arise when the hyperparameters are such that many $\hat{f}_i$ are close to $y_i \pm \sigma\sqrt{3\nu}$. However, despite the negative $\mathbf{W}_{ii}$, the covariance matrix $\mathbf{\Sigma}_{LA}$ is positive definite if $\hat{\mathbf{f}}$ is a local maximum of the conditional posterior.

EP behaves similarly as well. If there is a disagreement between the cavity distribution $q_{-i}(f_i) = \mathcal{N}(\mu_{-i}, \sigma^2_{-i})$ and the likelihood $p(y_i|f_i)$ but the observation is not a clear outlier, the uncertainty in the tilted distribution increases towards the observation and the tilted distribution can even become two-modal. The moment matching (7) results in an increase of the marginal posterior variance, $\hat{\sigma}_i^2 > \sigma_i^2$, which causes $\tilde{\tau}_i$ to decrease (9) and possibly to become negative. Sequential EP usually

runs smoothly when all the outliers are clear and $p(\mathbf{f}|\mathcal{D},\theta,\nu,\sigma^2)$ has a unique mode. The site precisions corresponding to the outlying observations may become negative but their absolute values remain small compared to the site precisions of the regular observations. However, if some of the negative sites become very small they may notably decrease the approximate marginal precisions $\tau_i = \sigma_i^{-2}$ of the a priori dependent sites because of the prior correlations defined by $\mathbf{K}$. It follows that the uncertainty in the cavity distributions may increase considerably, that is, the cavity precisions, $\tau_{-i} = \tau_i - \tilde{\tau}_i$, may become very small or negative. This may cause both stability and convergence problems which will be illustrated in the following sections with the help of simple regression examples.

5.1 Simple Regression Examples

Figure 1 shows two one-dimensional regression problems in which standard EP may run into problems. In example 1 (the left subfigures), there are two outliers y_1 and y_2 providing conflicting information in a region with no regular observations ($1 < x < 3$). In this example the posterior mass of the length-scale is concentrated to sufficiently large value so that the GP prior is stiff and keeps the marginal posterior $p(\mathbf{f}|\mathcal{D})$ (shown in the lower left panel) and the conditional posterior $p(\mathbf{f}|\mathcal{D},\hat{\theta},\hat{\nu},\hat{\sigma}^2)$ at the MAP estimate unimodal. Both sequential and parallel EP converge with the MAP estimate for the hyperparameters.

The corresponding predictive distribution is visualized in the upper left panel of Figure 1 showing a considerable increase in the posterior uncertainty when $1 < x < 3$. The lower left panel shows comparison of the predictive distribution of $f(x)$ at $x = 2$ obtained with the different approximations described in Section 3. The hyperparameters are estimated separately for each method. The smooth MCMC estimate of the predictive density of the latent value $f_* = f(x_*)$ at input location x_* is calculated by integrating analytically over $\mathbf{f}$ for each posterior draw of the residual variances $\mathbf{V}$ and averaging the resulting Gaussian distributions $q(f_*|x_*,\mathbf{V},\theta)$. The MCMC estimate (with integration over the hyperparameters) is unimodal but shows small side bumps when the latent function value is close to the observations y_1 and y_2. The standard EP estimate covers well the posterior uncertainty on the latent value but both the Laplace method and fVB underestimate it. At the other input locations where the uncertainty is small, all methods give very similar estimates.

Even though EP remains stable in example 1 with the MAP estimates of the hyperparameters, it is not stable with all hyperparameter values. If ν and σ^2 were sufficiently small, so that the likelihood $p(y_i|f_i)$ was narrow as a function of f_i, and the length-scale was small inducing small correlations between inputs far apart, there would be significant posterior uncertainty about the unknown $f(x)$ when $1 < x < 3$ and the true conditional posterior would be multimodal. Due to the small prior covariances of the observations y_1 and y_2 with the other data points $y_3,...,y_n$, the cavity distributions $q_{-1}(f_1)$ and $q_{-2}(f_2)$ would differ strongly from the approximative marginal posterior distributions $q(f_1)$ and $q(f_2)$. This difference would lead to a very small (or even negative) cavity precisions τ_{-1} and τ_{-2} during the EP iterations which causes stability problems as will be illustrated in section 5.2.

The second one-dimensional regression example, visualized in the upper right panel of Figure 1, is otherwise similar with example 1 except that the nonlinearity of the true function is much stronger when $-5 < x < 0$, and the observations y_1 and y_2 are closer in the input space. The stronger nonlinearity requires a much smaller length-scale for a good data fit and the outliers y_1 and y_2 provide more conflicting information (and stronger multimodality) due to the larger prior covariance. The lower right panel shows comparison of the approximative predictive distributions of $f(x)$ when $x = 2$.

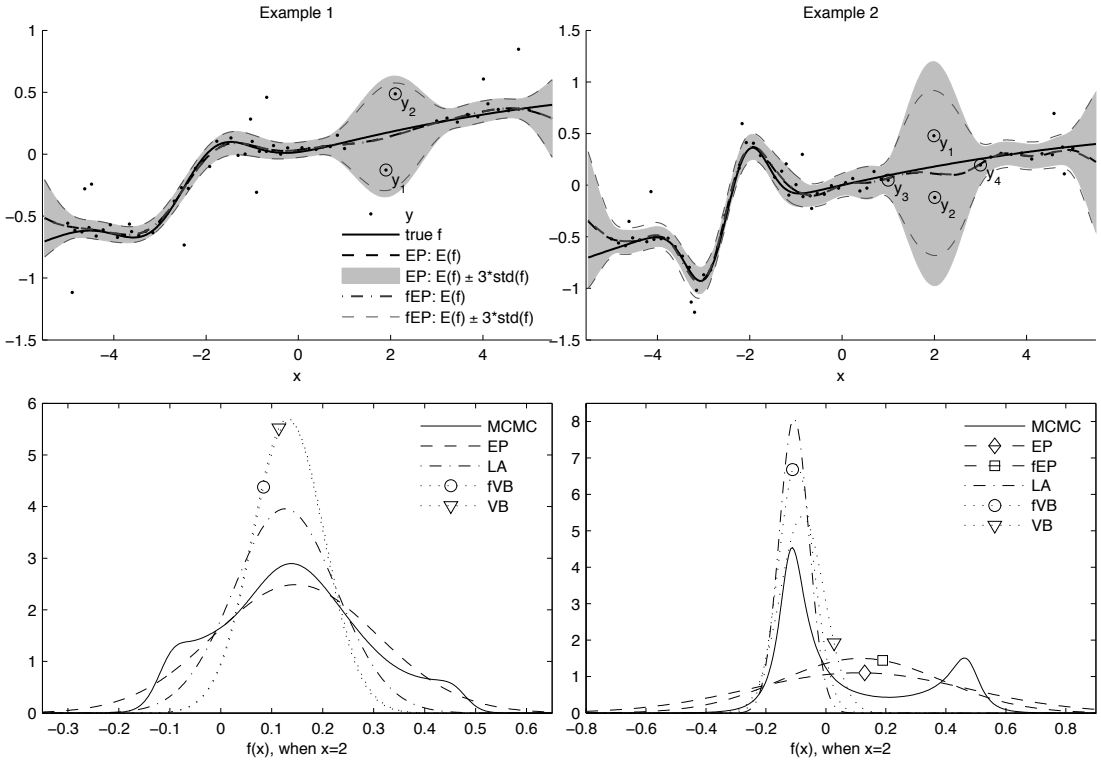
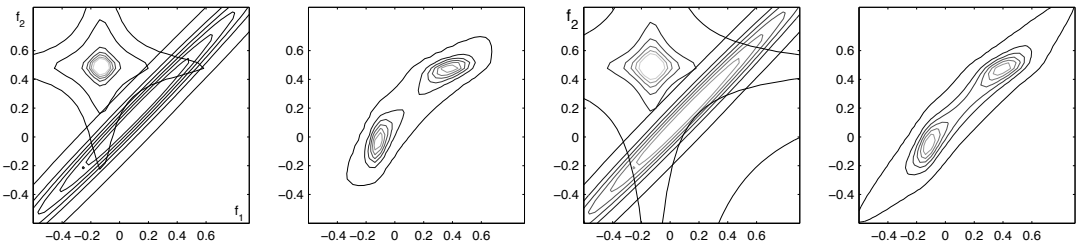


Figure 1: The upper row: Two one-dimensional regression examples, where standard EP may fail to converge with certain hyperparameter values, unless damped sufficiently. The EP approximations obtained by both the regular updates $\eta = 1$ (EP) and the fractional updates $\eta = 0.5$ (fEP) are visualized. The lower row: Comparison of the approximative predictive distributions of the latent value $f(x)$ at $x = 2$. With MCMC all the hyperparameters are sampled and for all the other approximations (except fVB in example 2, see the text for explanation) the hyperparameters are fixed to the corresponding MAP estimates. Notice that the MCMC estimate of the predictive distribution is unimodal in example 1 and multimodal in example 2. With smaller lengthscale values the conditional posterior $p(\mathbf{f}|\mathcal{D},\theta,\nu,\sigma^2)$ can be multimodal also in example 1.

The MCMC estimate has two separate modes near the observations y_1 and y_2. The Laplace and fVB approximations are sharply localized at the mode near y_1 but the standard EP approximation (EP1) is very wide trying to preserve the uncertainty about the both modes. Contrary to example 1, also the conditional posterior $q(\mathbf{f}|\mathcal{D},\theta,\nu,\sigma)$ is two-modal if the hyperparameters are set to their MAP-estimates.

5.2 EP Updates with the Student-*t* Sites

Next we discuss the problems with the standard EP updates with the help of example 1. Figure 2 illustrates a two-dimensional tilted distribution of the latent values f_1 and f_2 related to the observations y_1 and y_2 in example 1. A relatively small lengthscale (0.9) is chosen so that there is



(a) $q(f_1, f_2|y_3, ..., y_n)$ with $p(y_1|f_1)p(y_2|f_2)$ (b) $q(f_1, f_2|y_3, ..., y_n) \times p(y_1|f_1)p(y_2|f_2)$ (c) $q(f_1, f_2|y_3, ..., y_n)$ with $p(y_1|f_1)^\eta p(y_2|f_2)^\eta$ (d) $q(f_1, f_2|y_3, ..., y_n) \times p(y_1|f_1)^\eta p(y_2|f_2)^\eta$

Figure 2: An illustration of a two-dimensional tilted distribution related to the two problematic data points y_1 and y_2 in example 1. Compared to the MAP value used in Figure 1, shorter lengthscale (0.9) is selected so that the true conditional posterior is multimodal. Panel (a) visualizes the joint likelihood $p(y_1|f_1)p(y_2|f_2)$ together with the generalized 2-dimensional cavity distribution $q(f_1, f_2|y_3, ..., y_n)$ obtained by one round of undamped sequential EP updates on sites $\tilde{t}_i(f_i)$, for $i = 3, ..., n$. Panel (b) visualizes the corresponding two-dimensional tilted distribution $\hat{p}_i(f_1, f_2) \propto q(f_1, f_2|y_3, ..., y_n)p(y_1|f_1)p(y_2|f_2)$. Panels (c) and (d) show the same with only a fraction $\eta = 0.5$ of the likelihood terms included in the tilted distribution, which corresponds to fractional EP updates on these sites.

quite strong prior correlation between f_1 and f_2. Suppose that all other sites have already been updated once with undamped sequential EP starting from a zero initialization ($\tilde{\tau}_i = 0$ and $\tilde{\nu}_i = 0$ for $i = 1, ..., n$). Panel (a) visualizes a generalized 2-dimensional cavity distribution $q(f_1, f_2|y_3, \ldots, y_n)$ together with the joint likelihood $p(y_1, y_2|f_1, f_2) = p(y_2|f_2)p(y_2|f_2)$, and panel (b) shows the contours of the resulting two dimensional tilted distribution which has two separate modes. If the site $\tilde{t}_1(f_1)$ is updated next in the sequential manner with no damping, $\tilde{\tau}_1$ will get a large positive value and the approximation $q(f_1, f_2)$ fits tightly around the mode near the observation y_1. After this, when the site $\tilde{t}_2(f_2)$ is updated, it gets a large negative precision, $\tilde{\tau}_2 < 0$, since the approximation needs to be expanded towards the observation y_2. It follows that, the marginal precision of f_1 is updated to a smaller value than $\tilde{\tau}_1$. Therefore, during the second sweep the cavity precision $\tau_{-1} = \sigma_1^{-2} - \tilde{\tau}_1$ becomes negative, and site 1 can no longer be updated. If the EP updates were done in parallel, both the cavity and the site precisions would be positive after the first posterior update, but $q(f_1, f_2)$ would be tightly centered between the modes. After a couple of parallel loops over all the sites, one of the problematic sites gets a too small negative precision because the approximation needs to be expanded to cover all the marginal uncertainty in the tilted distributions which leads to a negative cavity precision for the other site.

Skipping updates on the sites with negative cavity variances can keep the algorithm numerically stable (see, for example, Minka and Lafferty, 2002). Also increasing damping reduces $\Delta\tilde{\tau}_i$ so that the negative cavity precisions are less likely to emerge. However, these modifications are not enough to ensure convergence. After a few EP iterations, the marginal posterior distribution of a problematic site, for instance $q(f_1)$, is centered between the observations (see, for example, Figure 1). At the same time, the respective cavity distribution, $q_{-1}(f_1)$, is centered near the other problematic observation, y_2. Combining such cavity distribution with the likelihood term, $p(y_1|f_1)$, gives a tilted distribution with significant mass around both observations. If the site precisions, $\tilde{\tau}_1$ and $\tilde{\tau}_2$,

are sufficiently large (corresponding to a tight posterior approximation), the variance of the tilted distribution will be larger than that of the marginal posterior and thus the site precision, $\tilde{\tau}_1$ will be decreased. The same happens for the other site. The site precisions are decreased for a few iterations after which the posterior marginals are so wide that the variances of the tilted distributions are smaller than the posterior marginal variances. At this point the site precisions start again to increase gradually. This leads to oscillation between small and large site precisions as illustrated in Figure 3.

With a smaller δ the oscillations are slower and with a sufficiently small δ the amplitude of the oscillations may gradually decrease leading to convergence, as in the panel (b) of Figure 3. However, the convergence is not guaranteed since the conditions of the inner-loop maximization in (10) are not guaranteed to be fulfilled in sequential or parallel EP. For example, a sequential EP update can be considered as a one inner-loop step where only one site is updated, followed by an outer-loop step which updates all the marginal posteriors as $q_{s_i}(f_i) = q(f_i)$. Since the update of one site does not maximize the inner-loop objective, the conditions used to form the upper bound of the convex part in (10) are not met (Opper and Winther, 2005). Therefore, the outer-loop objective is not guaranteed to decrease and the new approximate marginal posteriors may be worse than in the previous iteration.

Example 2 is more difficult in the sense that convergence requires damping at least with $\delta = 0.5$. With sequential EP the convergence depends also on the update order of the sites and $\delta < 0.3$ is needed for convergence with all permutations. Furthermore, if the double-loop approach of Section 4 is considered, the best step size, that minimizes the inner-loop objective in the current search direction, can change (and also increase) considerably between subsequent inner-loop iterations which makes the continuous step-size adjustments very useful.

Also fractional updates improve the stability of EP. Figures 2(c)–(d) illustrate the same approximate tilted distribution as Figures 2(a)–(b) but now only a fraction $\eta = 0.5$ of the likelihood terms are included. This corresponds to the first round fractional updates on these sites with zero initialization. Because of the flattened likelihood $p(y_1|f_1)^\eta p(y_2|f_2)^\eta$ the 2-dimensional tilted distribution is still two-modal but less sharply peaked compared to standard EP on the left. It follows that also the one-dimensional tilted distributions have smaller variances and the consecutive fractional updates (12) of the sites 1 and 2 do not widen the marginal variances σ_1^2 and σ_2^2 as much. This helps to keep the cavity precisions positive by increasing the approximate marginal posterior precisions and reducing the possible negative increments on the site precisions $\tilde{\tau}_1$ and $\tilde{\tau}_2$. This is possible because the different divergence measure allows for a more localized approximation at $1 < x < 3$. In addition, the property that a fraction $(1-\eta)$ of the site precisions is left in the cavity distributions helps to keep the cavity precisions positive during the algorithm. Figure 1 shows a comparison of standard (EP) and fractional EP (fEP, $\eta = 0.5$) with the MAP estimates of the hyperparameters. In example 1 both methods produce very similar predictive distribution because the posterior is unimodal. In example 2 (the lower right panel) fractional EP gives a much smaller predictive uncertainty estimate when $x = 2$ than standard EP which in turn puts more false posterior mass in the tails when compared to MCMC.

The practical guidelines presented in Section 4 bring additional stability in the above described problematic situations. Modification 1 helps to avoid immediate problems from a too large step size by ensuring that each parallel EP update increases the inner-loop objective defined by (10). Modification 2 reduces the step size δ so that the cavity variances, defined as $\tau_{-i} = \tau_{s_i} - \tilde{\tau}_i$ with fixed $\boldsymbol{\lambda}_s = \{\nu_{s_i}, \tau_{s_i}\}$, will remain positive during the inner-loop updates. Modification 3 reduces the oscillations by ensuring that the inner-loop maximization is done within some tolerance, that is,

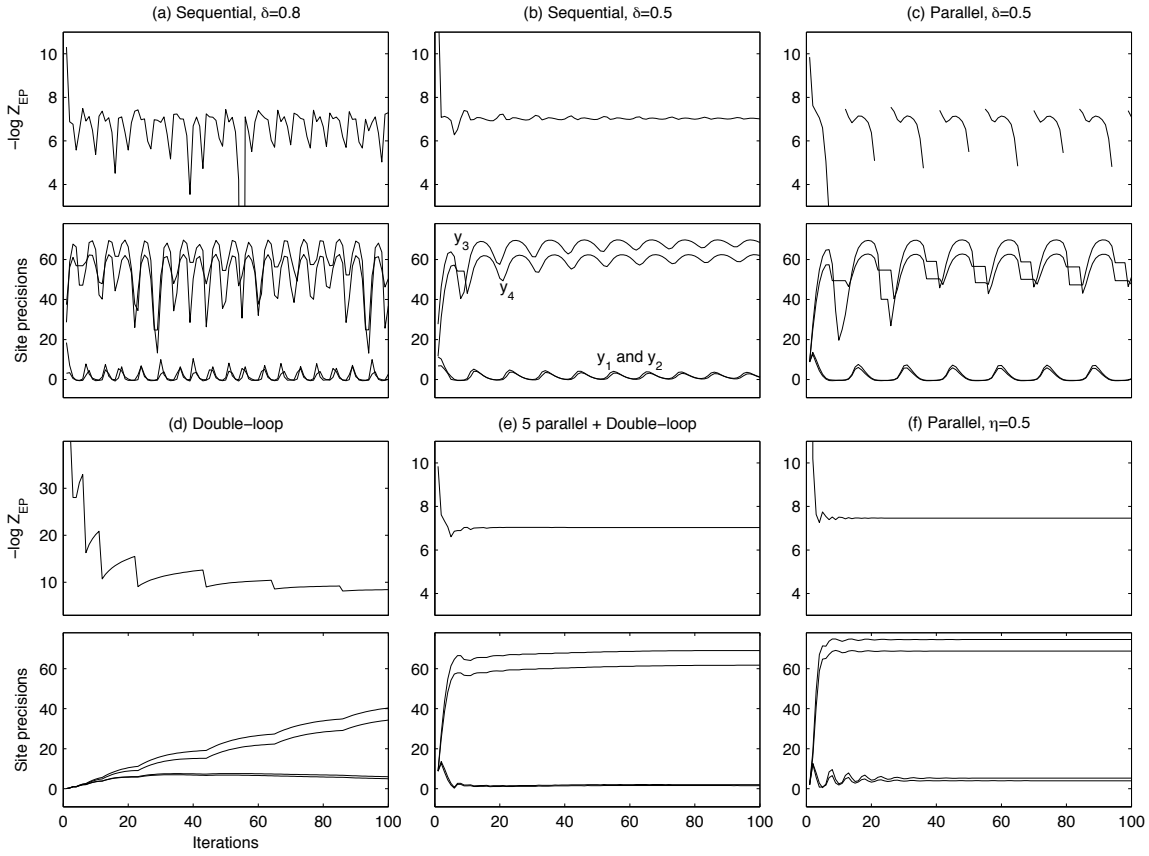


Figure 3: A convergence comparison between sequential and parallel EP as well as the double-loop algorithm in example 2 (the right panel in Figure 1). For each method both the objective $-\log Z_{EP}$ and the site precisions $\tilde{\tau}_i$ related to data points $y_1, ..., y_4$ (see Figure 1) are shown. See Section 5.3 for explanation.

the moments of $\hat{p}_i(f_i)$ and $q(f_i)$ are consistent for fixed $\boldsymbol{\lambda}_s$ before updating $q_{s_i}(f_i)$. For example, a poor choice of δ may require many iterations for achieving inner-loop consistency in the examples 1 or 2, and a too large δ can easily lead to a decrease of the inner-loop objective function or even negative cavity precisions for the sites 1 or 2. Finally, if an unsuccessful update is made due to an unsuitable δ, modification 4 enables automatic determination of a better step size by making use of the concavity of the inner-loop maximization as well as the tilted and marginal moments evaluated at the previous steps with the same $\boldsymbol{\lambda}_s$.

5.3 Convergence Comparisons

Figure 3 illustrates the convergence properties of the different EP algorithms using the data from example 2. The hyperparameters were set to: $\nu = 2$, $\sigma = 0.1$, $\sigma_{\text{se}} = 3$ and $l_k = 0.88$. Panel (a) shows the negative marginal likelihood approximation during the first 100 sweeps with sequential EP and the damping set to $\delta = 0.8$. The panel below shows the site precisions corresponding to

the observations $y_1, ..., y_4$ marked in the upper right panel of Figure 1. With this damping level the site parameters keep oscillating with no convergence and there are also certain parameter values between iterations 50-60 where the marginal likelihood is not defined because of negative cavity precisions (the updates for such sites are skipped until the next iteration). Whenever $\tilde{\tau}_1$ and $\tilde{\tau}_2$ become very small they also inflict large decrease in the site precisions of the nearby sites 3 and 4. These fluctuations affect other sites the more the larger their prior correlations are (defined by the GP prior) with the sites 1 and 2. Panel (b) shows the same graphs with larger amount of damping $\delta = 0.5$. Now the oscillations gradually decrease as more iterations are done but convergence is still very slow. Panel (c) shows the corresponding data with parallel EP and the same amount of damping. The algorithm does not converge and the oscillations are much larger compared to sequential EP. Also the marginal likelihood is not defined at many iterations because of negative cavity precisions.

Panel (d) in Figure 3 illustrates the convergence of the double-loop algorithm with no parallel initialization. There are no oscillations present because the increase of the objective (10) is verified at every iteration and sufficient inner-loop optimality is obtained before proceeding with the outer-loop minimization. However, compared to sequential or parallel EP, the convergence is very slow and it takes over 100 iterations to get the site parameters to the level that sequential EP attains with only a couple of iterations. Panel (e) shows that much faster convergence can be obtained by initializing with 5 parallel iterations and then switching to the double-loop algorithm. There is still some slow drift visible in the site parameters after 20 iterations but changes in the marginal likelihood estimate are very small. Small changes in the site parameters indicate inconsistencies in the moment matching conditions (7) and consequently also the gradient of the marginal likelihood estimate may be slightly inaccurate if the implicit derivatives of $\log Z_{EP}$ with respect to $\boldsymbol{\lambda}_-$ and $\boldsymbol{\lambda}_s$ are assumed zero in the gradient evaluations (Opper and Winther, 2005). Panel (f) shows that parallel EP converges without damping if fractional updates with $\eta = 0.5$ are applied. Because of the different divergence measure the posterior approximation is more localized (see Figure 1) and also the cavity distributions are closer to the respective marginal distributions. It follows that the site precisions related to y_1 and y_2 are larger and no damping is required to keep the updates stable.

5.4 The Marginal Likelihood Approximation

Figure 4 shows contours of the approximate log marginal likelihood with respect to $\log(l_k)$ and $\log(\sigma^2_{\text{se}})$ in the examples of Figure 1. The contours in the first column are obtained by applying first sequential EP with $\delta = 0.8$ and using the double-loop algorithm if it does not converge. The hyperparameter values for which the sequential algorithm does not converge are marked with black dots and the maximum marginal likelihood estimate of the hyperparameters is marked with ($\times$). The second column shows the corresponding results obtained with fractional EP ($\eta = 0.5$) and the corresponding hyperparameter estimates are marked with ($\circ$). For comparison, log marginal likelihood estimates determined with the annealed importance sampling (AIS) (Neal, 2001) are shown in the third column.

In the both examples there is an area of problematic EP updates with smaller length-scales which corresponds to the previously discussed ambiguity about the unknown function near data points y_1 and y_2 in Figure 1. There is also a second area of problematic updates at larger length-scale values in example 2. With larger length-scales the model is too stiff and it is unable to explain large proportion of the data points in the strongly nonlinear region ($-4 < x < -1$) and consequently there

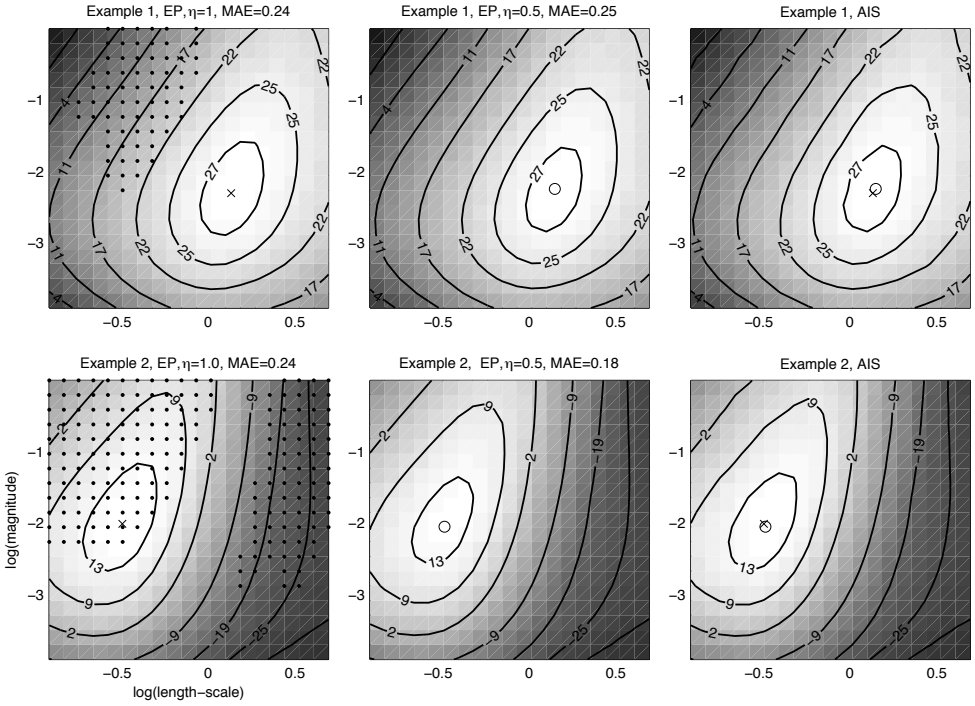


Figure 4: The approximate log marginal likelihood $\log p(\mathbf{y}|\mathbf{X},\theta,\nu,\sigma^2)$ as a function of the log-length-scale $\log(l_k^2)$ and the log-magnitude $\log(\sigma_{se}^2)$ in the examples shown in Figure 1. The marginal likelihood approximation is visualized with both standard EP ($\eta = 1$) and fractional EP ($\eta = 0.5$). The mode of the hyperparameters is marked with $\times$ and $\circ$ for standard and fractional EP respectively. For comparison the marginal is also approximated by annealed importance sampling (AIS). For both standard and fractional EP the mean absolute errors (MAE) over the region with respect to the AIS estimate are also shown. The noise parameter σ^2 and the degrees of freedom ν are fixed to the MAP-estimates obtained with $\eta = 1$. The hyperparameter values in which sequential EP with $\delta = 0.8$ does not converge are marked with black dots in the two leftmost panels.

exist no unique unimodal solution. It is clear that with the first artificial example the optimization of the hyperparameters with sequential EP can fail if not initialized carefully or not enough damping is used. In the second example the sequential EP approximation corresponding to the MAP values cannot even be evaluated because the mode lies in the area of nonconvergent hyperparameter values. In visual comparison with AIS both standard and fractional EP give very similar and accurate approximations in the first example (the contours are drawn at the same levels for each method). In the second example there are more visible differences: standard EP tends to overestimate the marginal likelihood due to the larger posterior uncertainties (see Figure 1) whereas fractional EP underestimates it slightly. This is congruent with the properties of the different divergence measure used in the moment matching. The difference between the hyperparameter values at the modes between standard and fractional EP is otherwise less than 5% except that in the second example σ and ν are ca. 30% larger with fractional EP.

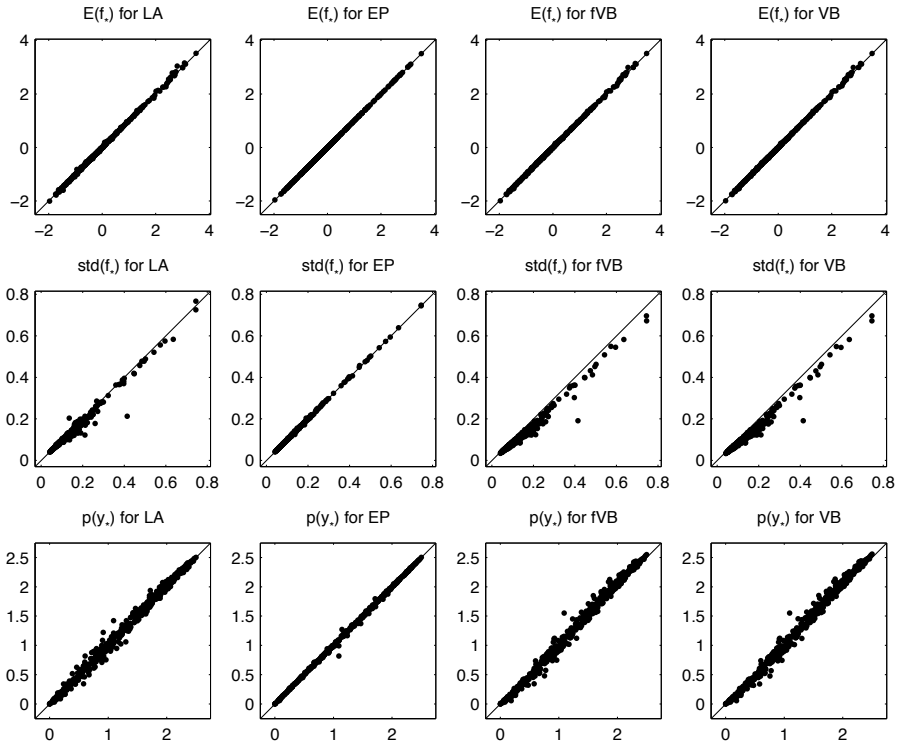


Figure 5: A comparison of the approximative predictive means $\mathrm{E}(f_*|\mathbf{x}_*,\mathcal{D})$, standard deviations $\mathrm{std}(f_*|\mathbf{x}_*,\mathcal{D})$, and predictive densities $q(y_*|\mathbf{x}_*,\mathcal{D})$ provided by the different approximation methods using 10-fold cross-validation on the Boston housing data. The hyperparameters are fixed to the posterior means obtained by a MCMC run on all data. Each dot corresponds to one data point for which the x-coordinate is the MCMC estimate and the y-coordinate the corresponding approximative value obtained with LA, EP, fVB, or VB.

6. Experiments

Four data sets are used to compare the approximative methods: 1) An artificial regression example by Friedman (1991) involving a nonlinear function of 5 inputs. To create a feature selection problem, five irrelevant input variables were added to the data. We generated 10 data sets with 100 training points and 10 randomly selected outliers as described by Kuss (2006). 2) Boston housing data with 506 observations for which the task is to predict the median house prices in the Boston metropolitan area with 13 input variables (see, e.g., Kuss, 2006). 3) Data that involves the prediction of concrete quality based on 27 input variables for 215 experiments (Vehtari and Lampinen, 2002). 4) Data for which the task is to predict the compressive strength of concrete based on 8 input variables for 1030 observations (Yeh, 1998).

6.1 Predictive Comparisons with Fixed Hyperparameters

First we compare the quality of the approximate predictive distributions $q(f_*|\mathbf{x}_*,\mathcal{D},\theta,\nu,\sigma^2)$, where $\mathbf{x}_*$ is the prediction location and $f_* = f(\mathbf{x}_*)$, between all the approximative methods. We ran a full

MCMC on the housing data to determine the posterior mean estimates for the hyperparameters. Then the hyperparameters were fixed to these values and a 10-fold cross-validation was done with all the approximations including MCMC. The predictive means and standard deviations of the latent values as well as the predictive densities of the test observations obtained with Laplace's method (LA), EP, fVB, and VB are plotted against the MCMC estimate in Figure 5. Excluding MCMC, the predictive densities were approximated by numerically integrating over the Gaussian approximation of f_* in $q(y_*|\mathbf{x}_*, \mathcal{D}, \theta, \nu, \sigma^2) = \int p(y_*|f_*, \nu, \sigma^2) q(f_*|\mathbf{x}_*, \mathcal{D}, \theta, \nu, \sigma^2) df_*$. EP gives the most accurate estimates for all the predictive statistics, and clear differences to MCMC can only be seen in the predictive densities of y_* which indicates that accurate mean and variance estimates of the latent value may not always be enough when deriving other predictive statistics. This contrast somewhat to the corresponding results in GP classification where Gaussian approximation was shown to be very accurate in estimating predictive probabilities (Nickisch and Rasmussen, 2008). Both fVB and VB approximate the mean well but are overconfident in the sense that they underestimate the standard deviations, overestimate the larger predictive densities, and underestimate the smaller predictive densities. LA gives similar mean estimates with the VB approximations, but approximates the standard deviations slightly better especially with larger values. Put together, all methods provide decent estimates with fixed hyperparameters but larger performance differences are possible with other hyperparameter values (depending on the non-Gaussianity of the true conditional posterior) and especially when the hyperparameters are optimized.

6.2 Predictive Comparisons with Estimation of the Hyperparameters

In this section we compare the predictive performance of LA, EP, fVB, VB, and MCMC with estimation of the hyperparameters. The predictive performance was measured with the mean absolute error (MAE) and the mean log predictive density (MLPD). These were evaluated for the Friedman data using a test set of 1000 latent variables for each of the 10 simulated data sets. A 10-fold cross validation was used for the Boston housing and concrete quality data whereas a 2-fold cross-validation was used for the compressive strength data because of the large number of observations. To assess the significance of the differences between the model performances, 95% credible intervals of the MLPD measures were approximated by Bayesian bootstrap as described by Vehtari and Lampinen (2002). Gaussian observation model (GA) is selected as a baseline model for comparisons. With GA, LA, EP, and VB the hyperparameters were estimated by optimizing the marginal posterior densities whereas with MCMC all parameters were sampled. The fVB approach was implemented following Kuss (2006) where the hyperparameters are adapted in the M-step of the EM-algorithm. The variational lower bound associated with the M-step was augmented with the same hyperpriors that were used with the other methods.

Since the MAP inference on the degrees of freedom parameter ν proved challenging due to possible identifiability issues, the LA, EP, fVB, and VB approximations are tested both with ν fixed to 4 (LA1, EP1, fVB1, VB1) and optimized together with the other hyperparameters (LA2, EP2, fVB2, VB2). $\nu = 4$ was chosen as a robust default alternative to the normal distribution which allows for outliers but still has finite variance compared to the extremely wide-tailed alternatives with $\nu \leq 2$. With EP we also tested a simple approach (from now on EP3) to approximate the integration over the posterior uncertainty of ν. We selected 15 values ν_j from the interval $[1.5, 20]$ linearly in the log-log scale and ran the optimization of all the other hyperparameters with ν fixed

to these values. The conditional posterior of the latent values was approximated as

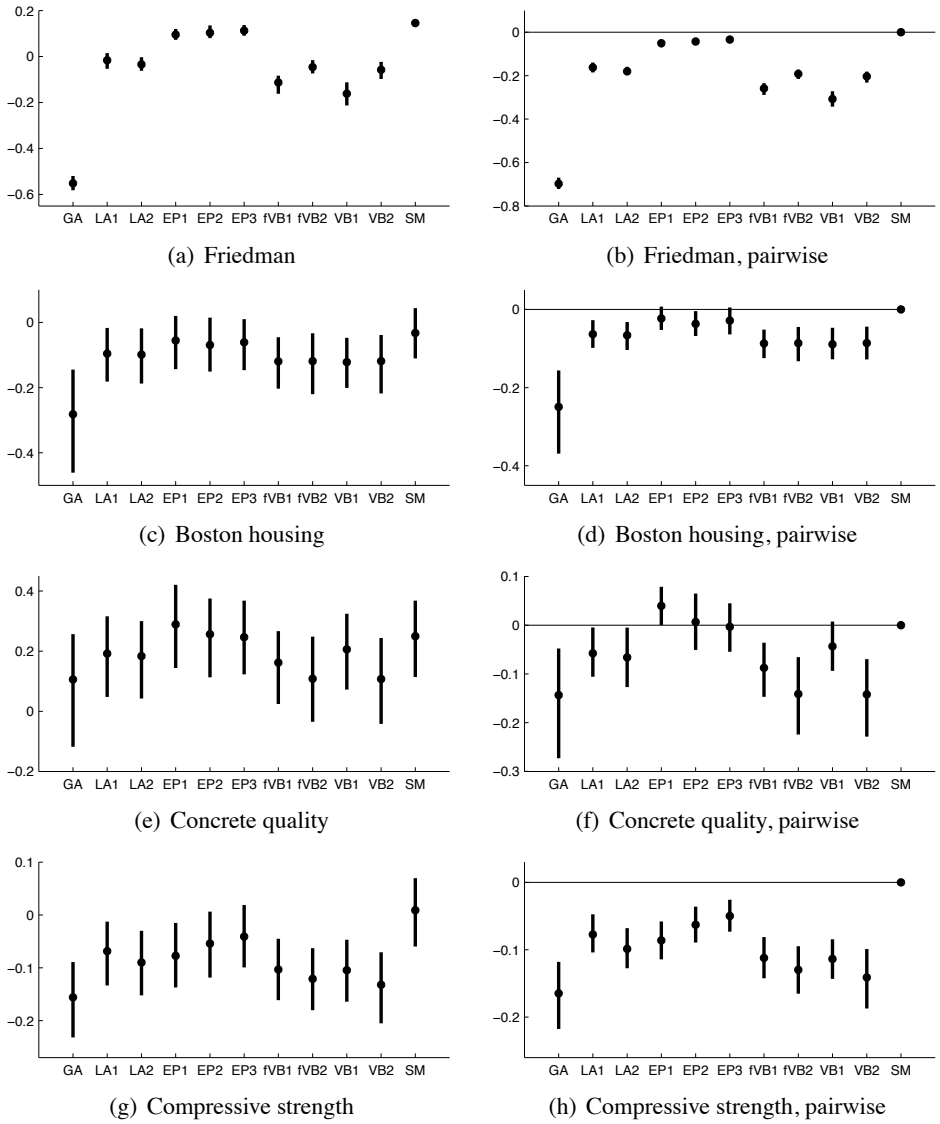
$$p(f_*|\mathbf{x}_*, \mathcal{D}) \approx \sum_j w_j q(f_*|\mathbf{x}_*, \mathcal{D}, \theta_j, \sigma_j^2, \nu_j),$$

where $\{\theta_j, \sigma_j^2\} = \arg\max_{\theta,\sigma^2} q(\theta, \sigma^2|\mathcal{D}, \nu_j)$ and $w_j = q(\theta_j, \sigma_j^2, \nu_j|\mathcal{D}) / \left(\sum_k q(\theta_k, \sigma_k^2, \nu_k|\mathcal{D})\right)$. This can be viewed as a crude approximation of the integration over ν where $p(\theta, \sigma^2|\nu, \mathcal{D})$ is assumed to be very narrowly distributed around the mode. This approximation requires optimization of θ and σ^2 with all the preselected values of ν and therefore θ and σ^2 were initialized to the previous mode to speed up the computations.

The squared exponential covariance (2) was used for all models. Uniform priors were assumed for θ and σ^2 on log-scale and for ν on log-log-scale. The input and target variables were scaled to zero mean and unit variances. ν was initialized to 4, σ to 0.5 and the magnitude σ_{se}^2 to 1. The optimization was done with different random initializations for the length-scales $l_1, ..., l_d$ and the result with the highest posterior marginal density $q(\theta, \nu, \sigma^2|\mathcal{D})$ was chosen. The MCMC inference on the latent values was done with both Gibbs sampling based on the scale-mixture model (3) and direct application of the scaled HMC as described by Vanhatalo and Vehtari (2007). The sampling of the hyperparameters was tested with both slice sampling and HMC. The scale-mixture Gibbs sampling (SM) combined with the slice sampling of the hyperparameters resulted in the best mixing of the chains and gave the best predictive performance which is why only those results are reported. The convergence and quality of the MCMC runs was checked by both visual inspections as well as by calculating the potential scale reduction factors, the effective number of independent samples, and the autocorrelation times (Gelman et al., 2004; Geyer, 1992). Based on the convergence diagnostics, burn-in periods were excluded from the beginning of the chains and the remaining draws were thinned to form the final MCMC estimates.

Figures 6(a), (c), (e) and (g) show the MLPD values together with their 95% credible intervals for all the methods in the four data sets. To illustrate the differences between the approximations more clearly figures 6(b), (d), (f) and (h) show the pairwise comparisons of the log posterior predictive densities to SM. The mean values of the pairwise differences together with their 95% credible intervals are visualized. The Student-*t* model with the SM implementation is significantly better than the Gaussian model with a probability above 95% in all data sets. SM also performs significantly better than all the other approximations on the Friedman and compressive strength data, and on the housing data only EP1 is not significantly worse. The differences are considerably smaller in the concrete quality data on which EP1 actually performs better than SM. One possible explanation for this is a wrong assumption on the noise model (evidence for a covariate dependent noise was found in other experiments). Another possibility is the experimental design used in the data collection; a large proportion of the observations can be classified based on one of the input variables with a very small length scale which is why averaging over this parameter may lead to worse performance.

Additional pairwise comparisons not shown in Figure 6 reveal that either EP1 or EP2 is significantly better than LA, VB, and fVB in all data sets except the compressive strength data for which significant difference is not found when compared to LA1. If the better performing method for estimating ν is selected in either LA, fVB, or VB, LA is better than fVB and VB on the Friedman data and the compressive strength data. No significant differences were found between fVB or VB in pairwise comparisons.



(a) Friedman (b) Friedman, pairwise

(c) Boston housing (d) Boston housing, pairwise

(e) Concrete quality (f) Concrete quality, pairwise

(g) Compressive strength (h) Compressive strength, pairwise

Figure 6: Left column: The mean log posterior predictive density (MLPD) and its 95% central credible interval. The Gaussian observation model (GA) is shown for reference. The Student-t model is inferred with LA, EP, fVB, VB, and scale-mixture based Gibbs sampling (SM). Number 1 after a method means that ν is fixed, number 2 that it is optimized, and number 3 stands for the simple approximative numerical integration over ν. Right column: Pairwise comparisons of the log posterior predictive densities with respect to SM. The mean together with its 95% central credible interval are shown. Values greater than zero indicate that a method is better than SM.

The optimization of ν proved challenging and sensitive to the initialization of the hyperparameters. The most difficult was fVB for which ν often drifted slowly towards infinity. This may be due to our implementation that was made following Kuss (2006) or more likely to the EM style

	GA	LA1	LA2	EP1	EP2	EP3	fVB1	fVB2	VB1	VB2	SM
mean	0.07	1.0	0.8	0.8	7.0	13	15	8.9	1.6	1.8	280
max	0.09	1.0	1.2	1.1	16	26	39	22	3.3	3.8	440
fixed	0.1	1.0		5.5			2.4		1.9		–

Table 1: Two upper rows: The relative CPU times required for the hyperparameter inference. The times are scaled to yield 1 for LA1 separately for each of the four data sets. Both the relative mean (mean) as well as the maximum (max) over the data sets are reported. The third row: The average relative CPU times over the four data sets with the hyperparameters fixed to 28 preselected configurations.

optimization of the hyperparameters. With LA, EP, and VB the integration over $\mathbf{f}$ is redone in the inner-loop for all objective evaluations in the hyperparameter optimization, whereas with fVB the optimization is pursued with fixed approximation $q(\mathbf{f}|\mathcal{D},\theta,\nu,\sigma^2)$. The EP-based marginal likelihood estimate was the most robust with regards to the hyperparameter initialization. According to pairwise comparisons LA2 was significantly worse than LA1 only in the compressive strength data. EP2 was significantly better than EP1 in the housing and compressive strength data but significantly worse with the housing data. With fVB and VB optimization of ν gave significantly better performance only with the simulated Friedman data, and significant decrease was observed with VB2 in the housing and compressive strength data. In pairwise comparisons, the crude numerical integration over ν (EP3) was significantly better than EP1 and EP2 with the housing and compressive strength data, but never significantly worse. These results give evidence that the EP approximation is more reliable in the hyperparameter inference because of the more accurate marginal likelihood estimates which is in line with the results in GP classification (Nickisch and Rasmussen, 2008).

In terms of MAE the Student-*t* model was significantly better than GA in all data sets besides the concrete quality data, in which only EP1 gave better results. If the best performing hyperparameter inference scheme is selected for each method, EP is significantly better than the others on all the data sets excluding the compressive strength data in which the differences were not significant. EP was better than SM on the Friedman and concrete quality data but no other significant differences were found in comparisons with SM. LA was significantly better than fVB and VB on the compressive strength data whereas on the simulated Friedman data VB was better than LA and fVB.

Table 1 summarizes the total CPU times required for the posterior inference including the hyperparameter optimization and the predictions. The CPU times are scaled to give one for LA1 and both the mean and maximum over the four data sets are reported. The running times of the fastest Student-*t* approximations are roughly 10-fold compared to the baseline method GA. EP1, where $\nu = 4$, is surprisingly fast compared to LA but it gets much slower with the optimization of ν (EP2). This is explained by the increasing number of double-loop iterations required to achieve convergence with the larger number of difficult posterior distributions as ν gets smaller values. EP3 is clearly more demanding compared to EP1 or EP2 because the optimization has to be repeated with every preselected value of ν. fVB is quite slow compared to LA or VB because of the slowly progressing EM-based hyperparameter adaptation. The running times of LA and VB are quite similar with ν both fixed and optimized. The running times are suggestive since they depend much on the implementations, convergence thresholds and the hyperparameter initializations. Table 1 shows also the average relative running times over the four data sets (excluding MCMC) with the hyperparam-

eters fixed to 28 different configurations (fixed). The configurations were created by first including the MCMC mean for each data set and then generating all combinations of three clearly different values of ν, σ, and σ_{se} around the MCMC mean with randomly selected lengthscales. The average relative running time is higher with EP because many difficult hyperparameter configurations were created.

7. Discussion

Much research has been done on EP and it has been found very accurate and computationally efficient in many practical applications. Although non-log-concave site functions may be problematic for EP it has been used and found effective for many potentially difficult models such as the Gaussian mixture likelihoods (Kuss, 2006; Stegle et al., 2008) as well as "spike and slab" priors (Hernández-Lobato et al., 2008). Modifications such as the damping and fractional updates as well as alternative double-loop algorithms have been proposed to improve the stability in difficult cases but the practical implementation issues have not been discussed that much. In this work we have given another demonstration of the good predictive performance of EP in a challenging model but also analyzed the convergence problems and the EP improvements from a practical point of view. In addition, we have presented practical guidelines for a robust parallel EP implementation that can be applied for other non-log-concave likelihoods as well.

We have described the properties of the EP algorithm and its modifications with the Student-*t* observation model, but the same key challenges can also be considered with respect to a general observation model with a non-log-concave likelihood. With a Gaussian prior on $\mathbf{f}$ and a log-concave likelihood, each site approximation increases the posterior precision and all the site precisions remain positive throughout the EP iterations as was shown by Seeger (2008). With a non-log-concave likelihood, however, negative site precisions may occur. The negative site precisions are natural and well justified because a non-log-concave likelihood can generate local increases of the posterior uncertainty which cannot otherwise be modeled with the Gaussian approximation. For example, as discussed here and by Vanhatalo et al. (2009), with the Student-*t* model the negative site precisions correspond to the outlying observations. Through the prior covariances of $\mathbf{f}$, the negative site precisions decrease also the approximate marginal posterior precisions of the other site approximations with positive site precisions. This may become a problem during the sequential or the parallel EP iterations if some of the approximate marginal posterior precisions decrease close to the level of the corresponding site precisions. In such cases the respective cavity precisions become very small which can both induce numerical instabilities in the tilted moment integrations (Seeger, 2008) and make the respective sites very sensitive to the subsequent EP updates. If the EP updates are not constrained some of the cavity precisions may also become negative in which case the tilted moments and the following updates are no longer well defined.

Both of the well-known EP modifications help to alleviate the above described problem. Damping takes more conservative update steps so that the negative site precision increments are less likely to decrease the other cavity precisions too much. Fractional EP keeps the cavity precisions larger by leaving a fraction of the site precisions in the cavity but leads to different approximation which may underestimate the posterior uncertainties. The double-loop algorithm is computationally demanding but admissible steps in the concave inner-loop maximization ensure that the cavity and the tilted distributions remain well defined at all times. And most importantly, the inner-loop maximiza-

tion forms an upper-bound which provably converges to a stationary solution satisfying the moment matching conditions (7).

The general modifications described in Section 4 bring additional stability with reasonable computational cost. Modification 1 is a principled way to avoid immediate problems arising from a too large step size. It ensures that each parallel EP update results in an increase of the inner-loop objective, and it is computationally cheap with likelihoods for which the tilted moments can be determined analytically (e.g., finite Gaussian mixtures). Modification 2 is also computationally cheap and it ensures that the cavity distributions (defined with fixed $\boldsymbol{\lambda}_s$) remain well defined at all times. If the current step size does not result in a sufficient decrease of the energy the extra tilted moment evaluations required in modification 1 can be used in determining a better step length based on the gradient information according to modification 4 with little additional computational cost.

Modification 3 comes with a considerable computational cost if a small tolerance is required for the inner-loop iterations. However, in our experiments with the Student-*t* model, a relaxed double-loop scheme with a maximum of two inner-loop iterations and two step-size adjustments steps (only if required) were sufficient to achieve convergence. In practice this requires at most three additional matrix inversions per iteration compared to the regular parallel EP but unfortunately also the number of outer loop iterations tended to increase with the more difficult data sets and hyperparameter values. In these cases the main challenge was the difficult inner-loop moment matching which can be partly related to a too inflexible approximating family and partly to a suboptimal search direction defined by parallel EP. Considering the better convergence properties of sequential EP (see Section 5.3), for instance a scheme, where the inner-loop optimization of the more difficult sites (whose cavity distributions differ notably from the respective marginals) was done sequentially and the remaining sites were optimized with parallel updates, could lead to better overall performance.

The nonlinear GP regression combined with the Student-*t* model makes the inference problem challenging because the potential multimodality of the conditional posterior depends on the hyperparameter values. As we have demonstrated by examples, standard EP may not converge with the MAP estimates of the hyperparameters. Therefore, in practical applications, one cannot simply discard all problematic hyperparameter values. Instead some estimate of the marginal likelihood is required also in the more difficult cases. In our examples these situations were related to two modes in the conditional posterior (caused by two outliers) quite far away from each other which requires a very large local increase of the marginal variances from the unimodal posterior approximation. (It should also be noted that moderately damped sequential EP worked fine with many other multimodal posterior distributions.) The globally unimodal assumption is not the best in such cases although the true underlying function is unimodal, but we think that it is important to get some useful posterior approximation. Whether one prefers the possible false certainty provided by the Laplace or VB approximations, or the possible false uncertainty of EP, is a matter of taste but we prefer the latter one.

It is also important that the inference procedure gives some clue of the potential inadequacy of the approximating family so that more elaborate models can be considered. In addition to the examination of the posterior approximation, the need for double-loop iterations with the MAP hyperparameter estimates may be one indication of an unsuitable model. One can also compare the cavity distributions, which can be regarded as the LOO estimates of the latent values, with the respective marginal approximations. If for certain sites most of the LOO information comes from the corresponding site approximations there is reason to suspect that the approximation is not suit-

able. Our EP implementation enables a robust way of forming such approximations and in case of problems it also enables automatic switching to fractional updates.

The presented EP approach for approximative inference with GP models is implemented in the freely available GPstuff software package (`http://www.lce.hut.fi/research/mm/gpstuff/`). The software also allows experimenting with other non-log-concave likelihoods by implementing the necessary tilted moment integrations in a separate likelihood function.

Acknowledgments

We would like to thank Janne Ojanen and the anonymous reviewers for their valuable comments to improve the manuscript. The study was supported financially by the Academy of Finland (grants 218248, 129230, 129670) and by the Finnish Foundation for Technology Promotion.

References

Christopher M. Bishop. *Pattern Recognition and Machine Learning*. Springer Science +Business Media LLC, 2006.

Botond Cseke and Tom Heskes. Properties of Bethe free energies and message passing in Gaussian models. *Journal of Artificial Intelligence Research*, 41:1–24, 2011.

A. Philip Dawid. Posterior expectations for large observations. *Biometrika*, 60(3):664–667, 1973.

Bruno De Finetti. The Bayesian approach to the rejection of outliers. In *Proceedings of the fourth Berkeley Symposium on Mathematical Statistics and Probability*, pages 199–210. University of California Press, 1961.

Jerome H. Friedman. Multivariate adaptive regression splines. *Annals of Statistics*, 19(1):1–67, 1991.

Andrew Gelman, John B. Carlin, Hal S. Stern, and Donald B. Rubin. *Bayesian Data Analysis*. Chapman & Hall/CRC, second edition, 2004.

John Geweke. Bayesian treatment of the independent Student-*t* linear model. *Journal of Applied Econometrics*, 8:519–540, 1993.

Charles J. Geyer. Practical markov chain monte carlo. *Statistical Science*, 7(12):473–483, 1992.

Mark N. Gibbs and David J. C. MacKay. Variational Gaussian process classifiers. *IEEE Transactions on Neural Networks*, 11(6):1458–1464, 2000.

Paul W. Goldberg, Christopher K. I. Williams, and Christopher M. Bishop. Regression with input-dependent noise: A Gaussian process treatment. In M. I. Jordan, M. J. Kearns, and S. A Solla, editors, *Advances in Neural Information Processing Systems 10*. MIT Press, Cambridge, MA, 1998.

José Miguel Hernández-Lobato, Tjeerd Dijkstra, and Tom Heskes. Regulator discovery from gene expression time series of malaria parasites: a hierarchical approach. In J. C. Platt, D. Koller,

Y. Singer, and S. Roweis, editors, *Advances in Neural Information Processing Systems 20*, pages 649–656. MIT Press, Cambridge, MA, 2008.

Tom Heskes and Onno Zoeter. Expectation propagation for approximate inference in dynamic Bayesian networks. In A. Darwiche and N. Friedman, editors, *Uncertainty in Artificial Intelligence: Proceedings of the Eighteenth Conference (UAI-2002)*, pages 216–233. Morgan Kaufmann, San Francisco, CA, 2002.

Tom Heskes, Manfred Opper, Wim Wiegerinck, Ole Winther, and Onno Zoeter. Approximate inference techniques with expectation constraints. *Journal of Statistical Mechanics: Theory and Experiment*, 2005:P11015, 2005.

Malte Kuss. *Gaussian Process Models for Robust Regression, Classification, and Reinforcement Learning*. PhD thesis, Technische Universität Darmstadt, 2006.

Chuanhai Liu and Donald B. Rubin. ML estimation of the t distribution using EM and its extensions, ECM and ECME. *Statistica Sinica*, 5:19–39, 1995.

Thomas P. Minka. *A Family of Algorithms for Approximate Bayesian Inference*. PhD thesis, Massachusetts Institute of Technology, 2001a.

Thomas P. Minka. Expectation propagation for approximate Bayesian inference. In *Proceedings of the 17th Conference in Uncertainty in Artificial Intelligence (UAI-2001)*, pages 362–369. Morgan Kaufmann, San Francisco, CA, 2001b.

Thomas P. Minka. Power EP. Technical report, Microsoft Research, Cambridge, 2004.

Thomas P. Minka. Divergence measures and message passing. Technical report, Microsoft Research, Cambridge, 2005.

Thomas P. Minka and John Lafferty. Expectation-propagation for the generative aspect model. In *Proceedings of the 18th Conference on Uncertainty in Artificial Intelligence (UAI-2002)*, pages 352–359. Morgan Kaufmann, San Francisco, CA, 2002.

Andrew Naish-Guzman and Sean Holden. Robust regression with twinned Gaussian processes. In J. C. Platt, D. Koller, Y. Singer, and S. Roweis, editors, *Advances in Neural Information Processing Systems 20*, pages 1065–1072. MIT Press, Cambridge, MA, 2008.

Radford M. Neal. Monte Carlo Implementation of Gaussian Process Models for Bayesian Regression and Classification. Technical Report 9702, Dept. of statistics and Dept. of Computer Science, University of Toronto, 1997.

Radford M. Neal. Annealed importance sampling. *Statistics and Computing*, 11(2):125–139, 2001.

Hannes Nickisch and Carl E. Rasmussen. Approximations for binary Gaussian process classification. *Journal of Machine Learning Research*, 9:2035–2078, 2008.

Anthony O'Hagan. On outlier rejection phenomena in Bayes inference. *Journal of the Royal Statistical Society (Series B)*, 41(3):358–367, 1979.

Manfred Opper and Cédric Archambeau. The variational Gaussian approximation revisited. *Neural Computation*, 21(3):786–792, 2009.

Manfred Opper and Ole Winther. Expectation consistent approximate inference. *Journal of Machine Learning Research*, 6:2177–2204, 2005.

Carl E. Rasmussen and Hannes Nickisch. Gaussian processes for machine learning (GPML) toolbox. *Journal of Machine Learning Research*, 11:3011–3015, 2010.

Carl Edward Rasmussen and Christopher K. I. Williams. *Gaussian Processes for Machine Learning*. MIT Press, Cambridge, MA, 2006.

Håvard Rue, Sara Martino, and Nicolas Chopin. Approximate Bayesian inference for latent Gaussian models by using integrated nested Laplace approximations. *Journal of the Royal statistical Society (Series B)*, 71(2):1–35, 2009.

Matthias Seeger. Expectation propagation for exponential families. Technical report, Max Planck Institute for Biological Cybernetics, Tübingen, Germany, 2005.

Matthias Seeger. Bayesian inference and optimal design for the sparse linear model. *Journal of Machine Learning Research*, 9:759–813, 2008.

Matthias Seeger and Hannes Nickisch. Fast convergent algorithms for expectation propagation approximate Bayesian inference. In *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, volume 15, pages 652–660. JMLR W&CP, 2011.

Lawrence F. Shampine. Vectorized adaptive quadrature in MATLAB. *Journal of Computational and Applied Mathematics*, 211:131–140, 2008.

Oliver Stegle, Sebastian V. Fallert, David J. C. MacKay, and Søren Brage. Gaussian process robust regression for noisy heart rate data. *Biomedical Engineering, IEEE Transactions on*, 55(9):2143–2151, 2008.

Luke Tierney and Joseph B. Kadane. Accurate approximations for posterior moments and marginal densities. *Journal of the American Statistical Association*, 81(393):82–86, 1986.

Michael E. Tipping and Neil D. Lawrence. A variational approach to robust Bayesian interpolation. In *In Proceedings of the IEEE International Workshop on Neural Networks for Signal Processing*, pages 229–238. IEEE, 2003.

Michael E. Tipping and Neil D. Lawrence. Variational inference for Student-*t* models: Robust bayesian interpolation and generalised component analysis. *Neurocomputing*, 69:123–141, 2005.

Marcel van Gerven, Botond Cseke, Robert Oostenveld, and Tom Heskes. Bayesian source localization with the multivariate Laplace prior. In Y. Bengio, D. Schuurmans, J. Lafferty, C. K. I. Williams, and A. Culotta, editors, *Advances in Neural Information Processing Systems 22*, pages 1901–1909, 2009.

Jarno Vanhatalo and Aki Vehtari. Sparse log Gaussian processes via MCMC for spatial epidemiology. *JMLR Workshop and Conference Proceedings*, 1:73–89, 2007.

Jarno Vanhatalo, Pasi Jylänki, and Aki Vehtari. Gaussian process regression with Student-*t* likelihood. In Y. Bengio, D. Schuurmans, J. Lafferty, C. K. I. Williams, and A. Culotta, editors, *Advances in Neural Information Processing Systems 22*, pages 1910–1918, 2009.

Aki Vehtari and Jouko Lampinen. Bayesian model assessment and comparison using cross-validation predictive densities. *Neural Computation*, 14(10):2439–2468, 2002.

Mike West. Outlier models and prior distributions in Bayesian linear regression. *Journal of the Royal Statistical Society (Series B)*, 46(3):431–439, 1984.

Christopher K. I. Williams and David Barber. Bayesian classification with Gaussian processes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(12):1342–1351, 1998.

I-Cheng Yeh. Modeling of strength of high performance concrete using artificial neural networks. *Cement and Concrete Research*, 28(12):1797–1808, 1998.

Journal of Machine Learning Research 12 (2011) 3259-3281 Submitted 11/10; Revised: 8/11; Published 11/11

The Sample Complexity of Dictionary Learning

Daniel Vainsencher DANIELV@TX.TECHNION.AC.IL
Shie Mannor SHIE@EE.TECHNION.AC.IL
Department of Electrical Engineering
Technion, Israel Institute of Technology
Haifa 32000, Israel

Alfred M. Bruckstein FREDDY@CS.TECHNION.AC.IL
Department of Computer Science
Technion, Israel Institute of Technology
Haifa 32000, Israel

Editor: Gábor Lugosi

Abstract

A large set of signals can sometimes be described sparsely using a dictionary, that is, every element can be represented as a linear combination of few elements from the dictionary. Algorithms for various signal processing applications, including classification, denoising and signal separation, learn a dictionary from a given set of signals to be represented. Can we expect that the error in representing by such a dictionary a previously unseen signal from the same source will be of similar magnitude as those for the given examples? We assume signals are generated from a fixed distribution, and study these questions from a statistical learning theory perspective.

We develop generalization bounds on the quality of the learned dictionary for two types of constraints on the coefficient selection, as measured by the expected L_2 error in representation when the dictionary is used. For the case of l_1 regularized coefficient selection we provide a generalization bound of the order of $O\left(\sqrt{np\ln(m\lambda)/m}\right)$, where n is the dimension, p is the number of elements in the dictionary, λ is a bound on the l_1 norm of the coefficient vector and m is the number of samples, which complements existing results. For the case of representing a new signal as a combination of at most k dictionary elements, we provide a bound of the order $O(\sqrt{np\ln(mk)/m})$ under an assumption on the closeness to orthogonality of the dictionary (low Babel function). We further show that this assumption holds for *most* dictionaries in high dimensions in a strong probabilistic sense. Our results also include bounds that converge as $1/m$, not previously known for this problem. We provide similar results in a general setting using kernels with weak smoothness requirements.

Keywords: dictionary learning, generalization bound, sparse representation

1. Introduction

A common technique in processing signals from $\mathcal{X} = \mathbb{R}^n$ is to use sparse representations; that is, to approximate each signal x by a "small" linear combination a of elements d_i from a dictionary $D \in \mathcal{X}^p$, so that $x \approx Da = \sum_{i=1}^{p} a_i d_i$. This has various uses detailed in Section 1.1. The smallness of a is often measured using either $\|a\|_1$, or the number of non zero elements in a, often denoted $\|a\|_0$. The approximation error is measured here using a Euclidean norm appropriate to the vector space.

We denote the approximation error of x using dictionary D and coefficients from a set A by

$$h_{A,D}(x) = \min_{a \in A} \|Da - x\|, \tag{1}$$

where A is one of the following sets determining the sparsity required of the representation:

$$H_k = \{a : \|a\|_0 \leq k\}$$

induces a "hard" sparsity constraint, which we also call k sparse representation, while

$$R_\lambda = \{a : \|a\|_1 \leq \lambda\}$$

induces a convex constraint that is considered a "relaxation" of the previous constraint.

The dictionary learning problem is to find a dictionary D minimizing

$$E(D) = \mathbb{E}_{x \sim \nu} h_{A,D}(x), \tag{2}$$

where ν is a distribution over signals that is known to us only through samples from it. The problem addressed in this paper is the "generalization" (in the statistical learning sense) of dictionary learning: to what extent does the performance of a dictionary chosen based on a finite set of samples indicate its expected error in (2)? This clearly depends on the number of samples and other parameters of the problem such as the dictionary size. In particular, an obvious algorithm is to represent each sample using itself, if the dictionary is allowed to be as large as the sample, but the performance on unseen signals is likely to disappoint.

To state our goal more quantitatively, assume that an algorithm finds a dictionary D suited to k sparse representation, in the sense that the average representation error $E_m(D)$ on the m examples given to the algorithm is low. Our goal is to bound the generalization error ε, which is the additional expected error that might be incurred:

$$E(D) \leq (1+\eta)E_m(D) + \varepsilon, \tag{3}$$

where $\eta \geq 0$ is sometimes zero, and the bound ε depends on the number of samples and problem parameters. Since efficient algorithms that find the optimal dictionary for a given set of samples (also known as empirical risk minimization, or ERM, algorithms) are not known for dictionary learning, we prove uniform convergence bounds that apply simultaneously over all admissible dictionaries D, thus bounding from above the sample complexity of the dictionary learning problem. In particular, such a result means that every procedure for approximate minimization of empirical error (empirical dictionary learning) is also a procedure for approximate dictionary learning (as defined above) in a similar sense.

Many analytic and algorithmic methods relying on the properties of finite dimensional Euclidean geometry can be applied in more general settings by applying kernel methods. These consist of treating objects that are not naturally represented in $\mathbb{R}^n$ as having their similarity described by an inner product in an abstract *feature space* that is Euclidean. This allows the application of algorithms depending on the data only through a computation of inner products to such diverse objects as graphs, DNA sequences and text documents (Shawe-Taylor and Cristianini, 2004). Is it possible to extend the usefulness of dictionary learning techniques to this setting? We address sample complexity aspects of this question as well.

1.1 Background and Related Work

Sparse representations are by now standard practice in diverse fields such as signal processing, natural language processing, etc. Typically, the dictionary is assumed to be known. The motivation for sparse representations is indicated by the following results, in which we assume the signals come from $\mathcal{X} = \mathbb{R}^n$ are normalized to have length 1, and the representation coefficients are constrained to $A = H_k$ where $k < n, p$ and typically $h_{A,D}(x) \ll 1$.

- Compression: If a signal x has an approximate sparse representation in some commonly known dictionary D, it can be stored or transmitted more economically with reasonable precision. Finding a good sparse representation can be computationally hard but if D fulfills certain geometric conditions, then its sparse representation is unique and can be found efficiently (see, e.g., Bruckstein et al., 2009).

- Denoising: If a signal x has a sparse representation in some known dictionary D, and $\tilde{x} = x + \nu$, where the random noise ν is Gaussian, then the sparse representation found for $\tilde{x}$ will likely be very close to x (for example Chen et al., 2001).

- Compressed sensing: Assuming that a signal x has a sparse representation in some known dictionary D that fulfills certain geometric conditions, this representation can be approximately retrieved with high probability from a small number of random linear measurements of x. The number of measurements needed depends on the sparsity of x in D (Candes and Tao, 2006).

The implications of these results are significant when a dictionary D is known that sparsely represents simultaneously many signals. In some applications the dictionary is chosen based on prior knowledge, but in many applications the dictionary is learned based on a finite set of examples. To motivate dictionary learning, consider an image representation used for compression or denoising. Different types of images may have different properties (MRI images are not similar to scenery images), so that learning a dictionary specific to each type of images may lead to improved performance. The benefits of dictionary learning have been demonstrated in many applications (Protter and Elad, 2007; Peyré, 2009).

Two extensively used techniques related to dictionary learning are Principal Component Analysis (PCA) and K-means clustering. The former finds a single subspace minimizing the sum of squared representation errors which is very similar to dictionary learning with $A = H_k$ and $p = k$. The latter finds a set of locations minimizing the sum of squared distances between each signal and the location closest to it which is very similar to dictionary learning with $A = H_1$ where p is the number of locations. Thus we could see dictionary learning as PCA with multiple subspaces, or as clustering where multiple locations are used to represent each signal. The sample complexities of both algorithms are well studied (Bartlett et al., 1998; Biau et al., 2008; Shawe-Taylor et al., 2005; Blanchard et al., 2007).

This paper does not address questions of computational cost, though they are very relevant. Finding optimal coefficients for k sparse representation (that is, minimizing (1) with $A = H_k$) is NP-hard in general (Davis et al., 1997). Dictionary learning as the optimization problem of minimizing (2) is less well understood, even for empirical ν (consisting of a finite number of samples), despite over a decade of work on related algorithms with good empirical results (Olshausen and Field, 1997; Lewicki et al., 1998; Kreutz-Delgado et al., 2003; Aharon et al., 2006; Lee et al., 2007; Mairal et al., 2010).

The only prior work we are aware of that addresses generalization in dictionary learning, by Maurer and Pontil (2010), addresses the convex representation constraint $A = R_\lambda$; we discuss the relation of our work to theirs in Section 2.

2. Results

Except where we state otherwise, we assume signals are generated in the unit sphere $\mathbb{S}^{n-1}$. Our results are:

A new approach to dictionary learning generalization. Our first main contribution is an approach to generalization bounds in dictionary learning that is complementary to the approach used by Maurer and Pontil (2010). The previous result, given below in Theorem 6 has generalization error bounds (the ε of inequality (3)) of order

$$O\left(\sqrt{p\min(p,n)\left(\lambda+\sqrt{\ln(m\lambda)}\right)^2/m}\right)$$

on the squared representation error. A notable feature of this result is the weak dependence on the signal dimension n. In Theorem 1 we quantify the complexity of the class of functions $h_{A,D}$ over all dictionaries whose columns have unit length, where $A \subset R_\lambda$. Combined with standard methods of uniform convergence this results in generalization error bounds ε of order $O\left(\sqrt{np\ln(m\lambda)/m}\right)$ when $\eta = 0$. While our bound does depend strongly on n, this is acceptable in the case $n < p$, also known in the literature as the "over-complete" case (Olshausen and Field, 1997; Lewicki et al., 1998). Note that our generalization bound applies with different constants to the representation error itself and many variants including the squared representation error, and has a weak dependence on λ. The dependence on λ is significant, for example, when $\|a\|_1$ is used as a weighted penalty term by solving $\min_a \|Da - X\| + \gamma \cdot \|a\|_1$; in this case $\lambda = O\left(\gamma^{-1}\right)$ may be quite large.

Fast rates. For the case $\eta > 0$ our methods allow bounds of order $O(np\ln(\lambda m)/m)$. The main significance of this is in that the general statistical behavior they imply occurs in dictionary learning. For example, generalization error has a "proportional" component which is reduced when the empirical error is low. Whether fast rates results can be proved in the dimension free regime is an interesting question we leave open. Note that due to lower bounds by Bartlett et al. (1998) of order $\sqrt{m^{-1}}$ on the k-means clustering problem, which corresponds to dictionary learning for 1-sparse representation, fast rates may be expected only with $\eta > 0$, as presented here.

We now describe the relevant function class and the bounds on its complexity, which are proved in Section 3. The resulting generalization bounds are given explicitly at the end of this section.

Theorem 1 *For every $\varepsilon > 0$, the function class*

$$\mathcal{G}_\lambda = \left\{h_{R_\lambda,D} : \mathbb{S}^{n-1} \to \mathbb{R} : D \in \mathbb{R}^{n\times p}, \|d_i\| \le 1\right\},$$

taken as a metric space with the distance induced by $\|\cdot\|_\infty$, has a subset of cardinality at most $(4\lambda/\varepsilon)^{np}$, such that every element from the class is at distance at most ε from the subset.

While we give formal definitions in Section 3, such a subset is called an ε cover, and such a bound on its cardinality is called a covering number bound.

Extension to k sparse representation. Our second main contribution is to extend both our approach and that of Maurer and Pontil (2010) to provide generalization bounds for dictionaries for k

sparse representations, by using a bound λ on the l_1 norm of the representation coefficients when the dictionaries are close to orthogonal. Distance from orthogonality is measured by the Babel function (which, for example, upper bounds the magnitude of the maximal inner product between distinct dictionary elements) defined below and discussed in more detail in Section 4.

Definition 2 (Babel function, Tropp 2004) *For any $k \in \mathbb{N}$, the Babel function $\mu_k : \mathbb{R}^{n\times m} \to \mathbb{R}^+$ is defined by:*

$$\mu_k(D) = \max_{i\in\{1,\ldots,p\}} \max_{\Lambda\subset\{1,\ldots,p\}\setminus\{i\};|\Lambda|=k} \sum_{j\in\Lambda} \left|\left\langle d_j, d_i\right\rangle\right|.$$

The following proposition, which is proved in Section 3, bounds the 1-norm of the dictionary coefficients for a k sparse representation and also follows from analysis previously done by Donoho and Elad (2003) and Tropp (2004).

Proposition 3 *Let each column d_i of D fulfill $\|d_i\| \in [1,\gamma]$ and $\mu_{k-1}(D) \le \delta < 1$, then a coefficient vector $a \in \mathbb{R}^p$ minimizing the k-sparse representation error $h_{H_k,D}(x)$ exists which has $\|a\|_1 \le \gamma k/(1-\delta)$.*

We now consider the class of all k sparse representation error functions. We prove in Section 3 the following bound on the complexity of this class.

Corollary 4 *The function class*

$$\mathcal{F}_{\delta,k} = \left\{h_{H_k,D} : \mathbb{S}^{n-1} \to \mathbb{R} : \mu_{k-1}(D) < \delta, d_i \in \mathbb{S}^{n-1}\right\},$$

taken as a metric space with the metric induced by $\|\cdot\|_\infty$, has a covering number bound of at most $(4k/(\varepsilon(1-\delta)))^{np}$.

The dependence of the last two results on $\mu_{k-1}(D)$ means that the resulting bounds will be meaningful only for algorithms which explicitly or implicitly prefer near orthogonal dictionaries. Contrast this to Theorem 1 which does not require significant conditions on the dictionary.

Asymptotically almost all dictionaries are near orthogonal. A question that arises is what values of μ_{k-1} can be expected for parameters n, p, k? We shed some light on this question through the following probabilistic result, which we discuss in Section 4 and prove in Appendix B.

Theorem 5 *Suppose that D consists of p vectors chosen uniformly and independently from $\mathbb{S}^{n-1}$. Then we have*

$$P(\mu_k > \delta) \le \sqrt{\frac{\pi}{2}}\, p(p-1) \exp\left(-\frac{(n-2)\left(\frac{\delta}{k}\right)^2}{2}\right).$$

Since low values of the Babel function have implications to representation finding algorithms, this result is of interest also outside the context of dictionary learning. Essentially it means that random dictionaries whose cardinality is sub-exponential in $(n-2)/k^2$ have low Babel function.

New generalization bounds for l_1 case. The covering number bound of Theorem 1 implies several generalization bounds for the problem of dictionary learning for l_1 regularized representations which we give here. These differ from bounds by Maurer and Pontil (2010) in depending more strongly on the dimension of the space, but less strongly on the particular regularization term. We

first give the relevant specialization of the result by Maurer and Pontil (2010) for comparison and for reference as we will later build on it. This result is independent of the dimension n of the underlying space, thus the Euclidean unit ball B may be that of a general Hilbert space, and the errors measured by $h_{A,D}$ are in the same norm.

Theorem 6 (Maurer and Pontil 2010) *Let $A \subset R_\lambda$, and let ν be any distribution on the unit sphere B. Then with probability at least $1-e^{-x}$ over the m samples in E_m drawn according to ν, for all dictionaries $D \subset B$ with cardinality p:*

$$Eh_{A,D}^2 \leq E_m h_{A,D}^2 + \sqrt{\frac{p^2\left(14\lambda + 1/2\sqrt{\ln\left(16m\lambda^2\right)}\right)^2}{m}} + \sqrt{\frac{x}{2m}}.$$

Using the covering number bound of Theorem 1 and a bounded differences concentration inequality (see Lemma 21), we obtain the following result. The details are given in Section 3.

Theorem 7 *Let $\lambda > e/4$, with ν a distribution on $\mathbb{S}^{n-1}$. Then with probability at least $1-e^{-x}$ over the m samples in E_m drawn according to ν, for all D with unit length columns:*

$$Eh_{R_\lambda,D} \leq E_m h_{R_\lambda,D} + \sqrt{\frac{np\ln\left(4\sqrt{m}\lambda\right)}{2m}} + \sqrt{\frac{x}{2m}} + \sqrt{\frac{4}{m}}.$$

Using the same covering number bound and the general result Corollary 23 (given in Section 3), we obtain the following fast rates result. A slightly more general result is easily derived by using Proposition 22 instead.

Theorem 8 *Let $\lambda > e/4$, $np \geq 20$ and $m \geq 5000$ with ν a distribution on $\mathbb{S}^{n-1}$. Then with probability at least $1-e^{-x}$ over the m samples in E_m drawn according to ν, for all D with unit length columns:*

$$Eh_{R_\lambda,D} \leq 1.1E_m h_{R_\lambda,D} + 9\frac{np\ln\left(4\lambda m\right)+x}{m}.$$

Note that the absolute loss $h_{R_\lambda,D}$ in the new bounds can be replaced with the quadratic loss $h_{R_\lambda,D}^2$ used in Theorem 6, at a small cost: an added factor of 2 inside the ln, and the same applies to many other loss functions. This applies also to the cover number based bounds given below.

Generalization bounds for k sparse representation. Proposition 3 and Corollary 4 imply certain generalization bounds for the problem of dictionary learning for k sparse representations, which we give here.

A straight forward combination of Theorem 2 of Maurer and Pontil (2010) (given here as Theorem 6) and Proposition 3 results in the following theorem.

Theorem 9 *Let $\delta < 1$ with ν a distribution on $\mathbb{S}^{n-1}$. Then with probability at least $1-e^{-x}$ over the m samples in E_m drawn according to ν, for all D s.t. $\mu_{k-1}(D) \leq \delta$ and with unit length columns:*

$$Eh_{H_k,D}^2 \leq E_m h_{H_k,D}^2 + \frac{p}{\sqrt{m}}\left(\frac{14k}{1-\delta} + \frac{1}{2}\sqrt{\ln\left(16m\left(\frac{k}{1-\delta}\right)^2\right)}\right) + \sqrt{\frac{x}{2m}}.$$

In the case of clustering we have $k = 1$ and $\delta = 0$ and this result approaches the rates of Biau et al. (2008).

The following theorems follow from the covering number bound of Corollary 4 and applying the general results of Section 3 as for the l_1 sparsity results.

Theorem 10 *Let $\delta < 1$ with ν a distribution on $\mathbb{S}^{n-1}$. Then with probability at least $1 - e^{-x}$ over the m samples in E_m drawn according to ν, for all D s.t. $\mu_{k-1}(D) \leq \delta$ and with unit length columns:*

$$Eh_{H_k,D} \leq E_m h_{H_k,D} + \sqrt{\frac{np \ln \frac{4\sqrt{m}k}{1-\delta}}{2m}} + \sqrt{\frac{x}{2m}} + \sqrt{\frac{4}{m}}.$$

Theorem 11 *Let $\delta < 1$, $np \geq 20$ and $m \geq 5000$ with ν a distribution on $\mathbb{S}^{n-1}$. Then with probability at least $1 - e^{-x}$ over the m samples in E_m drawn according to ν, for all D s.t. $\mu_{k-1}(D) \leq \delta$ and with unit length columns:*

$$Eh_{H_k,D} \leq 1.1 E_m h_{H_k,D} + 9 \frac{np \ln \left(\frac{4\sqrt{m}k}{1-\delta} \right) + x}{m}.$$

Generalization bounds for dictionary learning in feature spaces. We further consider applications of dictionary learning to signals that are not represented as elements in a vector space, or that have a very high (possibly infinite) dimension.

In addition to providing an approximate reconstruction of signals, sparse representation can also be considered as a form of analysis, if we treat the choice of non zero coefficients and their magnitude as features of the signal. In the domain of images, this has been used to perform classification (in particular, face recognition) by Wright et al. (2008). Such analysis does not require that the data itself be represented in $\mathbb{R}^n$ (or in any vector space); it is enough that the similarity between data elements is induced from an inner product in a feature space. This requirement is fulfilled by using an appropriate kernel function.

Definition 12 *Let $\mathcal{R}$ be a set of data representations, then a kernel function $\kappa : \mathcal{R}^2 \to \mathbb{R}$ and a feature mapping $\phi : \mathcal{R} \to \mathcal{H}$ are such that:*

$$\kappa(x, y) = \langle \phi(x), \phi(y) \rangle_{\mathcal{H}}$$

where $\mathcal{H}$ is some Hilbert space.

As a concrete example, choose a sequence of n words, and let ϕ map a document to the vector of counts of appearances of each word in it (also called bag of words). Treating $\kappa(a,b) = \langle \phi(a), \phi(b) \rangle$ as the similarity between documents a and b, is the well known "bag of words" approach, applicable to many document related tasks (Shawe-Taylor and Cristianini, 2004). Then the statement $\phi(a) + \phi(b) \approx \phi(c)$ does not imply that c can be reconstructed from a and b, but we might consider it indicative of the content of c. The dictionary of elements used for representation could be decided via dictionary learning, and it is natural to choose the dictionary so that the bags of words of documents are approximated well by small linear combinations of those in the dictionary.

As the example above suggests, the kernel dictionary learning problem is to find a dictionary D minimizing

$$\mathbb{E}_{x \sim \nu} h_{\phi,A,D}(x),$$

where we consider the representation error function

$$h_{\phi,A,D}(x) = \min_{a \in A} \|(\Phi D)\, a - \phi(x)\|_{\mathcal{H}},$$

in which Φ acts as ϕ on the elements of D, $A \in \{R_\lambda, H_k\}$, and the norm $\|\cdot\|_{\mathcal{H}}$ is that induced by the kernel on the feature space $\mathcal{H}$.

Analogues of all the generalization bounds mentioned so far can be replicated in the kernel setting. The dimension free results of Maurer and Pontil (2010) apply most naturally in this setting, and may be combined with our results to cover also dictionaries for k sparse representation, under reasonable assumptions on the kernel.

Proposition 13 *Let ν be any distribution on $\mathcal{R}$ such that $x \sim \nu$ implies that $\phi(x)$ is in the unit ball $B_{\mathcal{H}}$ of $\mathcal{H}$ with probability 1. Then with probability at least $1 - e^{-x}$ over the m samples in E_m drawn according to ν, for all $D \subset \mathcal{R}$ with cardinality p such that $\Phi D \subset B_{\mathcal{H}}$ and $\mu_{k-1}(\Phi D) \leq \delta < 1$:*

$$Eh^2_{\phi,H_k,D} \leq E_m h^2_{\phi,H_k,D} + \sqrt{\frac{p^2 \left(14k/(1-\delta) + 1/2\sqrt{\ln\left(16m\left(\frac{k}{1-\delta}\right)^2\right)}\right)^2}{m}} + \sqrt{\frac{x}{2m}}.$$

Note that in $\mu_{k-1}(\Phi D)$ the Babel function is defined in terms of inner products in $\mathcal{H}$, and can therefore be computed efficiently by applications of the kernel.

In Section 5 we prove the above result and also cover number bounds as in the linear case considered before. In the current setting, these bounds depend on the Hölder smoothness order α of the feature mapping ϕ. Formal definitions are given in Section 5 but as an example, the well known Gaussian kernel has $\alpha = 1$. We give now one of the generalization bounds using this method.

Theorem 14 *Let $\mathcal{R}$ have ε covers of order $(C/\varepsilon)^n$. Let $\kappa : \mathcal{R}^2 \to \mathbb{R}^+$ be a kernel function s.t. $\kappa(x,y) = \langle \phi(X), \phi(Y) \rangle$, for ϕ which is uniformly L-Hölder of order $\alpha > 0$ over $\mathcal{R}$, and let $\gamma = \max_{x \in \mathcal{R}} \|\phi(x)\|_{\mathcal{H}}$. Let $\delta < 1$, and ν any distribution on $\mathcal{R}$, then with probability at least $1 - e^{-x}$ over the m samples in E_m drawn according to ν, for all dictionaries $D \subset \mathcal{R}$ of cardinality p s.t. $\mu_{k-1}(\Phi D) \leq \delta < 1$ (where Φ acts like ϕ on columns):*

$$Eh_{H_k,D} \leq E_m h_{H_k,D} + \gamma \left(\sqrt{\frac{np \ln\left(\sqrt{m} C^\alpha \frac{k\gamma^2 L}{1-\delta}\right)}{2\alpha m}} + \sqrt{\frac{x}{2m}} \right) + \sqrt{\frac{4}{m}}.$$

The covering number bounds needed to prove this theorem and analogs for the other generalization bounds are proved in Section 5.

3. Covering Numbers of $\mathcal{G}_\lambda$ and $\mathcal{F}_{\delta,k}$

The main content of this section is the proof of Theorem 1 and Corollary 4. We also show that in the k sparse representation setting a finite bound on λ does not occur generally thus an additional restriction, such as the near-orthogonality on the set of dictionaries on which we rely in this setting,

is necessary. Lastly, we recall known results from statistical learning theory that link covering numbers to generalization bounds.

We recall the definition of the covering numbers we wish to bound. Anthony and Bartlett (1999) give a textbook introduction to covering numbers and their application to generalization bounds.

Definition 15 (Covering number) *Let (M,d) be a metric space and $S \subset M$. Then the ε covering number of S defined as $N(\varepsilon,S,d) = \min\{|A| \,|A \subset M \text{ and } S \subset (\bigcup_{a\in A} B_d(a,\varepsilon))\}$ is the size of the minimal ε cover of S using d.*

To prove Theorem 1 and Corollary 4 we first note that the space of all possible dictionaries is a subset of a unit ball in a Banach space of dimension np (with a norm specified below). Thus (see formalization in Proposition 5 of Cucker and Smale, 2002) the space of dictionaries has an ε cover of size $(4/\varepsilon)^{np}$. We also note that a uniformly L Lipschitz mapping between metric spaces converts ε/L covers into ε covers. Then it is enough to show that Ψ_λ defined as $D \mapsto h_{R_\lambda,D}$ and Φ_k defined as $D \mapsto h_{H_k,D}$ are uniformly Lipschitz (when Φ_k is restricted to the dictionaries with $\mu_{k-1}(D) \le c < 1$). The proof of these Lipschitz properties is our next goal, in the form of Lemmas 18 and 19.

The first step is to be clear about the metrics we consider over the spaces of dictionaries and of error functions.

Definition 16 (Induced matrix norm) *Let $p,q \ge 1$, then a matrix $A \in \mathbb{R}^{n\times m}$ can be considered as an operator $A : \left(\mathbb{R}^m, \|\cdot\|_p\right) \to \left(\mathbb{R}^n, \|\cdot\|_q\right)$. The p,q induced norm is $\|A\|_{p,q} \triangleq \sup_{x\in\mathbb{R}^m \|x\|_p=1} \|Ax\|_q$.*

Lemma 17 *For any matrix D, $\|D\|_{1,2}$ is equal to the maximal Euclidean norm of any column in D.*

Proof That the maximal norm of a column bounds $\|D\|_{1,2}$ can be seen geometrically; $Da/\|a\|_1$ is a convex combination of column vectors, then $\|Da\|_2 \le \max_{d_i} \|d_i\|_2 \|a\|_1$ because a norm is convex. Equality is achieved for $a = e_i$, where d_i is the column of maximal norm. ■

The images of Ψ_λ and Φ_k are sets of representation error functions—each dictionary induces a set of precisely representable signals, and a representation error function is simply a map of distances from this set. Representation error functions are clearly continuous, 1-Lipschitz, and into $[0,1]$. In this setting, a natural norm over the images is the supremum norm $\|\cdot\|_\infty$.

Lemma 18 *The function Ψ_λ is λ-Lipschitz from $\left(\mathbb{R}^{n\times m}, \|\cdot\|_{1,2}\right)$ to $C\left(\mathbb{S}^{n-1}\right)$.*

Proof Let D and D' be two dictionaries whose corresponding elements are at most $\varepsilon > 0$ far from one another. Let x be a unit signal and Da an optimal representation for it. Then $\|(D-D')a\|_2 \le \|D-D'\|_{1,2}\|a\|_1 \le \varepsilon\lambda$. If $D'a$ is very close to Da in particular it is not a much worse representation of x, and replacing it with the optimal representation under D', we have $h_{R_\lambda,D'}(x) \le h_{R_\lambda,D}(x) + \varepsilon\lambda$. By symmetry we have $|\Psi_\lambda(D)(x) - \Psi_\lambda(D')(x)| \le \lambda\varepsilon$. This holds for all unit signals, then $\|\Psi_\lambda(D) - \Psi_\lambda(D')\|_\infty \le \lambda\varepsilon$. ■

This concludes the proof of Theorem 1. We now provide a proof for Proposition 3 which is used in the corresponding treatment for covering numbers under k sparsity.

Proof (Of Proposition 3) Let D^k be a submatrix of D whose k columns from D achieve the minimum on $h_{H_k,D}(x)$ for $x \in \mathbb{S}^{n-1}$. We now consider the Gram matrix $G = \left(D^k\right)^\top D^k$ whose diagonal entries are the norms of the elements of D^k, therefore at least 1. By the Gersgorin theorem (Horn and Johnson, 1990), each eigenvalue of a square matrix is "close" to a diagonal entry of the matrix; the absolute difference between an eigenvalue and its diagonal entry is upper bounded by the sum of the absolute values of the remaining entries of the same row. Since a row in G corresponds to the inner products of an element from D^k with every element from D^k, this sum is upper bounded by δ for all rows. Then we conclude the eigenvalues of the Gram matrix are lower bounded by $1-\delta > 0$. Then in particular G has a symmetric inverse G^{-1} whose eigenvalues are positive and bounded from above by $1/(1-\delta)$. The maximal magnitude of an eigenvalue of a symmetric matrix coincides with its induced norm $\|\cdot\|_{2,2}$, therefore $\left\|G^{-1}\right\|_{2,2} \leq 1/(1-\delta)$.

Linear dependence of elements of D^k would imply a non-trivial nullspace for the invertible G. Then the elements of D^k are linearly independent, which implies that the unique optimal representation of x as a linear combination of the columns of D^k is $D^k a$ with

$$a = \left(\left(D^k\right)^\top D^k\right)^{-1}\left(D^k\right)^\top x.$$

Using the above and the definition of induced matrix norms, we have

$$\|a\|_2 \leq \left\|\left(\left(D^k\right)^\top D^k\right)^{-1}\right\|_{2,2}\left\|\left(D^k\right)^\top x\right\|_2 \leq (1-\delta)^{-1}\left\|\left(D^k\right)^\top x\right\|_2.$$

The vector $\left(D^k\right)^\top x$ is in $\mathbb{R}^k$ and by the Cauchy Schwartz inequality $\langle d_i, x\rangle \leq \gamma$, then $\left\|\left(D^k\right)^\top x\right\|_2 \leq \sqrt{k}\left\|\left(D^k\right)^\top x\right\|_\infty \leq \sqrt{k}\gamma$. Since only k entries of a are non zero, $\|a\|_1 \leq \sqrt{k}\|a\|_2 \leq k\gamma/(1-\delta)$. ■

Lemma 19 *The function Φ_k is a $k/(1-\delta)$-Lipschitz mapping from the set of normalized dictionaries with $\mu_{k-1}(D) < \delta$ with the metric induced by $\|\cdot\|_{1,2}$ to $C\left(\mathbb{S}^{n-1}\right)$.*

The proof of this lemma is the same as that of Lemma 18, except that a is taken to be an optimal representation that fulfills $\|a\|_1 \leq \lambda = k/(1-\mu_{k-1}(D))$, whose existence is guaranteed by Proposition 3. As outlined in the beginning of the current section, this concludes the proof of Corollary 4.

The next theorem shows that unfortunately, Φ is *not* uniformly L-Lipschitz for any constant L, requiring its restriction to an appropriate subset of the dictionaries.

Theorem 20 *For any $1 < k < n, p$, there exists $c > 0$ and q, such that for every $\varepsilon > 0$, there exist D, D' such that $\|D - D'\|_{1,2} < \varepsilon$ but $|(h_{H_k,D}(q) - h_{H_k,D'}(q))| > c$.*

Proof First we show that for any dictionary D there exist $c > 0$ and $x \in \mathbb{S}^{n-1}$ such that $h_{H_k,D}(x) > c$. Let $\nu_{\mathbb{S}^{n-1}}$ be the uniform probability measure on the sphere, and A_c the probability assigned by it to the set within c of a k dimensional subspace. As $c \searrow 0$, A_c also tends to zero, then there exists $c > 0$ s.t. $\binom{p}{k}A_c < 1$. Then for that c and any dictionary D there exists a set of positive measure on which

$h_{H_k,D} > c$, let q be a point in this set. Since $h_{H_k,D}(x) = h_{H_k,D}(-x)$, we may assume without loss of generality that $\langle e_1, q\rangle \geq 0$.

We now fix the dictionary D; its first $k-1$ elements are the standard basis $\{e_1,\ldots,e_{k-1}\}$, its kth element is $d_k = \sqrt{1-\varepsilon^2/4}e_1 + \varepsilon e_k/2$, and the remaining elements are chosen arbitrarily. Now construct D' to be identical to D except its kth element is $v = \sqrt{1-\varepsilon^2/4}e_1 + lq$ choosing l so that $\|v\|_2 = 1$. Then there exist $a,b \in \mathbb{R}$ such that $q = aD'_1 + bD'_k$ and we have $h_{H_k,D'}(q) = 0$, fulfilling the second part of the theorem. On the other hand, since $\langle e_1, q\rangle \geq 0$, we have $l \leq \varepsilon/2$, and then we find $\|D - D'\|_{1,2} = \|\varepsilon e_k/2 - lq\|_2 \leq \|\varepsilon e_k/2\| + \|lq\| = \varepsilon/2 + l \leq \varepsilon$. ■

To conclude the generalization bounds of Theorems 7, 8, 10, 11 and 14 from the covering number bounds we have provided, we use the following results. Both specialize well known results to the case of l_∞ cover number bounds, thereby improving constants and simplifying the proofs. The first proof is simple enough we include it at the end of this section. The second result[1] (along with its corollary) gives fast rate bounds as in the more general results by Mendelson (2003) and Bartlett et al. (2005).

Lemma 21 *Let $\mathcal{F}$ be a class of $[0,B]$ functions with covering number bound $(C/\varepsilon)^d > e/B^2$ under the supremum norm. Then for every $x > 0$, with probability of at least $1-e^{-x}$ over the m samples in E_m chosen according to ν, for all $f \in \mathcal{F}$:*

$$Ef \leq E_m f + B\left(\sqrt{\frac{d\ln(C\sqrt{m})}{2m}} + \sqrt{\frac{x}{2m}}\right) + \sqrt{\frac{4}{m}}.$$

Proposition 22 *Let $\mathcal{F}$ be a class of $[0,1]$ functions that can be covered for any $\varepsilon > 0$ by at most $(C/\varepsilon)^d$ balls of radius ε in the L_∞ metric where $C \geq e$ and $\beta > 0$. Then with probability at least $1-\exp(-x)$, we have for all $f \in \mathcal{F}$:*

$$Ef \leq (1+\beta)E_m f + K(d,m,\beta)\frac{d\ln(Cm)+x}{m},$$

where $K(d,m,\beta) = \sqrt{2\left(\frac{9}{\sqrt{m}}+2\right)\left(\frac{d+3}{3d}\right)+1} + \left(\frac{9}{\sqrt{m}}+2\right)\left(\frac{d+3}{3d}\right) + 1 + \frac{1}{2\beta}$.

The corollary we use to obtain Theorems 8 and 11 follows because $K(d,m,\beta)$ is non-increasing in d,m.

Corollary 23 *Let $\mathcal{F}, x$ be as above. For $d \geq 20$, $m \geq 5000$ and $\beta = 0.1$ we have with probability at least $1-\exp(-x)$ for all $f \in \mathcal{F}$:*

$$Ef \leq 1.1E_m f + 9\frac{d\ln(Cm)+x}{m}.$$

Proof (Of Lemma 21) We wish to bound $\sup_{f\in\mathcal{F}} Ef - E_m f$. Take $\mathcal{F}_\varepsilon$ to be a minimal ε cover of $\mathcal{F}$, then for an arbitrary f, denoting f_ε an ε close member of $\mathcal{F}_\varepsilon$, $Ef - E_m f \leq Ef_\varepsilon - E_m f_\varepsilon + 2\varepsilon$. In particular, $\sup_{f\in\mathcal{F}} Ef - E_m f \leq 2\varepsilon + \sup_{f\in\mathcal{F}_\varepsilon} Ef - E_m f$. To bound the supremum on the now finite

1. We thank Andreas Maurer for suggesting this result and a proof elaborated in Appendix A.

class of functions, note that $Ef - E_m f$ is an average of m independent copies of the identical zero mean bounded variable $Ef - E_1 f$.

Applying Hoeffding's inequality, we have $P\left(Ef - E_m f > t\right) \leq \exp\left(-2mB^{-2}t^2\right)$.

The probability that any of the $|\mathcal{F}_\varepsilon|$ differences under the supremum is larger than t may be bounded as $P\left(\sup_{f\in\mathcal{F}_\varepsilon} Ef - E_m f \geq t\right) \leq |\mathcal{F}_\varepsilon| \cdot \exp\left(-2mB^{-2}t^2\right) \leq \exp\left(d\ln\left(C/\varepsilon\right) - 2mB^{-2}t^2\right)$.

In order to control the probability with x as in the statement of the lemma, we take $-x = d\ln\left(C/\varepsilon\right) - 2mB^{-2}t^2$ or equivalently we choose $t = \sqrt{B^2/2m}\sqrt{d\ln\left(C/\varepsilon\right) + x}$. Then with probability $1 - e^{-x}$ we bound $\sup_{f\in\mathcal{F}} Ef - E_m f \leq 2\varepsilon + t$. Using the covering number bound assumption and the sublinearity of $\sqrt{\cdot}$, we have by $\sup_{f\in\mathcal{F}} Ef - E_m f \leq 2\varepsilon + B\left(\sqrt{d\ln\left(C/\varepsilon\right)/2m} + \sqrt{x/2m}\right)$. The proof is completed by taking $\varepsilon = 1/\sqrt{m}$. ■

4. On the Babel Function

The Babel function is one of several metrics defined in the sparse representations literature to quantify an "almost orthogonality" property that dictionaries may enjoy. Such properties have been shown to imply theoretical properties such as uniqueness of the optimal k sparse representation. In the algorithmic context, Donoho and Elad (2003) and Tropp (2004) use the Babel function to show that particular efficient algorithms for finding sparse representations fulfill certain quality guarantees when applied to such dictionaries. This reinforces the practical importance of the learnability of this class of dictionary. We proceed to discuss some elementary properties of the Babel function, and then state a bound on the proportion of dictionaries having sufficiently good Babel function.

Measures of orthogonality are typically defined in terms of inner products between the elements of the dictionary. Perhaps the simplest of these measures of orthogonality is the following special case of the Babel function.

Definition 24 *The coherence of a dictionary D is $\mu_1(D) = \max_{i\neq j}\left|\left\langle d_i, d_j\right\rangle\right|$.*

The proof of Proposition 3 demonstrates that the Babel function quantifies the effects of non orthogonality on the representation of a signal with particular level $k+1$ of sparsity. Is enough to bound the Babel function using coherence? only at a cost of significantly tightening our requirements on dictionaries. While the coherence and Babel measures are indeed related by the inequalities

$$\mu_1\left(D\right) \leq \mu_k\left(D\right) \leq k\mu_1\left(D\right),$$

the factor k gap between the bounds cannot be improved. The tightness of the right inequality is witnessed by a dictionary including $k+1$ copies of the same element. That of the left inequality is witnessed by the following example. Let D consist of k pairs of elements, so that the subspace spanned by each pair is orthogonal to all other elements, and such that the inner product between the elements of any single pair is half. In this case $\mu_k(D) = \mu_1(D) = 1/2$. However note that to ensure $\mu_k < 1$ only restricting μ_1 requires the constraint $\mu_1(D) < 1/k$, which is not fulfilled in our example.

To better understand $\mu_k\left(D\right)$, we consider first its extreme values. When $\mu_k\left(D\right) = 0$, for any $k > 1$, this means that D is an orthogonal set (therefore $p \leq n$). The maximality of $\mu_k\left(D\right) = k$ we have seen before.

A well known generic class of dictionaries with more elements than a basis is that of *frames* (see Duffin and Schaeer, 1952), which includes many wavelet systems and filter banks. Some frames can be trivially seen to fulfill our condition on the Babel function.

Proposition 25 *Let $D \in \mathbb{R}^{n\times p}$ be a frame of $\mathbb{R}^n$, so that for every $v \in \mathbb{S}^{n-1}$ we have that $\sum_{i=1}^{n} |\langle v, d_i\rangle|^2 \leq B$, with $\|d_i\|_2 = 1$ for all i, and $B < 1 + 1/k$. Then $\mu_{k-1}(D) < 1$.*

This may be easily verified by considering the inner products of any dictionary element with any other k elements as a vector in $\mathbb{R}^k$; the frame condition bounds its squared Euclidean norm by $B-1$ (we remove the inner product of the element with itself in the frame expression). Then use the equivalence of l_1 and l_2 norms.

4.1 Proportion of Dictionaries with $\mu_{k-1}(D) < \delta$

We return to the question of the prevalence of dictionaries having $\mu_{k-1} < \delta$. Are almost all dictionaries such? If the answer is affirmative, it implies that Theorem 11 is quite strong, and representation finding algorithms such as basis pursuit are almost always exact, which might help prove properties of dictionary learning algorithms. If the opposite is true and few dictionaries have low Babel function, the results of this paper are weak. While there might be better probability measures on the space of dictionaries, we consider one that seems natural: suppose that a dictionary D is constructed by choosing p unit vectors uniformly from $\mathbb{S}^{n-1}$; what is the probability that $\mu_{k-1}(D) < \delta$? how does this depend on p, k?

Theorem 5 gives us the following answer to these questions. Asymptotically *almost all dictionaries under the uniform measure are learnable with $\tilde{O}(np)$ examples*, as long as $k \ln p = o(\sqrt{n})$.

5. Dictionary Learning in Feature Spaces

We propose in Section 2 a scenario in which dictionary learning is performed in a feature space corresponding to a kernel function. Here we show how to adapt the different generalization bounds discussed in this paper for the particular case of $\mathbb{R}^n$ to more general feature spaces, and the dependence of the sample complexities on the properties of the kernel function or the corresponding feature mapping. We begin with the relevant specialization of the results of Maurer and Pontil (2010) which have the simplest dependence on the kernel, and then discuss the extensions to k sparse representation and to the cover number techniques presented in the current work.

A general feature space, denoted $\mathcal{H}$, is a Hilbert space to which Theorem 6 applies as is, under the simple assumption that the dictionary elements and signals are in its unit ball; this assumption is guaranteed by some kernels such as the Gaussian kernel. Then we take ν on the unit ball of $\mathcal{H}$ to be induced by some distribution ν' on the domain of the kernel, and the theorem applies to any such ν' on $\mathcal{R}$. Nothing more is required if the representation is chosen from R_λ. The corresponding generalization bound for k sparse representations when the dictionary elements are nearly orthogonal in the feature space is given in Proposition 13.

Proof (Of Proposition 13) Proposition 3 applies with the Euclidean norm of $\mathcal{H}$, and $\gamma = 1$. We apply Theorem 6 with $\lambda = k/(1-\delta)$. ■

The results so far show that generalization in dictionary learning can occur despite the potentially infinite dimension of the feature space, without considering practical issues of representation

and computation. We now make the domain and applications of the kernel explicit in order to address a basic computational question, and allow the use of cover number based generalization bounds to prove Theorem 14. We now consider signals represented in a metric space $(\mathcal{R},d)$, in which similarity is measured by the kernel κ corresponding to the feature map $\phi : \mathcal{R} \to \mathcal{H}$. The elements of a dictionary D are now from $\mathcal{R}$, and we denote ΦD their mapping by ϕ to $\mathcal{H}$. The representation error function used is $h_{\phi,A,D}$.

We now show that the approximation error in the feature space is a quadratic function of the coefficient vector; the quadratic function for particular D and x may be found by applications of the kernel.

Proposition 26 *Computing the representation error at a given x,a,D requires $O\left(p^2\right)$ kernel applications in general, and only $O\left(k^2+p\right)$ when a is k sparse.*

The squared error expands to

$$\sum_{i=1}^{p} a_i \sum_{j=1}^{p} a_j \kappa\left(d_i,d_j\right) + \kappa\left(x,x\right) - 2\sum_{i=1}^{p} a_i \kappa\left(x,d_i\right).$$

We note that the k sparsity constraint on a poses algorithmic difficulties beyond those addressed here. Some of the common approaches to these, such as orthogonal matching pursuit (Chen et al., 1989), also depend on the data only through their inner products, and may therefore be adapted to the kernel setting.

The cover number bounds depend strongly on the dimension of the space of dictionary elements. Taking $\mathcal{H}$ as the space of dictionary elements is the simplest approach, but may lead to vacuous or weak bounds, for example in the case of the Gaussian kernel whose feature space is infinite dimensional. Instead we propose to use the space of data representations $\mathcal{R}$, whose dimensions are generally bounded by practical considerations. In addition, we will assume that the kernel is not "too wild" in the following sense.

Definition 27 *Let $L,\alpha > 0$, and let (A,d') and (B,d) be metric spaces. We say a mapping $f : A \to B$ is uniformly L Hölder of order α on a set $S \subset A$ if $\forall x,y \in S$, the following bound holds:*

$$d\left(f(x),f(y)\right) \leq L \cdot d'(x,y)^{\alpha}.$$

The relevance of this smoothness condition is as follows.

Lemma 28 *A Hölder function maps an ε cover of S to an $L\varepsilon^{\alpha}$ cover of its image $f(S)$. Thus, to obtain an ε cover of the image of S, it is enough to begin with an $(\varepsilon/L)^{1/\alpha}$ cover of S.*

A Hölder feature map ϕ allows us to bound the cover numbers of the dictionary elements in $\mathcal{H}$ using their cover number bounds in $\mathcal{R}$. Note that not every kernel corresponds to a Hölder feature map (the Dirac δ kernel is a counter example: any two distinct elements are mapped to elements at a mutual distance of 1), and sometimes analyzing the feature map is harder than analyzing the kernel. The following lemma bounds the geometry of the feature map using that of the kernel.

Lemma 29 *Let $\kappa(x,y) = \langle \phi(x), \phi(y) \rangle$, and assume further that κ fulfills a Hölder condition of order α uniformly in each parameter, that is, $|\kappa(x,y) - \kappa(x+h,y)| \leq L\|h\|^{\alpha}$. Then ϕ uniformly fulfills a Hölder condition of order $\alpha/2$ with constant $\sqrt{2L}$.*

This result is not sharp. For example, for the Gaussian case, both kernel and the feature map are Hölder order 1.

Proof Using the Hölder condition, we have that $\|\phi(x)-\phi(y)\|_{\mathcal{H}}^2 = \kappa(x,x)-\kappa(x,y)+\kappa(y,y)-\kappa(x,y) \leq 2L\|x-y\|^{\alpha}$. All that remains is to take the square root of both sides. ■

For a given feature mapping ϕ, set of representations $\mathcal{R}$, we define two families of function classes so:

$$\begin{aligned} \mathcal{W}_{\phi,\lambda} &= \left\{h_{\phi,R_\lambda,D} : D \in \mathcal{D}^p\right\} \text{ and} \\ Q_{\phi,k,\delta} &= \left\{h_{\phi,H_k,D} : D \in \mathcal{D}^p \wedge \mu_{k-1}(\Phi D) \leq \delta\right\}. \end{aligned}$$

The next proposition completes this section by giving the cover number bounds for the representation error function classes induced by appropriate kernels, from which various generalization bounds easily follow, such as Theorem 14.

Proposition 30 *Let $\mathcal{R}$ be a set of representations with a cover number bound of $(C/\varepsilon)^n$, and let either ϕ be uniformly L Hölder condition of order α on $\mathcal{R}$, or κ be uniformly L Hölder of order 2α on $\mathcal{R}$ in each parameter, and let $\gamma = \sup_{d\in\mathcal{R}}\|\phi(d)\|_{\mathcal{H}}$. Then the function classes $\mathcal{W}_{\phi,\lambda}$ and $Q_{\phi,k,\delta}$ taken as metric spaces with the supremum norm, have ε covers of cardinalities at most $\left(C\left(\lambda\gamma L/\varepsilon\right)^{1/\alpha}\right)^{np}$ and $\left(C\left(k\gamma^2 L/\left(\varepsilon\left(1-\delta\right)\right)\right)^{1/\alpha}\right)^{np}$, respectively.*

Proof We first consider the case of l_1 constrained coefficients. If $\|a\|_1 \leq \lambda$ and $\max_{d\in\mathcal{D}}\|\phi(d)\|_{\mathcal{H}} \leq \gamma$ then by considerations applied in Section 3, to obtain an ε cover of the image of dictionaries $\{\min_a \|(\Phi D)a-\phi(x)\|_{\mathcal{H}} : D \in \mathcal{D}\}$, it is enough to obtain an $\varepsilon/(\lambda\gamma)$ cover of $\{\Phi D : D \in \mathcal{D}\}$. If also the feature mapping ϕ is uniformly L Hölder of order α over $\mathcal{R}$ then an $(\lambda\gamma L/\varepsilon)^{-1/\alpha}$ cover of the set of dictionaries is sufficient, which as we have seen requires at most $\left(C\left(\lambda\gamma L/\varepsilon\right)^{1/\alpha}\right)^{np}$ elements.

In the case of l_0 constrained representation, the bound on λ due to Proposition 3 is $\gamma k(1-\delta)$, and the result follows from the above by substitution. ■

6. Conclusions

Our work has several implications on the design of dictionary learning algorithms as used in signal, image, and natural language processing. First, the fact that generalization is only logarithmically dependent on the l_1 norm of the coefficient vector widens the set of applicable approaches to penalization. Second, in the particular case of k sparse representation, we have shown that the Babel function is a key property for the generalization of dictionaries. It might thus be useful to modify dictionary learning algorithms so that they obtain dictionaries with low Babel functions, possibly through regularization or through certain convex relaxations. Third, mistake bounds (e.g., Mairal et al. 2010) on the quality of the solution to the coefficient finding optimization problem may lead to generalization bounds for practical algorithms, by tying such algorithms to k sparse representation.

The upper bounds presented here invite complementary lower bounds. The existing lower bounds for $k = 1$ (vector quantization) and for $k = p$ (representation using PCA directions) are applicable, but do not capture the geometry of general k sparse representation, and in particular do not clarify the effective dimension of the unrestricted class of dictionaries for it. We have not excluded the possibility that the class of unrestricted dictionaries has the same dimension as that of those with a small Babel function. The best upper bound we know for the larger class, being the trivial one of order $O\left(\binom{p}{k} n^2 / m\right)$, leaves a significant gap for future exploration.

We view the dependence on μ_{k-1} from an "algorithmic luckiness" perspective (Herbrich and Williamson, 2003): if the data are described by a dictionary with low Babel function the generalization bounds are encouraging.

Acknowledgments

We thank Shahar Mendelson for helpful discussions. A.B. was partly supported by the European Communitys FP7-FET program, SMALL project, under grant agreement no. 225913. S.M and D.V. were partially supported by the ISF under contract 890015.

Appendix A. Generalization with Fast Rates

In this appendix we give a proof, essentially due to Andreas Maurer, of the fast rates result Proposition 22. The assumption of l_∞ cover numbers allows a much simpler argument than that in the more general results by Mendelson (2003) and Bartlett et al. (2005), which also leads to better constants for this case.

Proof (Of Proposition 22) We take $\mathcal{G}$ to be an $\frac{1}{m}$ cover of $\mathcal{F}$ as guaranteed by the assumption. Then for any $f \in \mathcal{F}$, there exists $g \in \mathcal{G}$ such that $\|f - g\|_\infty \leq \frac{1}{m}$, and Lemmas 31 and 33 apply. we have with probability at least $1 - \exp(-x)$, for every $f \in \mathcal{F}$:

$$\begin{aligned} Ef - E_m f &\leq Eg + \frac{1}{m} - \left(E_m g - \frac{1}{m}\right) && (4)\\ &= \frac{2}{m} + Eg - E_m g \\ &\leq \frac{2}{m} + \sqrt{\frac{2 \operatorname{Var} g \left(d \ln(Cm) + x\right)}{m}} + \frac{2\left(d \ln(Cm) + x\right)}{3m} && (5)\\ &\leq \frac{2}{m} + \left(\sqrt{\operatorname{Var} f} + \frac{2}{m}\right) \sqrt{\frac{2\left(d \ln(Cm) + x\right)}{m}} + \frac{2\left(d \ln(Cm) + x\right)}{3m} && (6) \end{aligned}$$

Inequality (4) follows from Lemma 33 and

$$Ef \leq Eg + \frac{1}{m} \text{ and } E_m f \geq E_m g - \frac{1}{m}.$$

Inequality (5) follows from Lemma 31:

$$Pr\left(\exists g \in \mathcal{G} : Eg > E_m g + \sqrt{\frac{2 \operatorname{Var} g \left(d \ln(Cm) + x\right)}{m}} + \frac{2\left(d \ln(Cm) + x\right)}{3m}\right) \leq e^{-x}.$$

Inequality (6) follows from Lemma 33 because

$$\sqrt{\frac{2\operatorname{Var} g\left(d\ln\left(Cm\right)+x\right)}{m}}=\sqrt{\operatorname{Var} g}\sqrt{\frac{2\left(d\ln\left(Cm\right)+x\right)}{m}}\leq\left(\sqrt{\operatorname{Var} f}+\frac{2}{m}\right)\sqrt{\frac{2\left(d\ln\left(Cm\right)+x\right)}{m}}.$$

After slight rearrangement, we have

$$\begin{aligned}
Ef-E_mf &\leq \sqrt{\frac{2\operatorname{Var} f\left(d\ln\left(Cm\right)+x\right)}{m}}+\frac{2}{m}\sqrt{\frac{2\left(d\ln\left(Cm\right)+x\right)}{m}}+\frac{2\left(d\ln\left(Cm\right)+x\right)}{3m}+\frac{2}{m}\\
&\leq \sqrt{\frac{2\operatorname{Var} f\left(d\ln\left(Cm\right)+x\right)}{m}}+\left(\frac{9}{\sqrt{m}}+2\right)\frac{\left(d\ln\left(Cm\right)+x\right)}{3m}+\frac{2}{m} \qquad (7)\\
&\leq \sqrt{\frac{2Ef\left(d\ln\left(Cm\right)+x\right)}{m}}+\left(\frac{9}{\sqrt{m}}+2\right)\frac{d\ln\left(Cm\right)+x}{3m}+\frac{2}{m} \qquad (8)\\
&\leq \sqrt{\frac{2Ef\left(d\ln\left(Cm\right)+x\right)}{m}}+\left(\frac{9}{\sqrt{m}}+2\right)\left(\frac{d+3}{3d}\right)\frac{d\ln\left(Cm\right)+x}{m} \qquad (9)
\end{aligned}$$

Simple algebra, the fact that $\operatorname{Var} f\leq Ef$ for a $[0,1]$ valued function f and Lemma 37 respectively justify inequalities (7), (8) and (9).

For convenience, we denote $K=\left(\frac{9}{\sqrt{m}}+2\right)\left(\frac{d+3}{3d}\right)$. We also denote $A=E_mf+K\frac{d\ln(Cm)+x}{m}$ and $B=\left(d\ln\left(Cm\right)+x\right)/m$, and note we have shown that with probability at least $1-\exp\left(-x\right)$ we have $Ef-A\leq\sqrt{2BEf}$, which by Lemma 34 implies $Ef\leq A+B+\sqrt{2AB+B^2}$. By substitution and Lemma 36 we conclude that then:

$$\begin{aligned}
Ef &\leq A+B+\sqrt{2AB+B^2}\\
&= E_mf+K\frac{d\ln\left(Cm\right)+x}{m}+B+\sqrt{2BE_mf+2BK\frac{d\ln\left(Cm\right)+x}{m}+B^2}\\
&\leq E_mf+K\frac{d\ln\left(Cm\right)+x}{m}+B+\sqrt{2BE_mf}+\sqrt{2BK\frac{d\ln\left(Cm\right)+x}{m}+B^2}\\
&= E_mf+\sqrt{2E_mf\frac{d\ln\left(Cm\right)+x}{m}}+\left(\sqrt{\left(2K+1\right)}+K+1\right)\frac{d\ln\left(Cm\right)+x}{m}
\end{aligned}$$

using Lemma 36 for the second inequality.

From Lemma 35 with $a=E_mf$ and $b=2\left(d\ln\left(Cm\right)+x\right)/m$ we find that for every $\lambda>0$

$$\sqrt{2E_mf\left(d\ln\left(Cm\right)+x\right)/m}\leq\lambda E_mf+\frac{1}{2\lambda}\left(d\ln\left(Cm\right)+x\right)/m$$

and the proposition follows. ■

The following lemma encapsulates the probabilistic part of the analysis.

Lemma 31 *Let $\mathcal{G}$ be a class of $[0,1]$ functions, of finite cardinality $|\mathcal{G}|\leq(Cm)^d$. Then*

$$Pr\left(\exists g\in\mathcal{G}:Eg>E_mg+\sqrt{\frac{2\operatorname{Var} g\left(d\ln\left(Cm\right)+x\right)}{m}}+\frac{2\left(d\ln\left(Cm\right)+x\right)}{3m}\right)\leq e^{-x}.$$

In proving Lemma 31, we use the following well known fact, which we recall for its notations.

Lemma 32 (Bernstein Inequality) *Let X_i be independent zero mean variables with $|X_i| \leq c$ almost surely then $Pr\left(\frac{1}{m}\sum_{i=1}^{m} X_i > \varepsilon\right) \leq \exp\left(-\frac{m\varepsilon^2}{2\sigma^2+2c\varepsilon/3}\right)$*

Proof (Of Lemma 31) Denote $X_i = Eg - g(s_i)$, where $\{s_i\}_{i=1}^{m}$ is a set of IID random variables, we recall the notation $E_m g = (1/m)\sum_{i=1}^{m} g(s_i)$, then $\sum_{i=1}^{m} X_i = \sum_{i=1}^{m}(Eg - g(s_i)) = mEg - \sum_{i=1}^{m} g(s_i) = m(Eg - E_m) \Rightarrow \frac{1}{m}\sum_{i=1}^{m} X_i = Eg - E_m$.

Using the fact our X_i are *IID* and the translation invariance of variance, we have

$$\begin{aligned}\sigma^2 &= \frac{1}{m}\sum_{i=1}^{m} \mathrm{Var}(X_i)\\ &= \mathrm{Var}(X_i)\\ &= \mathrm{Var}(Eg - g(s_i))\\ &= \mathrm{Var}\, g.\end{aligned}$$

Since $\|g\|_\infty \leq 1$, we also know$|X_i| \leq 1$.

Applying the Bernstein Inequality we get $Pr(Eg - E_m g > \varepsilon) \leq \exp\left(-\frac{m\varepsilon^2}{2\,\mathrm{Var}\, g+2\varepsilon/3}\right)$ for any $\varepsilon > 0$. We wish to bound the probability of a large deviation by $\exp(-y)$, so it is enough for ε to satisfy:

$$\begin{aligned}\exp\left(-\frac{m\varepsilon^2}{2\,\mathrm{Var}\, g+2\varepsilon/3}\right) \leq \exp(-y) &\iff -\frac{m\varepsilon^2}{2\,\mathrm{Var}\, g+2\varepsilon/3} \leq -y\\ &\iff y \leq \frac{m\varepsilon^2}{2\,\mathrm{Var}\, g+2\varepsilon/3}\\ &\iff y\left(2\,\mathrm{Var}\, g + \frac{2\varepsilon}{3}\right) \leq m\varepsilon^2\\ &\iff 0 \leq \varepsilon^2 - \frac{2y}{3m}\varepsilon - \frac{2y\,\mathrm{Var}\, g}{m}.\end{aligned}$$

This quadratic inequality in ε has the roots: $\left(2y/(3m) \pm \sqrt{(2y/(3n))^2 + 8y\,\mathrm{Var}\, g/m}\right)/2$ and a positive coefficient for ε^2, then we require ε to not be between the roots. The root closer to $-\infty$ is always negative because $\sqrt{(2y/(3m))^2 + 8y\,\mathrm{Var}\, g/m} \geq \sqrt{(2y/(3m))^2} = 2y/(3m)$, but the other is always strictly positive, so it is enough to take ε greater than both. In particular, by Lemma 36, we may choose $\varepsilon = 2y/(3m) + \sqrt{2y\,\mathrm{Var}\, g/m} \geq \left(2y/(3m) \pm \sqrt{(2y/(3m))^2 + 8y\,\mathrm{Var}\, g/m}\right)/2$, and conclude that

$$Pr\left(Eg - E_n g > \frac{2y}{3m} + \sqrt{\frac{2y\,\mathrm{Var}\, g}{m}}\right) \leq \exp(-y).$$

Taking a union bound over all $|\mathcal{G}|$, we have:

$$Pr\left(\exists g\in\mathcal{G}: Eg-E_m g > y/\frac{2y}{3m}+\sqrt{\frac{2y\operatorname{Var} g}{m}}\right)\leq|\mathcal{G}|\exp(-y) \iff$$
$$Pr\left(\exists g\in\mathcal{G}: Eg-E_m g > \frac{2y}{3m}+\sqrt{\frac{2y\operatorname{Var} g}{m}}\right)\leq\exp(\ln|\mathcal{G}|-y)\Rightarrow$$
$$Pr\left(\exists g\in\mathcal{G}: Eg-E_m g > \frac{2y}{3m}+\sqrt{\frac{2y\operatorname{Var} g}{m}}\right)\leq\exp\left(\ln(Cm)^d-y\right).$$

Then we take $-x=\ln(Cm)^d-y \iff y=\ln(Cm)^d+x$ and have:

$$Pr\left(\exists g\in\mathcal{G}: Eg-E_m g > \frac{2d\ln(Cm)+x}{3m}+\sqrt{\frac{2\operatorname{Var} g\left(\ln(Cm)^d+x\right)}{m}}\right)\leq\exp(-x).$$

■

Lemma 33 *Let $\|f-g\|_\infty\leq\varepsilon$. Then under any distribution we have $|Ef-Eg|\leq\varepsilon$, and $\sqrt{\operatorname{Var} g}-\sqrt{\operatorname{Var} f}\leq 2\varepsilon$*

Proof The first part is clear. For the second, we need mostly the triangle inequality for norms:

$$\begin{aligned}\sqrt{\operatorname{Var} f}-\sqrt{\operatorname{Var} g} &= \sqrt{E(f-Ef)^2}-\sqrt{E(g-Eg)^2}\\ &= \|f-Ef\|_{L_2}-\|g-Eg\|_{L_2}\\ &\leq \|f-Ef-g+Eg\|_{L_2}\\ &\leq \|f-g\|_{L_2}+\|Ef-Eg\|_{L_2}\\ &\leq \|f-g\|_\infty+\|E(f-g)\|_{L_2}\\ &\leq 2\|f-g\|_\infty\\ &\leq 2\varepsilon.\end{aligned}$$

■

Lemma 34 *If $A,B\geq 0$ and $Ef-A\leq\sqrt{2EfB}$ then $Ef\leq A+B+\sqrt{2AB+B^2}$.*

Proof First note that if $Ef < A$, we are done, because $A, B \geq 0$, then we assume $Ef \geq A$. Squaring both sides of $Ef - A \leq \sqrt{2EfB}$ we find

$$\begin{aligned}
(Ef-A)^2 \leq 2EfB &\iff (Ef)^2 - 2EfA + A^2 \leq 2EfB \\
&\iff (Ef)^2 - 2EfA - 2EfB \leq -A^2 \\
&\iff (Ef)^2 - 2EfA - 2EfB + (A+B)^2 \leq -A^2 + (A+B)^2 \\
&\iff (Ef)^2 - 2Ef(A+B) + (A+B)^2 \leq -A^2 + (A+B)^2 \\
&\iff (Ef - (A+B))^2 \leq -A^2 + (A+B)^2 \\
&\left(\sqrt{\cdot} \text{ of non-negative expressions}\right) \\
&\iff Ef - (A+B) \leq +\sqrt{(A+B)^2 - A^2} \\
&\iff Ef \leq (A+B) + \sqrt{2AB + B^2}.
\end{aligned}$$

■

We omit the easy proofs of the next two lemmata.

Lemma 35 *For* $\beta > 0$, $\sqrt{2ab} \leq \beta a + \frac{b}{2\beta}$.

Lemma 36 *For any* $a, b \geq 0$, $\sqrt{a+b} \leq \sqrt{a} + \sqrt{b}$

Lemma 37 *For* $d, m \geq 1$, $x \geq 0$ *and* $C \geq e$ *we have*

$$\left(\frac{9}{\sqrt{m}} + 2\right) \frac{d \ln(Cm) + x}{3m} + \frac{2}{m} \leq \left(\frac{9}{\sqrt{m}} + 2\right) \left(\frac{d+3}{3d}\right) \frac{d \ln(Cm) + x}{m}.$$

Proof By the assumptions, $\frac{9}{\sqrt{m}} + 2 \geq 2$ (fact(a)) and $d \leq d \ln(Cm) + x$ (fact (b)). Then

$$\begin{aligned}
\left(\frac{9}{\sqrt{m}} + 2\right) \frac{d \ln(Cm) + x}{3m} + \frac{2}{m} &= \left(\frac{9}{\sqrt{m}} + 2\right) \frac{d \ln(Cm) + x}{3m} + \frac{2}{m} \\
&\leq \left(\frac{9}{\sqrt{m}} + 2\right) \frac{d \ln(Cm) + x + 3}{3m} \\
&= \left(\frac{9}{\sqrt{m}} + 2\right) \frac{d \ln(Cm) + x + \frac{3}{d} d}{3m} \\
&\leq \left(\frac{9}{\sqrt{m}} + 2\right) \frac{d \ln(Cm) + x + \frac{3}{d}(d \ln(Cm) + x)}{3m} \\
&= \left(\frac{9}{\sqrt{m}} + 2\right) \left(\frac{d+3}{3d}\right) \frac{d \ln(Cm) + x}{m}.
\end{aligned}$$

■

Appendix B. Proof of Theorem 5

In the proof we will use an isoperimetric inequality about the sphere in high dimensions.

Definition 38 *The* ε *expansion of a set* S *in a metric space* (X,d) *is defined as*

$$S_\varepsilon = \{x \in X | d(x,S) \leq \varepsilon\},$$

where $d(x,A) = \inf_{a\in A} d(x,a)$.

Lemma 39 (Lévy's isoperimetric inequality 1951) *Let* C *be one half of* $\mathbb{S}^{n-1}$, *then* $\mu\left(\left(\mathbb{S}^{n-1}\backslash C_\varepsilon\right)\right) \leq \sqrt{\frac{\pi}{8}} \exp\left(-\frac{(n-2)\varepsilon^2}{2}\right)$.

Proof (Of Theorem 5) For any $p \in \mathbb{N}$ we denote $[p] = \{1,\ldots,p\}$ and for $i \in [p]$, we define $W_i = \max_{\Lambda\subset[p]\backslash i,|\Lambda|=k} \sum_{\lambda\in\Lambda} |\langle d_i, d_\lambda\rangle|$. Then it is enough to prove that $P(\exists i \in [p] : W_i \geq \delta) \leq \sqrt{\pi/2}p(p-1)\exp\left(-(n-2)\left(\frac{\delta}{k}\right)^2/2\right)$.

W_i are identically distributed variables, then by a union bound, $P(\exists i \in [p] : W_i \geq \delta) \leq pP(W_1 \geq \delta)$.

By definition, $P(W_1 \geq \delta) = P\left(\max_{\Lambda\subset[p]\backslash 1,|\Lambda|=k} \sum_{j\in\Lambda} \left|\left\langle d_1, d_j\right\rangle\right| \geq \delta\right)$ and since $\sum_{j\in\Lambda} \left|\left\langle d_1, d_j\right\rangle\right| \leq k \max_{j\neq 1} \left|\left\langle d_1, d_j\right\rangle\right|$ always,

$$\begin{aligned} P(W_1 \geq \delta) &\leq P\left(k \max_{j\neq 1} \left|\left\langle d_1, d_j\right\rangle\right| \geq \delta\right) \\ &= P\left(\max_{j\neq 1} \left|\left\langle d_1, d_j\right\rangle\right| \geq \frac{\delta}{k}\right) \end{aligned}$$

Note that $\max_{j\neq 1} \left|\left\langle d_1, d_j\right\rangle\right| \geq \delta/k \iff \exists j \in [p]\backslash i : \left|d_1, d_j\right| \geq \delta/k$. Noting the random variables $\left|\left\langle d_1, d_j\right\rangle\right|$ are identically distributed, and using a union bound on the choice of j, we have $P(W_1 \geq \delta) \leq (p-1)P(|\langle d_1, d_2\rangle| \geq \delta/k)$.

Since $\langle d_1, d_2\rangle$ is invariant to applying to d_1 and d_2 the same orthonogonal transformation, we may assume without loss of generality that $d_2 = e_1$, and with another union bound note that $P(|\langle d_1, d_2\rangle| \geq \delta/k) = P(|\langle e_1, d_1\rangle| \geq \delta/k) \leq 2P(\langle e_1, d_1\rangle \geq \delta/k)$.

The fraction $\frac{\delta}{k}$ is positive, then the set of d_1 on which $\langle e_1, d_1\rangle < \delta/k$ holds includes the negative half sphere, and any point within δ/k of it. Then by the isoperimetric inequality of Lemma 39,

$$2P(\langle e_1, d_1\rangle \geq \delta/k) \leq \sqrt{\pi/2}\exp\left(-(n-2)(\delta/k)^2/2\right).$$

The theorem results by substitution. ■

References

Michal Aharon, Michael Elad, and Alfred M. Bruckstein. K-SVD: An algorithm for designing overcomplete dictionaries for sparse representation. *IEEE Transactions on Signal Processing*, 54 (11):4311–4322, 2006.

Martin Anthony and Peter L. Bartlett. *Neural Network Learning: Theoretical Foundations*. Cambridge University Press, 1999.

Peter L. Bartlett, Tamás Linder, and Gábor Lugosi. The minimax distortion redundancy in empirical quantizer design. *IEEE Transactions on Information Theory*, 44(5):1802–1813, 1998.

Peter L. Bartlett, Olivier Bousquet, and Shahar Mendelson. Local Rademacher complexities. *Ann. Statist.*, 33:1497–1537, 2005.

Gérard Biau, Luc Devroye, and Gábor Lugosi. On the performance of clustering in Hilbert spaces. *IEEE Transactions on Information Theory*, 54(2):781–790, 2008.

Gilles Blanchard, Olivier Bousquet, and Laurent Zwald. Statistical properties of kernel principal component analysis. *Machine Learning*, 66(2):259–294, 2007.

Alfred M. Bruckstein, David L. Donoho, and Michael Elad. From sparse solutions of systems of equations to sparse modeling of signals and images. *SIAM Review*, 51(1):34–81, 2009.

Emmanuel J. Candes and Terence Tao. Near-optimal signal recovery from random projections: Universal encoding strategies? *IEEE Transactions on Information Theory*, 52(12):5406–5425, 2006.

Scott S. Chen, David L. Donoho, and Michael A. Saunders. Atomic decomposition by basis pursuit. *SIAM Review*, 43(1):129–159, 2001.

Sheng Chen, Stephen A. Billings, and Wan Luo. Orthogonal least squares methods and their application to non-linear system identification. *International Journal of Control*, 50(5):1873–1896, 1989.

Felipe Cucker and Stephen Smale. On the mathematical foundations of learning. *Bull. Amer. Math. Soc*, 39(1):1–50, 2002.

Geoff Davis, Stèphane Mallat, and Marco Avellaneda. Adaptive greedy approximations. *Constructive approximation*, 13(1):57–98, 1997.

David L. Donoho and Michael Elad. Optimally sparse representation in general (nonorthogonal) dictionaries via ℓ_1 minimization. *Proceedings of the National Academy of Sciences*, 100(5): 2197–2202, 2003.

Richard J. Duffin and Albert C. Schaeer. A class of nonharmonic Fourier series. *Trans. Amer. Math. Soc*, 72:341–366, 1952.

Ralf Herbrich and Robert Williamson. Algorithmic luckiness. *Journal of Machine Learning Research*, 3:175–212, 2003.

Roger A. Horn and Charles R. Johnson. *Matrix Analysis*. Cambridge University Press, 1990.

Kenneth Kreutz-Delgado, Joseph F. Murray, Bhaskar D. Rao, Kjersti Engan, Te-Won Lee, and Terrance J. Sejnowski. Dictionary learning algorithms for sparse representation. *Neural Computation*, 15(2):349–396, 2003.

Honglak Lee, Alexis Battle, Rajat Raina, and Andrew Y. Ng. Efficient sparse coding algorithms. In *Advances in Neural Information Processing Systems 19*, pages 801–808. MIT Press, Cambridge, MA, 2007.

Paul Lévy. *Problèmes concrets d'analyse fonctionnelle*. Gauthier-Villars, Paris, 1951.

Michael S. Lewicki, Terrence J. Sejnowski, and Howard Hughes. Learning overcomplete representations. *Neural Computation*, 12:337–365, 1998.

Julien Mairal, Francis Bach, Jean Ponce, and Guillermo Sapiro. Online learning for matrix factorization and sparse coding. *Journal of Machine Learning Research*, 11:19–60, 2010.

Andreas Maurer and Massimiliano Pontil. K-dimensional coding schemes in hilbert spaces. *IEEE Transactions on Information Theory*, 56:5839–5846, November 2010.

Shahar Mendelson. A few notes on statistical learning theory. *Advanced Lectures on Machine Learning*, pages 1–40, 2003.

Bruno A. Olshausen and David J. Field. Sparse coding with an overcomplete basis set: a strategy employed by V1. *Vision Research*, 37:3311–3325, 1997.

Gabriel Peyré. Sparse modeling of textures. *Journal of Mathematical Imaging and Vision*, 34(1): 17–31, 2009.

Matan Protter and Michael Elad. Sparse and redundant representations and motion-estimation-free algorithm for video denoising. *Wavelets XII. Proceedings of the SPIE*, 6701:43, 2007.

John Shawe-Taylor and Nello Cristianini. *Kernel Methods for Pattern Analysis*. Cambridge University Press, 2004.

John Shawe-Taylor, Christopher K. I. Williams, Nello Cristianini, and Jaz Kandola. On the eigenspectrum of the Gram matrix and the generalization error of kernel-PCA. *IEEE Transactions on Information Theory*, 51(7):2510–2522, 2005.

Joel A. Tropp. Greed is good: Algorithmic results for sparse approximation. *IEEE Transactions on Information Theory*, 50:2231–2242, 2004.

John Wright, Allen Y. Yang, Arvind Ganesh, S. Shankar Sastry, and Yi Ma. Robust face recognition via sparse representation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 210–227, 2008.

Journal of Machine Learning Research 12 (2011) 3283-3310 Submitted 12/10; Revised 9/11; Published 11/11

An Asymptotic Behaviour of the Marginal Likelihood for General Markov Models

Piotr Zwiernik PIOTR.ZWIERNIK@GMAIL.COM
Institute for Pure and Applied Mathematics
460 Portola Plaza
Box 957121
Los Angeles, CA 90095-7121

Editor: Kenji Fukumizu

Abstract

The standard Bayesian Information Criterion (BIC) is derived under regularity conditions which are not always satisfied in the case of graphical models with hidden variables. In this paper we derive the BIC for the binary graphical tree models where all the inner nodes of a tree represent binary hidden variables. This provides an extension of a similar formula given by Rusakov and Geiger for naive Bayes models. The main tool used in this paper is the connection between the growth behavior of marginal likelihood integrals and the real log-canonical threshold.

Keywords: BIC, marginal likelihood, singular models, tree models, Bayesian networks, real log-canonical threshold

1. Introduction

A key step in the Bayesian learning of graphical models is to compute the *marginal likelihood* of the data, which is the *likelihood function* averaged over the parameters with respect to the prior distribution. Given a fully observed system, the theory of graphical models provides a simple way to obtain the marginal likelihood. This was explained for example by Cooper and Herskovits (1992) and Heckerman et al. (1995). However, when some of the variables in the system are *hidden* (never observed), the exact determination of the marginal likelihood is typically intractable (for example Chickering and Heckerman, 1997). This motivates the search for efficient techniques to approximate the marginal likelihood. In this paper we focus on the large sample behavior of the marginal likelihood called the *Bayes Information Criterion* (BIC).

To present basic results on the BIC we need to introduce some notation. Let X be a random variable with values in $[m] := \{1,\dots,m\}$. Its distribution $q = (q_1,\dots,q_m)$ can be identified with a point in the *probability simplex* $\Delta_{m-1} = \{x \in \mathbb{R}^m : \sum_i x_i = 1, x_i \geq 0\} \subseteq \mathbb{R}^m$. Consider a map $p : \Theta \to \Delta_{m-1}$ and let $\mathcal{M} = p(\Theta)$ be a parametric discrete model for X with the parameter space Θ and parametrization p. Let $X^{(N)} = X^1,\dots,X^N$ be a random sample from the distribution $q \in \Delta_{m-1}$. By Z_N we denote the marginal likelihood and by $L(\theta;X^{(N)},\mathcal{M}) = \mathbb{P}(X^{(N)}|\mathcal{M},\theta)$ the likelihood function. Thus

$$Z_N = \mathbb{P}(X^{(N)}|\mathcal{M}) = \int_\Theta L(\theta;X^{(N)},\mathcal{M})\varphi(\theta)\mathrm{d}\theta,$$

where θ denotes the model parameters and $\varphi(\theta)$ is a prior distribution on Θ. The *stochastic complexity* is defined by

$$F_N = -\log Z_N$$

and the entropy function by

$$S = -\sum_{i=1}^{m} q_i \log q_i.$$

In statistical theory to obtain the BIC we usually require that the asymptotic limit of the likelihood function, as $N \to \infty$, is maximized over a unique point in the interior of the parameter space where the Jacobian matrix of the parametrization is full rank. For the class of problems for which this assumption is satisfied Schwarz (1978) and Haughton (1988) showed that, as $N \to \infty$,

$$\mathbb{E}F_N = NS + \frac{d}{2}\log N + O(1),$$

where $d = \dim \Theta$ (Watanabe, 2009, Corollary 1.15 and Section 6). The same formula works if the limit of the likelihood is maximized over a finite number of points in the interior of Θ. Geometrically, for large sample sizes function Z_N concentrates around the maxima. This enables us to apply the Laplace approximation locally in the neighborhood of each maximum.

It can be proved (see Proposition 5) that the above formula can be generalized for the case when the set, over which the limit of the likelihood is maximized, forms a sufficiently regular compact subset of the parameter space. Denote this subset by $\widehat{\Theta}$. Then, as $N \to \infty$,

$$\mathbb{E}F_N = NS + \frac{d-d'}{2}\log N + O(1), \tag{1}$$

where $d' = \dim \widehat{\Theta}$. Note that in our case $\widehat{\Theta}$ is a set of zeros of a real analytic function. Therefore, it will be always a semi-analytic set, that is given by $\{g_1(\theta) \geq 0, \ldots, g_r(\theta) \geq 0\}$, where g_i are all analytic functions. It follows that the dimension is well defined (Bierstone and Milman, 1988, Remark 2.12).

In the case of models with hidden variables the locus of the points maximizing the limit of the likelihood may not be sufficiently regular. In this case the likelihood will have a different asymptotic behavior around different points and relatively more mass of the marginal likelihood integral will be related to neighborhoods of singular points (see Figure 1). For these points we cannot use the Laplace approximation. Nevertheless, the computation of the BIC is still possible using results of Watanabe (2009) and some earlier works of Arnold, Varchenko and collaborators (Arnold et al., 1988). This formula will differ from the standard BIC. First, the coefficient of $\log N$ can be different from $\frac{d-d'}{2}$ in (1). Second, we sometimes encounter an additional $\log\log N$ term affecting the asymptotics (see Theorem 4).

Let again q be the true data generating distribution and $\mathcal{M} = p(\Theta)$ a discrete parametric statistical model with the parameter space Θ. Let $\varphi : \Theta \to \mathbb{R}$ be a prior distribution. Throughout the paper we always assume:

(A1) The prior distribution φ is strictly positive, bounded and smooth on Θ.

(A2) There exists $\theta \in \Theta$ such that $p(\theta) = q$ and q lies in the interior of the probability simplex.

(A3) The set $\Theta \subseteq \mathbb{R}^d$ is a compact and *semianalytic set* of dimension d.

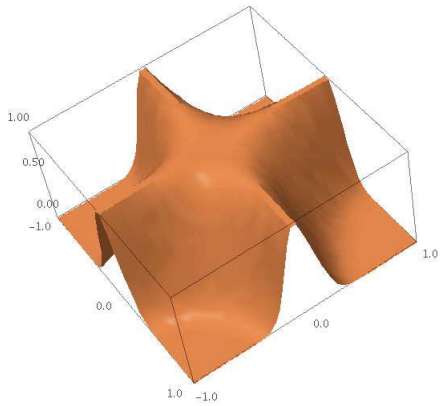


Figure 1: The case when the likelihood is maximized over a singular subset of $\Theta = [-1,1]^2$ given by $\theta_1\theta_2 = 0$.

In this paper we consider an important class of parametric models with large number of hidden variables, called *general Markov models*, assuming for simplicity that all the random variables in the system are binary. This model class is extensively used in phylogenetics (Semple and Steel, 2003, Chapter 8) and in the analysis of causal systems (see Pearl and Tarsi, 1986). We begin with a quick informal introduction to general Markov models which is then formalized in Section 3.1. Let $T = (V,E)$ be an undirected tree with the vertex set V and the edge set E. Let T^r denote T *rooted in* r, that is a tree with one distinguished vertex r and all the edges directed away from r. Consider the *Markov process* $Y = (Y_v)_{v\in V}$ on T^r, which by definition is the Bayesian network on T^r. Then, the *general Markov model* is a family of marginal distributions over the subvector of Y corresponding to the leaves of T^r. It is well known that this model class does not depend on the rooting. Therefore, we denote this model class, omitting the rooting, by $\mathcal{M}_T$.

For a tree T with n leaves we denote the subvector of Y corresponding to the leaves of T by $X = (X_1,\ldots,X_n)$ with some arbitrary numbering of leaves. The subvector of Y corresponding to the inner nodes is denoted by H. By construction the general Markov model is a statistical model for X. Let $q \in \mathcal{M}_T$ be the true distribution and $\widehat{\Sigma} = [\hat{\mu}_{ij}]$ be the covariance matrix of X. A surprising fact proved in this paper is that the zeros in $\widehat{\Sigma}$, or equivalently, marginal independencies between components of X, completely determine the asymptotics for the marginal likelihood.

We say that two nodes u,v of T are *separated* by another node w, if w lies on the unique path between u and v. Let l_2 denote the number of *inner* nodes v of T such that for each triple i,j,k of leaves separated in T by v we have $\hat{\mu}_{ij}\hat{\mu}_{ik}\hat{\mu}_{jk} = 0$ but there exist leaves i,j separated by v such that $\hat{\mu}_{ij} \neq 0$. In terms of the conditional independence defining the general Markov model, an inner node v contributes to l_2 if for every three leaves i,j,k such that $X_i \perp\!\!\!\perp X_j \perp\!\!\!\perp X_k | H_v$ at least two are marginally independent but there exist two leaves i,j such that $X_i \perp\!\!\!\perp X_j | H_v$ but not $X_i \perp\!\!\!\perp X_j$. In addition, we say that an inner node v is *degenerate (or q-degenerate)* if for any two leaves i,j separated by v we have $\hat{\mu}_{ij} = 0$. In other words v is degenerate if for every i,j such that $X_i \perp\!\!\!\perp X_j | H_v$ also $X_i \perp\!\!\!\perp X_j$. All other nodes are called *nondegenerate*.

We denote by n_e the number of edges of T and by n_v the number of its nodes. The following result is a special instance of (Watanabe, 2009, the Main Formula II, p. 34):

Theorem 1 *Let $T^r = (V,E)$ be a tree rooted in r and $X^{(N)}$ be a random sample from q. With assumptions (A1), (A2) and (A3), if there are no q-degenerate nodes then, as $N \to \infty$,*

$$\mathbb{E}F_N = NS + \frac{n_v + n_e - 2l_2}{2}\log N + O(1).$$

Note, in particular, that the above formula is independent of the rooting.

Example 1 *Let $n = 4$ and assume that data are generated from the Bayesian network given by the quartet tree in Figure 2. If q is such that $\widehat{\Sigma}$ has no zeros then $l_2 = 0$ and the coefficient of $\log N$ is*

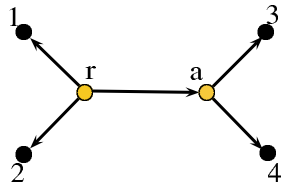


Figure 2: A quartet tree rooted in r.

$\frac{11}{2}$. *This corresponds to the classical BIC since the dimension of the parameter space is* 11. *If the true distribution $q \in \mathcal{M}_T$ satisfies in addition the marginal independence condition $X_1 \perp\!\!\!\perp (X_2, X_3, X_4)$ then $\hat{\mu}_{1i} = 0$ for $i = 2,3,4$ and r contributes to l_2. We depict this situation on the left hand side in Figure 3. Here the dashed edge means that for every pair i, j of leaves separated by this edge $\hat{\mu}_{ij} = 0$ and an inner node contributes to l_2 if its valency, in the forest with the dashed edges removed, is* 2. *In this case $l_2 = 1$ and the coefficient of $\log N$ is $\frac{9}{2}$. If, in addition, q satisfies $X_1 \perp\!\!\!\perp X_3 \perp\!\!\!\perp (X_2, X_4)$ then $l_2 = 2$ and hence the coefficient is $\frac{7}{2}$. The corresponding graph is depicted in the middle of Figure 3.*

Example 2 (Naive Bayes model) *Consider a star tree with one inner node and n leaves. If there are no degenerate nodes this corresponds to q being either a regular point or a type* 1 *singularity as defined by Rusakov and Geiger (2005). If $l_2 = 0$ then q is a regular point and the coefficient of $\log N$ is equal to $\frac{2n+1}{2}$. If $l_2 = 1$ then q is a type* 1 *singularity and the coefficient is equal to $\frac{2n-1}{2}$. This corresponds exactly to (Rusakov and Geiger, 2005, Theorem 4). If the inner node is degenerate this corresponds to the type 2 singularity which does not satisfy assumptions of Theorem 1.*

If q is such that there are degenerate nodes the computation of the BIC is much harder because the likelihood in this case maximizes over a singular subset of the parameter space. The case of star

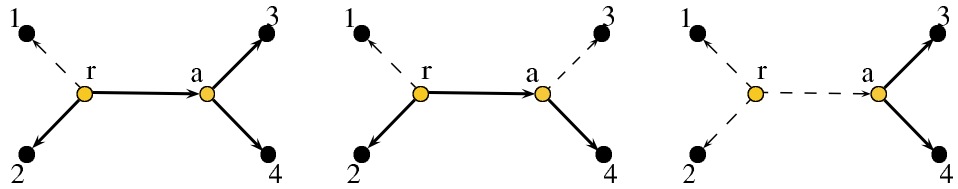


Figure 3: Three graphs representing submodels of the quartet tree model with some additional marginal independencies. In the third case the root is degenerate.

trees was investigated by Rusakov and Geiger (2005). In this paper we obtain a closed form formula for the BIC in the case of *trivalent trees*, whose all inner nodes have valency three. This is provided in Theorem 2 which together with Theorem 1 are the main results of this paper. The importance of trivalent trees follows mainly from the fact that any other tree model is a submodel of a model for a trivalent tree. They also form a natural class for models of evolution in biology.

If T is trivalent then for every inner node $v \in V$ there exist $A,B,C \subseteq [n]$ such that $A \cup B \cup C = [n]$ and A,B,C are separated by v. By the defining conditional independence conditions we have that $X_A \perp\!\!\!\perp X_B \perp\!\!\!\perp X_C | Y_v$, where $X_A = (X_i)_{i \in A}$. In this case we call v degenerate if q is such that $X_A \perp\!\!\!\perp X_B \perp\!\!\!\perp X_C$. Let l_0 denote the number of degenerate nodes.

Theorem 2 *Let $T^r = (V,E)$ be a rooted trivalent tree with $n \geq 3$ leaves and root r. With assumptions (A1), (A2) and (A3) if r is degenerate but all its neighbors are not, then, as $N \to \infty$,*

$$\mathbb{E}F_N = NS + \left(\frac{n_v + n_e - 2l_2}{2} - \frac{5l_0 + 1}{4} \right) \log N + O(1).$$

In all other cases, as $N \to \infty$,

$$\mathbb{E}F_N = NS + \left(\frac{n_v + n_e - 2l_2}{2} - \frac{5l_0}{4} \right) \log N - c \log\log N + O(1),$$

where c is a nonnegative integer. Moreover $c = 0$ always if either both r is nondegenerate or if r and all its neighbors are degenerate.

The coefficients of $\log N$ above are given in this special form to show the correction term with respect to the coefficient in the smooth case in Theorem 1.

Example 3 *Consider again the quartet model from Example 1. If $\widehat{\Sigma}$ has no zeros then $l_2 = 0$, $l_0 = 0$ and we get the same formula as previously with coefficient $\frac{11}{2}$. Now assume that q is such that in addition the marginal independence $X_1 \perp\!\!\!\perp X_2 \perp\!\!\!\perp (X_3, X_4)$ holds. The situation is depicted on the right hand side in Figure 3. The edge (r,a) is dashed since for any two leaves separated by this edge the corresponding covariance is zero. In this case $l_2 = 1$, $l_0 = 1$, the root is degenerate but all its neighbors are not and hence, by Theorem 2, the coefficient of $\log N$ is 3 and $c = 0$. Consider finally the case when all off-diagonal elements of $\widehat{\Sigma}$ are zero. In this case $l_2 = 0$ and $l_0 = 2$ and hence the coefficient of $\log N$ is also 3. However, later in Example 5 and Remark 29 we will see that c may be strictly greater than zero in this case.*

Following Rusakov and Geiger (2005), the main method of the proof is to change the coordinates of the model so that the induced parameterization becomes simple. This gives us a much better insight into the model structure which is described by Zwiernik and Smith (2011b) and Zwiernik and Smith (2011a). Since the BIC is invariant with respect to these changes, the reparameterized problem still gives the solution to the original question. Our main analytical tool is the real log-canonical threshold (see for example Watanabe, 2009). This is an important geometric invariant which in certain cases can be computed in a relatively simple way using discrete geometry. The relevance of this invariant to the BIC is given by Theorem 4. We remark that the techniques developed in this paper can be applied to obtain the BIC also in the case of non-trivalent trees.

The paper is organized as follows. In Section 2, following Watanabe (2009), we provide the theory of asymptotic expansion of marginal likelihood integrals. This theory enables us to analyze

the asymptotic behavior of the marginal likelihood without the standard regularity assumptions. In Section 3 we define Bayesian networks on rooted trees. We also obtain a basic result on the BIC in the case when the observed likelihood is maximised over a sufficiently smooth subset of the parameter space. This gives a simple proof of Theorem 1. The proof of Theorem 2 is more technical and hence divided into three main steps split between Sections 4, 5 and 6. Finally, in Section 7, we combine all these results.

2. Asymptotics of Marginal Likelihood Integrals

In this section we introduce the real log-canonical threshold and link it to the asymptotic behavior of marginal likelihood integrals. We present how this enables us to obtain the BIC in the case of a general class of statistical models, which is mostly based on previous results of Sumio Watanabe.

2.1 The Real Log-canonical Threshold

Given $\theta_0 \in \mathbb{R}^d$, let $\mathcal{A}_{\theta_0}(\mathbb{R}^d)$ be the ring of real-valued functions $f : \mathbb{R}^d \to \mathbb{R}$ that are analytic at θ_0. Given a subset $\Theta \subset \mathbb{R}^d$ satisfying (A3), let $\mathcal{A}_\Theta(\mathbb{R}^d)$ be the ring of real functions analytic at each point $\theta_0 \in \Theta$. If $f \in \mathcal{A}_\Theta(\mathbb{R}^d)$, then for every $\theta_0 \in \Theta$, f can be locally represented as power series centered at θ_0. Denote by $\mathcal{A}_\Theta^{\geq}(\mathbb{R}^d)$ the subset of $\mathcal{A}_\Theta(\mathbb{R}^d)$ consisting of all non-negative functions. Usually the ambient space is clear from the context and in this case we omit it in our notation writing $\mathcal{A}_{\theta_0}$, $\mathcal{A}_\Theta$ and $\mathcal{A}_\Theta^{\geq}$.

Definition 3 (The real log-canonical threshold) *Given a compact semianalytic set $\Theta \subseteq \mathbb{R}^d$ such that $\dim \Theta = d$, a real analytic function $f \in \mathcal{A}_\Theta^{\geq}(\mathbb{R}^d)$ and a smooth positive function $\varphi : \mathbb{R}^d \to \mathbb{R}$, consider the zeta function defined as*

$$\zeta(z) = \int_\Theta f(\theta)^{-z}\varphi(\theta)\mathrm{d}\theta. \tag{2}$$

By Theorem 2.4 of Watanabe (2009) this function is extended to a meromorphic function on the entire complex line and its poles are real and positive. The real log-canonical threshold of f denoted by $\mathrm{rlct}_\Theta(f;\varphi)$ is the smallest pole of $\zeta(z)$. By $\mathrm{mult}_\Theta(f;\varphi)$ we denote the multiplicity of this pole. By convention if $\zeta(z)$ has no poles then $\mathrm{rlct}_\Theta(f;\varphi) = \infty$ and $\mathrm{mult}_\Theta(f;\varphi) = d$. If $\varphi(\theta) \equiv 1$ then we omit φ in the notation writing $\mathrm{rclt}_\Theta(f)$ and $\mathrm{mult}_\Theta(f)$. Define $\mathrm{RLCT}_\Theta(f;\varphi)$ to be the pair $(\mathrm{rlct}_\Theta(f;\varphi), \mathrm{mult}_\Theta(f;\varphi))$, and we order these pairs so that $(r_1,m_1) > (r_2,m_2)$ if $r_1 > r_2$, or $r_1 = r_2$ and $m_1 < m_2$.

Let $\mathcal{M} = p(\Theta) \subseteq \Delta_{m-1}$ be a parametric discrete model and $q \in \Delta_{m-1}$ be a probability distribution. With $\mathcal{M}$ and q fixed the Kullback-Leibler distance $K : \Theta \to \mathbb{R}$ is defined by

$$K(\theta) = \sum_{i=1}^m q_i \log \frac{q_i}{p_i(\theta)}. \tag{3}$$

It is well known that $K(\theta) \geq 0$ on Θ and $K(\theta) = 0$ if and only if $p(\theta) = q$. If q is the true data generating distribution then assumption (A2) means that $\widehat{\Theta} = \{\theta : K(\theta) = 0\}$ is non-empty.

The following theorem gives the motivation to study the real log-canonical threshold in the statistical context.

Theorem 4 (Watanabe) *Let $\mathcal{M}$ be a parametric discrete statistical model, q the true data generating distribution and K the corresponding Kullback-Leibler distance. With assumptions (A1), (A2) and (A3), as $N \to \infty$,*

$$\mathbb{E}F_N = NS + \mathrm{rlct}_\Theta(K;\varphi)\log N + (\mathrm{mult}_\Theta(K;\varphi)-1)\log\log N + O(1).$$

To compute the real log-canonical threshold we split the integral in (2) into a sum of finitely many integrals over small neighbourhoods Θ_0 of some points $\theta_0 \in \Theta$ for which we have efficient tools of computation. We can always do this using the partition of unity since Θ is compact. For each of the local integrals we use Hironaka's theorem to reduce it to a locally monomial case. The details are presented by Watanabe (2009).

Let $\theta_0 \in \Theta$ and let W_0 be any sufficiently small open ball around θ_0 in $\mathbb{R}^d$. Then, by Theorem 2.4 of Watanabe (2009), $\mathrm{RLCT}_{W_0}(f;\varphi)$ does not depend on the choice of W_0 and hence it is denoted by $\mathrm{RLCT}_{\theta_0}(f;\varphi)$. If $f(\theta_0) \neq 0$ then $\mathrm{RLCT}_{W_0}(f;\varphi) = (\infty, d)$ and hence we can constrain only to points θ_0 such that $f(\theta_0) = 0$. In our context this means that we consider only points in the q-fiber $\widehat{\Theta}$.

The local computations give the answer to the global question because, by (Lin, 2011, Proposition 2.5), the set of pairs $\mathrm{RLCT}_{\Theta_0}(f;\varphi)$ for $\theta_0 \in \Theta$ has a minimum and

$$\mathrm{RLCT}_\Theta(f;\varphi) = \min_{\theta_0 \in \Theta} \mathrm{RLCT}_{\Theta_0}(f;\varphi), \tag{4}$$

where $\Theta_0 = W_0 \cap \Theta$ is the neighbourhood of θ_0 in Θ. For each $\theta_0 \in \Theta$ to compute $\mathrm{RLCT}_{\Theta_0}(f;\varphi)$ we consider two cases. If θ_0 lies in the interior of Θ then we can assume $\Theta_0 = W_0$ and hence $\mathrm{RLCT}_{\Theta_0}(f;\varphi) = \mathrm{RLCT}_{\theta_0}(f;\varphi)$. If $\theta_0 \in \mathrm{bd}(\Theta)$, where $\mathrm{bd}(\Theta)$ denotes the set of boundary points of Θ, the computations may change significantly because the real log-canonical threshold depends on the boundary conditions (cf. Example 2.7 of Lin, 2011). Nevertheless, it can be showed that at least if there exists an open subset $U \subseteq \mathbb{R}^d$ such that $U \supset \Theta_0$ and $f \in \mathcal{A}^{\geq}_U(\mathbb{R}^d)$ then

$$\mathrm{RLCT}_{\Theta_0}(f) \geq \mathrm{RLCT}_{\theta_0}(f). \tag{5}$$

Because in this case

$$\int_{W_0} (f(\theta))^{-z} \mathrm{d}\theta = \int_{\Theta_0} (f(\theta))^{-z} \mathrm{d}\theta + \int_{W_0 \setminus \Theta_0} (f(\theta))^{-z} \mathrm{d}\theta$$

which implies that $\mathrm{RLCT}_{\theta_0}(f) = \min\{\mathrm{RLCT}_{\Theta_0}(f), \mathrm{RLCT}_{W_0 \setminus \Theta_0}(f)\}$.

Finally, whenever $\widehat{\Theta} \neq \emptyset$ we have

$$\mathrm{RLCT}_\Theta(K) = \min_{\theta_0 \in \widehat{\Theta}} \mathrm{RLCT}_{\Theta_0}(K). \tag{6}$$

The following result enables us to obtain the BIC in the smooth case.

Proposition 5 *Let $\mathcal{M}$ be a parametric statistical model with parametrization p, and q be the true data generating distribution. Let $K \in \mathcal{A}^{\geq}_\Theta(\mathbb{R}^d)$ be the Kullback-Leibler distance defined in (3). Given (A1), (A2) and (A3) assume that there exists a smooth manifold $M \subseteq \mathbb{R}^d$ satisfying $\widehat{\Theta} = M \cap \Theta$. Then, as $N \to \infty$,*

$$\mathbb{E}F_N = NS + \frac{d-d'}{2}\log N + O(1),$$

where $d' = \dim \widehat{\Theta}$.

Proof By assumption (A1) there exist two constants $c, C > 0$ such that $c < \varphi(\theta) < C$ on Θ. Therefore

$$c\int_\Theta (K(\theta))^{-z}\mathrm{d}\theta < \zeta(z) < C\int_\Theta (K(\theta))^{-z}\mathrm{d}\theta$$

and it follows that $\mathrm{RLCT}_\Theta(K;\varphi) = \mathrm{RLCT}_\Theta(K)$. By Theorem 4 it suffices to prove the following lemma which generalises Proposition 3.3 of Saito (2007).

Lemma 6 *Let $\Theta \subset \mathbb{R}^d$ be a compact semianalytic set and $f \in \mathcal{A}^{\geq}_\Theta(\mathbb{R}^d)$. If there exists a smooth manifold $M \subseteq \mathbb{R}^d$ such that $\widehat{\Theta} = M \cap \Theta$ and $\theta_0 \in \widehat{\Theta}$ then $\mathrm{RLCT}_{\theta_0}(f) = \mathrm{RLCT}_\Theta(f) = (\frac{d-d'}{2}, 1)$ where $d' = \dim \widehat{\Theta}$.*

To prove this recall that the real log-canonical threshold $\mathrm{RLCT}_{\theta_0}(f)$ does not depend on the choice of a sufficiently small neighborhood W_0 of θ_0. Since $\widehat{\Theta} = M \cap \Theta$ and M is a smooth manifold it follows that for each point θ_0 of $\widehat{\Theta}$ there exists an open neighborhood W_0 of θ_0 in $\mathbb{R}^d$ with local coordinates $w_1, \ldots, w_d$ centered at θ_0 such that the local equation of $\widehat{\Theta}$ is $w_1^2 + \cdots + w_c^2 = 0$, where $c = d - d'$. A single blow-up π at the origin satisfies all the conditions of Hironaka's Theorem since in the new coordinates over one of the charts $f(\pi(u)) = u_1^2 a(u)$ where $a(u)$ is nowhere vanishing and $\pi'(u) = u_1^{c-1}$. For other charts the situation is the same and hence $\mathrm{RLCT}_{\theta_0}(f) = (c/2, 1)$. Since by (4) $\mathrm{RLCT}_\Theta(f) = \min_{\theta_0 \in \Theta} \mathrm{RLCT}_{W_0 \cap \Theta}(f)$ it suffices to show that if θ_0 is a boundary point of Θ then $\mathrm{RLCT}_{W_0 \cap \Theta}(f) \geq (c/2, 1)$. But this follows from (5) and the fact that $\mathrm{RLCT}_{\theta_0}(f) = (c/2, 1)$ as θ_0 is a smooth point of M. The lemma is hence proved. ■

3. General Markov Models

In this section we formally define the general Markov model $\mathcal{M}_T$ and give in Theorem 1 the asymptotic expansion of the marginal likelihood when q and $\mathcal{M}_T$ satisfy conditions of Proposition 5.

3.1 Definition of the Model Class

All random variables considered in this paper are assumed to be binary with the value either 0 or 1. Let $T^r = (V, E)$ be a rooted tree. For any directed edge $e = (k, l) \in E$ we say that k and l are *adjacent* and k is a *parent* of l and we denote it by $k = \mathrm{pa}(l)$. For every $\beta \in \{0,1\}^V$ let $p_\beta = \mathbb{P}(\bigcap_{v \in V} \{Y_v = \beta_v\})$. The *Markov process* on T^r is a sequence $Y = (Y_v)_{v \in V}$ of binary random variables such that for each $\beta = (\beta_v)_{v \in V} \in \{0,1\}^V$

$$p_\beta(\theta) = \theta^{(r)}_{\beta_r} \prod_{v \in V \setminus r} \theta^{(v)}_{\beta_v | \beta_{\mathrm{pa}(v)}}, \tag{7}$$

where $\theta^{(v)}_{\beta_v|\beta_{\mathrm{pa}(v)}} = \mathbb{P}(Y_v = \beta_v | Y_{\mathrm{pa}(v)} = \beta_{\mathrm{pa}(v)})$ and $\theta^{(r)}_{\beta_r} = \mathbb{P}(Y_r = \beta_r)$. In a more standard statistical language these models are just fully observed Bayesian networks on rooted trees. Recall that $n_e = |E|$ and $n_v = |V|$. Since $\theta^{(v)}_{0|i} + \theta^{(v)}_{1|i} = 1$ for all $v \in V$ and $i = 0, 1$ then the Markov process on T^r defined by (7) has exactly $2n_e + 1$ free parameters in the vector θ: one for the root distribution $\theta^{(r)}_1$ and two for each edge $(u, v) \in E$ given by $\theta^{(v)}_{1|0}$ and $\theta^{(v)}_{1|1}$. The parameter space is $\Theta_T = [0,1]^{2n_e+1}$.

The general Markov model on T^r is induced from the Markov process on T^r by assuming that all the inner nodes represent hidden random variables. Hence we consider induced marginal

probability distributions over the leaves of T^r. The set of leaves is denoted by L. We assume that T^r has n leaves and hence we can associate L with the set $[n]$ given some arbitrary numbering of the leaves. Let $Y = (X,H)$ where $X = (X_1,\ldots,X_n)$ denotes the variables represented by the leaves of T^r and H denotes the vector of variables represented by inner nodes, that is $X = (Y_v)_{v\in L}$ and $H = (Y_v)_{v\in V\setminus L}$. We define the general Markov model $\mathcal{M}_T$ to be the model in the probability simplex Δ_{2^n-1} obtained by summing out in (7) over all possible values of the inner nodes. By definition $\mathcal{M}_T$ is the image of the map $p : \Theta_T \to \Delta_{2^n-1}$ given by

$$p_\alpha(\theta) = \sum_{\mathcal{H}} \theta^{(r)}_{\beta_r} \prod_{v\in V\setminus r} \theta^{(v)}_{\beta_v|\beta_{\mathrm{pa}(v)}} \qquad \text{for any } \alpha \in \{0,1\}^L,$$

where $\mathcal{H}$ is the set of all vectors $\beta = (\beta_v)_{v\in V}$ such that $(\beta_v)_{v\in L} = \alpha$. Because the model class does not depend on the rooting we usually omit the root r in the notation. For a more detailed treatment see (Semple and Steel, 2003, Chapter 8).

3.2 The BIC in the Smooth Case

For $q \in \mathcal{M}_T$ let $\widehat{\Sigma} = [\hat{\mu}_{ij}] \in \mathbb{R}^{n\times n}$ be the corresponding covariance matrix of the random vector represented by the leaves of T. We define the q-fiber as

$$\widehat{\Theta}_T \quad = \quad \{\theta \in \Theta_T : p(\theta) = q\} \quad = \quad \{\theta \in \Theta_T : K(\theta) = 0\}.$$

The geometry of $\widehat{\Theta}_T$ is directly related to the real log-canonical threshold of the Kullback-Leibler distance. We now show that this geometry is determined by zeros in $\widehat{\Sigma}$. For this we need to introduce some new concepts. We say that that an edge $e \in E$ is *isolated relative to* q if $\hat{\mu}_{ij} = 0$ for all $i, j \in [n]$ such that $e \in E(ij)$, where $E(ij)$ denotes the set of edges in the path joining i and j. By $\widehat{E} \subseteq E$ we denote the set of all edges of T which are isolated relative to q. By $\widehat{T} = (V, E\setminus\widehat{E})$ we denote the forest obtained from T by removing edges in $\widehat{E}$.

We now define relations on $\widehat{E}$ and $E\setminus\widehat{E}$. For two edges e,e' with either $\{e,e'\} \subset \widehat{E}$ or $\{e,e'\} \subset E\setminus\widehat{E}$ write $e \sim e'$ if either $e = e'$ or e and e' are adjacent and all the edges that are incident with both e and e' are isolated relative to q. Let us now take the transitive closure of $\sim$ restricted to pairs of edges in $\widehat{E}$ to form an equivalence relation on $\widehat{E}$. Similarly, take the transitive closure of $\sim$ restricted to the pairs of edges in $E\setminus\widehat{E}$ to form an equivalence relation in $E\setminus\widehat{E}$. We will let $[\widehat{E}]$ and $[E\setminus\widehat{E}]$ denote the set of equivalence classes of $\widehat{E}$ and $E\setminus\widehat{E}$ respectively.

By construction all the inner nodes of T have either degree zero in $\widehat{T}$ or the degree is strictly greater than one. We say that a node $v \in V$ is *non-degenerate with respect to* q if either v is a leaf of T or $\deg v \geq 2$ in $\widehat{T}$. Otherwise we say that the node is *degenerate with respect to* q. Note that this coincides with the definition of a degenerate node given in the introduction. Moreover, the isolated edges in Examples 1 and 3 correspond precisely to the dashed edges in Figure 3. The set of all nodes which are degenerate with respect to q is denoted by $\widehat{V}$.

Proposition 7 (Zwiernik and Smith, 2011b) *Let T be a tree with n leaves. Let $q \in \mathcal{M}_T$ and let $\widehat{T}$ be defined as above. If each of the inner nodes of T has degree at least two in $\widehat{T}$ then $\widehat{\Theta}_T$ is a manifold with corners and $\dim\widehat{\Theta}_T = 2l_2$, where l_2 is the number of nodes which have degree two in $\widehat{T}$.*

In this way we can compute the asymptotic behavior of the marginal likelihood in the case when assumptions of Proposition 7 are satisfied.

Proposition 8 *Let T be a tree and $\mathcal{M}_T$ be the corresponding general Markov model. Let $q \in \mathcal{M}_T$ be the real distribution generating the data such that each inner node of T has degree at least two in $\widehat{T}$. Then*

$$\mathrm{RLCT}_{\Theta_T}(K) = \left(\frac{n_v + n_e - 2l_2}{2}, 1\right).$$

Proof Since every inner node of T has degree at least two in $\widehat{T}$ then by Proposition 7 there exists a smooth manifold $M \subseteq \mathbb{R}^{n_v+n_e}$ such that $\widehat{\Theta}_T = M \cap \Theta_T$ and $\dim \widehat{\Theta} = 2l_2$. The result follows from Proposition 5 and the fact that $\dim \Theta_T = 2n_e + 1 = n_v + n_e$. ■

By Theorem 4, Proposition 8 implies Theorem 1 since l_2 in its statement is exactly the number of inner nodes v such that the degree of v in $\widehat{T}$ is two.

Remark 9 *Theorem 1 is still true if (A1) is replaced by the assumption that the prior distribution is bounded on Θ_T and there exists an open subset of Θ_T with a non-empty intersection with $\widehat{\Theta}_T$ where the prior is strictly positive. In particular we can use conjugate Beta priors $\theta_{1|i}^{(v)} \sim \mathrm{Beta}(\alpha_i^{(v)}, \beta_i^{(v)})$ as long as $\alpha_i^{(v)}, \beta_i^{(v)} \geq 1$.*

4. The Ideal-theoretic Approach

In this section we define the real log-canonical threshold of an ideal. Theorem 11 translates the problem of finding the real log-canonical threshold of the Kullback-Leibler distance into algebra. We then analyse general Markov models from this perspective. In Theorem 14 we apply a useful change of coordinates which enables us to work out the real log-canonical threshold in the singular case.

4.1 The Real Log-canonical Threshold of an Ideal

Let $f_1, \ldots, f_r \in \mathcal{A}_\Theta$ then the *ideal generated* by $f_1, \ldots, f_r$ is a subset of $\mathcal{A}_\Theta$ denoted by

$$\langle f_1, \ldots, f_r \rangle = \{f \in \mathcal{A}_\Theta : \ f(\theta) = \sum_{i=1}^{r} h_i(\theta) f_i(\theta), h_i \in \mathcal{A}_\Theta\}.$$

Following Lin (2011) we generalize the notion of the real log-canonical thresholds to the ideal $I = \langle f_1, \ldots, f_r \rangle$. This mirrors the analytic definition of the log-canonical threshold of an ideal (see for example Lazarsfeld, 2004, Section 9.3.D). By definition

$$\mathrm{RLCT}_\Theta(I; \varphi) = \mathrm{RLCT}_\Theta(\langle f_1, \ldots, f_r \rangle; \varphi) := \mathrm{RLCT}_\Theta(f; \varphi),$$

where $f(\theta) = f_1^2(\theta) + \cdots + f_r^2(\theta)$. By (Lin, 2011, Proposition 4.5) the real log-canonical threshold does not depend on the choice of generators of I.

The following important proposition enables us to use the full power of the ideal-theoretic approach.

Proposition 10 *Let $f, g \in \mathcal{A}_\Theta(\mathbb{R}^d)$ and let I be an ideal in $\mathcal{A}_\Theta(\mathbb{R}^d)$. Then*

i *Let* $\rho : \Omega \to \Theta$ *be a proper real analytic isomorphism and* $\rho^* I = \{f \circ \rho : f \in I\}$ *be the pullback of* I *to* $\mathcal{A}_\Omega$. *Then,*

$$\mathrm{RLCT}_\Theta(I;\varphi) = \mathrm{RLCT}_\Omega(\rho^* I; (\varphi \circ \rho)|\rho'|),$$

where $|\rho'|$ *denotes the Jacobian of* ρ.

ii *If* φ *is positive and bounded on* Θ *then*

$$\mathrm{RLCT}_\Theta(I;\varphi) = \mathrm{RLCT}_\Theta(I).$$

iii *If there exist constants* $c, c' > 0$ *such that* $c\,g(\theta) \le f(\theta) \le c' g(\theta)$ *for every* $\theta \in \Theta$ *then* $\mathrm{RLCT}_\Theta(f) = \mathrm{RLCT}_\Theta(g)$.

iv *Let* $I = \langle f_1, \ldots, f_r \rangle$ *and* $J = \langle g_1, \ldots, g_r \rangle$ *where* $g_i = u_i f_i$ *for* $i = 1, \ldots, r$ *and there exist positive constants* c, C *such that* $c < u_i(\theta) < C$ *for all* $\theta \in \Theta$ *and for all* $i = 1, \ldots, r$. *Then* $\mathrm{RLCT}_\Theta(I) = \mathrm{RLCT}_\Theta(J)$.

The ideal-theoretic approach proves to be useful in a fairly general statistical context:

Theorem 11 (Lin, 2011) *Let* $p = (p_1, \ldots, p_m) : \Theta \to \Delta_{m-1}$ *be a polynomial mapping and* $\mathcal{M} = p(\Theta)$ *be the statistical model of* X *with values in* $[m]$. *For a given point* $q \in \mathcal{M}$ *define*

$$\overline{\mathscr{I}} = \langle p_1(\theta) - q_1, \ldots, p_m(\theta) - q_m \rangle \subset \mathcal{A}_\Theta. \tag{8}$$

Let q *denote the true data generating distribution and* $K(\theta)$ *be the corresponding Kullback-Leibler distance defined in (3). Moreover, let* φ *the prior distribution on* Θ *satisfying (A1). Then*

$$\mathrm{RLCT}_\Theta(K;\varphi) = \mathrm{RLCT}_\Theta(\overline{\mathscr{I}};\varphi) = \mathrm{RLCT}_\Theta(\overline{\mathscr{I}}), \tag{9}$$

where the second equation in (9) follows from Proposition 10 ii.

We now perform the change of coordinates $f_{\theta\omega} : \Theta_T \to \Omega_T$, $f_{p\kappa} : \Delta_{2^n-1} \to \mathcal{K}_T$ discussed in detail by Zwiernik and Smith (2011b). We have the following diagram, where the top row is the original parametrization and where the induced parameterisation ψ_T is given in the bottom row.

$$\begin{array}{ccc} \Theta_T & \xrightarrow{\quad p \quad} & \Delta_{2^n-1} \\ {\scriptstyle f_{\omega\theta}}\uparrow \;\downarrow{\scriptstyle f_{\theta\omega}} & & {\scriptstyle f_{\kappa p}}\uparrow \;\downarrow{\scriptstyle f_{p\kappa}} \\ \Omega_T & \overset{\psi_T}{\dashrightarrow} & \mathcal{K}_T \end{array}$$

Here $f_{\theta\omega}$ and $f_{p\kappa}$ are polynomial isomorphisms, and hence, by Proposition 10 (i), in our computations of the real-log canonical threshold, we can alternatively constrain to the bottom row of the diagram. We denote the coordinates of $\mathcal{K}_T$ by $\kappa = (\kappa_I)$ for $I \subseteq [n], I \neq \emptyset$. The coordinates of Ω_T are denoted by $\omega = ((s_v), (\eta_{uv}))$ for all $v \in V$ and $(u,v) \in E$. Both ω and κ have a statistical meaning as described by Zwiernik and Smith (2011b). However, in this work we use them in a purely algebraic manner. We just note that the coordinates of $\mathcal{K}_T$ are certain functions of the moments. In particular, $\kappa_i = \mathbb{E}X_i$ for $i = 1, \ldots, n$ and each κ_{ij} corresponds to the covariance between X_i and X_j. Interpretation of other coordinates is more complicated.

Simple linear constraints defining Θ_T become only slightly more complicated when expressed in the new parameters. The choice of parameter values is not free anymore in the sense that constraints for each of the parameters involve other parameters. The new parameter space Ω_T is given by $s_r \in [-1,1]$ and for each $(u,v) \in E$ (cf. Equation (19) in Zwiernik and Smith, 2011b)

$$\begin{aligned} -(1+s_v) &\leq (1-s_u)\eta_{uv} \leq (1-s_v) \\ -(1-s_v) &\leq (1+s_u)\eta_{uv} \leq (1+s_v). \end{aligned} \tag{10}$$

In the new coordinate system the situation is more tractable because ψ_T has a simpler structure.

Proposition 12 (Zwiernik and Smith, 2011b) *Let $T^r = (V,E)$ be a rooted trivalent tree with n leaves. Then for each $i = 1,\ldots,n$ one has $\kappa_i(\omega) = \frac{1}{2}(1-s_i)$ and*

$$\kappa_I(\omega) = \frac{1}{4}(1-s_{r(I)}^2) \prod_{v \in V(I)\setminus I} s_v^{\deg v - 2} \prod_{(u,v)\in E(I)} \eta_{uv} \quad \text{for all } |I| \geq 2,$$

where the degree of $v \in V(I)$ is considered in $T(I) = (V(I),E(I))$, which is the smallest subtree of T containing I.

Let $\mathscr{I}$ denote the pullback of the ideal $\overline{\mathscr{I}} \subseteq \mathcal{A}_{\Theta_T}$ to the ideal in $\mathcal{A}_{\Omega_T}$ induced by $f_{\theta\omega}$. Thus $\mathscr{I} = f_{\omega\theta}^* \overline{\mathscr{I}} = \{f \circ f_{\omega\theta} : f \in \overline{\mathscr{I}}\}$. The ideal describes $\widehat{\Omega}_T = f_{\theta\omega}(\widehat{\Theta}_T)$ as a subset of Ω_T. Let $[n]_{\geq k}$ denote all subsets of $[n]$ with at least k elements. Then the pullback of $\overline{\mathscr{I}}$ satisfies

$$\mathscr{I} = \langle \kappa_1 - \hat{\kappa}_1, \ldots, \kappa_n - \hat{\kappa}_n \rangle + \left(\sum_{I \in [n]_{\geq 2}} \langle \kappa_I(\omega) - \hat{\kappa}_I \rangle \right), \tag{11}$$

where $\hat{\kappa}_I$ are the corresponding coordinates of $f_{p\kappa}(q)$. Here the sum of ideals results in another ideal with the generating set which is the sum of generating sets of the summands.

For local computations we use the following reduction.

Proposition 13 (Lin, 2011) *Let $I \subseteq \mathcal{A}_{x_0}(\mathbb{R}^m)$, $J \subseteq \mathcal{A}_{y_0}(\mathbb{R}^n)$ be two ideals. If $\mathrm{RLCT}_{x_0}(I) = (\lambda_x, m_x)$ and $\mathrm{RLCT}_{y_0}(J) = (\lambda_y, m_y)$ then*

$$\mathrm{RLCT}_{(x_0,y_0)}(I+J) = (\lambda_x + \lambda_y, m_x + m_y - 1).$$

Theorem 14 *Let T^r be a rooted tree with n leaves and $q \in \mathcal{M}_T$. Let $\overline{\mathscr{I}}$ be the ideal defined by (8) and $\mathscr{I}$ the ideal defined by (11). Then*

$$\mathrm{RLCT}_{\Theta_T}(\overline{\mathscr{I}}) = \mathrm{RLCT}_{\Omega_T}(\mathscr{I}) = \min_{\omega_0 \in \widehat{\Omega}_T} \mathrm{RLCT}_{\Omega_0}(\mathscr{I}),$$

where Ω_0 is a sufficiently small neighborhood of ω_0 in Ω_T. Moreover, let $\mathscr{J} = \sum_{I\in[n]_{\geq 2}} \langle \kappa_I(\omega) - \hat{\kappa}_I \rangle$. Then, for every $\omega_0 \in \widehat{\Omega}_T$

$$\mathrm{RLCT}_{\omega_0}(\mathscr{I}) = \left(\frac{n}{2}, 0\right) + \mathrm{RLCT}_{\omega_0}(\mathscr{J}). \tag{12}$$

Proof Since $f_{\omega\theta}$ is an isomorphism with a constant Jacobian then the first part of the theorem follows from Proposition 10 (i). Let now W be an ε-box around $\omega_0 = ((s_v^0),(\eta_{uv}^0))$. If T is rooted in an inner leaf then by Proposition 12 the ideal $\mathscr{J}$ does not depend on $s_1,\ldots,s_n$. Since for every $i = 1,\ldots,n$ the expression $\kappa_i - \hat{\kappa}_i$ depends only on s_i then

$$\mathrm{RLCT}_{\omega_0}(\langle \kappa_1 - \hat{\kappa}_1, \ldots, \kappa_n - \hat{\kappa}_n \rangle) = (\frac{n}{2}, 1),$$

which can be easily checked (see for example Proposition 3.3 of Saito, 2007). Equation (12) follows from Proposition 13.

Now assume that T is rooted in one of the leaves. In this case both $\langle \kappa_1 - \hat{\kappa}_1, \ldots, \kappa_n - \hat{\kappa}_n \rangle$ and $\mathscr{J}$ depend on s_r because $\kappa_I(\omega) = (1 - s_r^2) f_I(\omega)$ for some monomial $f_I(\omega)$ whenever $r \in I$. Therefore, we cannot use Proposition 13 directly. However, by assumption (A2), q lies in the interior of the probability simplex and hence $\hat{\kappa}_i \in (0,1)$ for $i = 1,\ldots,n$ which is equivalent to $s_i^0 \in (-1,1)$. Therefore, for each ω_0 one can find two positive constants c, C such that $c \leq 1 - s_r^2 \leq C$ in W. By Proposition 10 (iv) the real log canonical threshold of $\mathscr{J}$ in W is equal to the real log-canonical threshold of a an ideal with generators induced from the generators of $\mathscr{J}$ by replacing each $1 - s_r^2$ by 1. Now again (12) follows from Proposition 13. ■

5. The Main Reduction Step

Recall that $\hat{\kappa}_{ij} = \mathrm{Cov}(X_i, X_j)$. In this section we prove a technical result which enables us to reduce the computations of $\mathrm{RLCT}_{\omega_0}(\mathscr{J})$ to two simpler cases. First, when q is such that $\hat{\kappa}_{ij} \neq 0$ for all $i, j \in [n]$. Second, when q is such that $\hat{\kappa}_{ij} = 0$ for all $i, j \in [n]$. Moreover, the second case is reduced to computations for monomial ideals which are amenable to various combinatorial techniques.

Let T be a trivalent tree with $n \geq 3$ leaves and let $q \in \mathcal{M}_T$. If all the equivalence classes in $[\widehat{E}]$ are singletones or $[\widehat{E}]$ is empty, which is equivalent to every inner node being of degree at least two in $\widehat{T}$, then Theorem 1 gives us the asymptotic behavior of the marginal likelihood. Thus, let assume that there is at least one class in $[\widehat{E}]$ which is not a singleton. Let $T_1,\ldots,T_k$ denote trees representing the equivalence classes in $[\widehat{E}]$ and let $S_1,\ldots,S_m$ denote trees induced by the connected components of $E \setminus \widehat{E}$. Let $L_1,\ldots L_k$ denote the sets of leaves of $T_1,\ldots,T_k$. For each S_i $i = 1,\ldots,m$ by Remark 5.2 (iv) of Zwiernik and Smith (2011b) its set of leaves denoted by $[n_i]$ is a subset of $[n]$. For each S_i the number of nodes, edges and nodes of degree 2 in $\widehat{T}$ is denoted by n_v^i, n_e^i and l_2^i respectively. We illustrate this notation in Figure 4 where the dashed edges represent edges in $\widehat{E}$. Simpler examples are given in Figure 3.

Lemma 15 *Let $T = (V,E)$ be a trivalent rooted tree with $n \geq 4$ leaves and let $q \in \mathcal{M}_T$. Let $\mathscr{J} = \sum_{I \in [n]_{\geq 2}} \langle \kappa_I(\omega) - \hat{\kappa}_I \rangle$ as in Theorem 14. If $\omega_0 \in \widehat{\Omega}_T$ then*

$$\mathrm{RLCT}_{\omega_0}(\mathscr{J}) = \sum_{i=1}^{m} \mathrm{RLCT}_{\omega_0}(\mathscr{J}(S_i)) + \sum_{i=1}^{k} \mathrm{RLCT}_{\omega_0}(\mathscr{J}(T_i)) + (0, 1 - m - k), \tag{13}$$

where $\mathscr{J}(S_i) = \sum_{I \in [n_i]_{\geq 2}} \langle \kappa_I(\omega) - \hat{\kappa}_I \rangle$ for $i = 1,\ldots,m$ and $\mathscr{J}(T_i) = \sum_{w,w' \in L_i} \langle \kappa_{ww'}(\omega) \rangle$ for $i = 1,\ldots,k$.

Proof We first show that $\sum_{I:\hat{\kappa}_I = 0} \langle \kappa_I(\omega) \rangle = \sum_{i,j:\hat{\kappa}_{ij}=0} \langle \kappa_{ij}(\omega) \rangle$. The inclusion "$\supseteq$" is clear. We now show "$\subseteq$". First note that for every $I \in [n]_{\geq 2}$ if $\hat{\kappa}_I = 0$ then either $\eta_e^0 = 0$ for an edge $e \in E(I)$ or

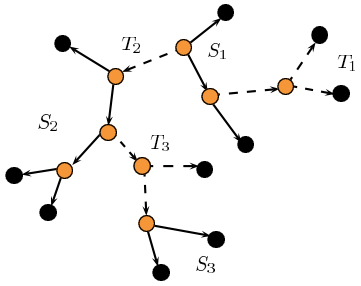


Figure 4: An example of a forest $\widehat{T}$ induced by a point q.

$s^2_{r(I)} = 1$. There exist $i, j \in I$ such that such that $\hat{\kappa}_{ij} = 0$ and the $r(ij) = r(I)$. It follows by Proposition 12 that $\kappa_I(\omega) = \kappa_{ij}(\omega) f(\omega)$ for a polynomial $f(\omega)$ and therefore the inclusion "$\subseteq$" is also true. This implies

$$\mathscr{J} = \sum_{I:\hat{\kappa}_I \neq 0} \langle \kappa_I(\omega) - \hat{\kappa}_I \rangle + \sum_{I:\hat{\kappa}_I = 0} \langle \kappa_I(\omega) \rangle = \sum_{i=1}^{m} \mathscr{J}(S_i) + \sum_{i,j:\hat{\kappa}_{ij}=0} \langle \kappa_{ij}(\omega) \rangle.$$

Hence, to proof the lemma, it suffices to show that for every $\omega_0 \in \widehat{\Omega}_T$

$$\mathrm{RLCT}_{\omega_0}\Big(\sum_{i=1}^{m} \mathscr{J}(S_i) + \sum_{i,j:\hat{\kappa}_{ij}=0} \langle \kappa_{ij}(\omega) \rangle\Big) \tag{14}$$

is equal to the right hand side of (13).

If $e \in E \setminus \widehat{E}$ then by definition there exist $i, j \in [n]$ such that $\hat{\kappa}_{ij} \neq 0$ and $e \in E(ij)$. Since, by Proposition 12, $\hat{\kappa}_{ij} = \eta_e^0 f(\omega_0)$ for a polynomial f then in particular $\eta_e^0 \neq 0$. It follows that for a sufficiently small ε for each $E' \subseteq E \setminus \widehat{E}$ one can find positive constants $c(\varepsilon), C(\varepsilon)$ such that $c(\varepsilon) \leq \prod_{e \in E'} \eta_e \leq C(\varepsilon)$ holds in the ε-box around ω_0. Similarly if $v \notin \widehat{V}$ (cf. Section 3.2) then there exist positive constants $d(\varepsilon), D(\varepsilon)$ such that $d(\varepsilon) \leq (1 - s_v^2) \leq D(\varepsilon)$ in the ε-box around ω_0. It follows by Proposition 10 (iv) that in computations of the real log-canonical threshold in (14) we can replace each $\kappa_{ij}(\omega)$ by

$$(1 - s^2_{r(ij)})^{\delta_{r(ij)}} \prod_{e \in E(ij) \cap \widehat{E}} \eta_e \tag{15}$$

where $\delta_{r(ij)} = 1$ if $r(ij) \in \widehat{V}$ and $\delta_{r(ij)} = 0$ otherwise. Thus, in (14) we can replace the ideal $\sum_{i,j:\hat{\kappa}_{ij}=0} \langle \kappa_{ij}(\omega) \rangle$ by the ideal $\mathscr{J}_1 = \sum_{i,j:\hat{\kappa}_{ij}=0} \langle (1 - s^2_{r(ij)})^{\delta_{r(ij)}} \prod_{e \in E(ij) \cap \widehat{E}} \eta_e \rangle$. However, if we define

$$\mathscr{J}_2 = \sum_{i=1}^{k} \sum_{w,w' \in L_i} \langle (1 - s^2_{r(ww')})^{\delta_{r(ww')}} \prod_{e \in E(ww')} \eta_e \rangle \tag{16}$$

then it can be checked that $\mathscr{J}_1 = \mathscr{J}_2$. To show that $\mathscr{J}_2 \subseteq \mathscr{J}_1$, fix $j = 1, \ldots, k$ and $w, w' \in L_j$, and show that the corresponding generator of $\mathscr{J}_2$ lies in $\mathscr{J}_1$. Note that by construction each of w, w' either has degree two in $\widehat{T}$ or is a leaf of T. Hence, by the definition of $\widehat{E}$, there exist $i, j \in [n]$ such that $E(ij) \cap \widehat{E} = E(ww')$. It follows that each generator in (16) is also in the set of generators of $\mathscr{J}_1$ and hence $\mathscr{J}_2 \subseteq \mathscr{J}_1$. To show the opposite inclusion, note that, if $E(ij)$ intersects with more than one component $T_1, \ldots, T_k$ then the corresponding generator in (15) is a product of some generators in (16) and hence it lies in $\mathscr{J}_2$.

Since the generators of every $\mathscr{J}(S_i)$ for $i=1,\ldots,m$ and every

$$\sum_{w,w'\in L_j}\langle(1-s^2_{r(ww')})^{\delta_{r(ww')}}\prod_{e\in E(ww')}\eta_e\rangle$$

for $j=1,\ldots,k$ involve disjoint sets of variables then by Proposition 13 the term in (14) is equal to

$$\sum_{i=1}^{m}\mathrm{RLCT}_{\omega_0}(\mathscr{J}(S_i))+\sum_{i=1}^{k}\mathrm{RLCT}_{\omega_0}\left(\sum_{w,w'\in L_j}\langle(1-s^2_{r(ww')})^{\delta_{r(ww')}}\prod_{e\in E(ww')}\eta_e\rangle\right)+(0,1-m-k).$$

Again by Proposition 10 (iv) for each $i=1,\ldots,k$

$$\mathrm{RLCT}_{\omega_0}\left(\sum_{w,w'\in L_i}\langle(1-s^2_{r(ww')})^{\delta_{r(ww')}}\prod_{e\in E(ww')}\eta_e\rangle\right)=\mathrm{RLCT}_{\omega_0}(\mathscr{J}(T_i))$$

which finishes the proof. ■

We note that, by Proposition 8 and the formula in (12) for each S_i:

$$\mathrm{RLCT}(\mathscr{J}(S_i))+\frac{n_i}{2}=\frac{n_v^i+n_e^i-2l_2^i}{2}. \tag{17}$$

6. The Case of Zero Covariances

In this subsection we assume that $q\in\mathcal{M}_T$ is such that $\hat{\kappa}_{ij}=0$ for all $i,j\in[n]$. This implies the full joint marginal independence $X_1\perp\!\!\!\perp\cdots\perp\!\!\!\perp X_n$. The aim is to prove the following proposition.

Proposition 16 *Let T be a trivalent tree with $n\neq 3$ leaves rooted in $r\in V$. Let $q\in\mathcal{M}_T$ be such that $\hat{\kappa}_{ij}=0$ for all $i,j\in[n]$. Let $\mathscr{J}$ be the ideal defined in Theorem 14. Then*

$$\min_{\omega_0\in\widehat{\Omega}_T}\mathrm{RLCT}_{\omega_0}(\mathscr{J})=\left(\frac{n}{4},m\right),$$

where $m=1$ if either r is a leaf of T or r together with all its neighbors are all inner nodes of T. In all other cases we cannot obtain an explicit upper bound for m and hence $m\geq 1$.

The strategy of the proof of Proposition 16 is as follows. First, in Section 6.1, we show that the local computations can be restricted to a special subset of $\widehat{\Omega}_T$ over which $\mathscr{J}$ can be replaced by a monomial ideal. Then, in Section 6.2, we present a method to compute the real log-canonical threshold of a monomial ideal. We use this method in Section 6.3.

6.1 The Deepest Singularity

Note that, by Lemma 15, $\mathrm{RLCT}_{\omega_0}(\mathscr{J})=\mathrm{RLCT}_{\omega_0}(\sum_{i,j\in[n]}\langle\kappa_{ij}(\omega)\rangle)$ so without loss of generality we will assume in this section that $\mathscr{J}=\sum_{i,j\in[n]}\langle\kappa_{ij}(\omega)\rangle$. Moreover, for each $v\in V$ these ideals depend on s_v only through the value of s_v^2. It follows that the computations can be reduced only to points satisfying $s_v\geq 0$ for all inner nodes v of T. Henceforth, in this section, we always assume this is the case. We define the *deepest singularity* of $\widehat{\Omega}_T$ as

$$\widehat{\Omega}_{\mathrm{deep}}:=\{\omega\in\widehat{\Omega}_T:\eta_e=0\text{ for all }e\in\widehat{E},\ s_v=1\text{ for all }v\in\widehat{V}\}.$$

We note that, since $\hat{\kappa}_{ij}=0$ for all $i,j\in[n]$, then $\widehat{E}=E$ and $\widehat{V}$ is equal to the set of all inner nodes of T. It follows that $\widehat{\Omega}_{\text{deep}}$ is an affine subspace constrained to Ω_T with all coordinates either 0 or 1.

Proposition 17 *Let T be a tree with n leaves. Let $q\in\mathcal{M}_T$ such that $\hat{\kappa}_{ij}=0$ for all $i,j\in[n]$. Then*

$$\min_{\omega_0\in\widehat{\Omega}_T}\text{RLCT}_{\omega_0}(\mathscr{J})=\min_{\omega_0\in\widehat{\Omega}_{\text{deep}}}\text{RLCT}_{\omega_0}(\mathscr{J}).$$

Proof We first show that $\widehat{\Omega}_T$ is a union of affine subspaces constrained to Ω_T with a common intersection given by $\widehat{\Omega}_{\text{deep}}$. Let $V_0\subseteq\widehat{V}$ and $E_0\subseteq\widehat{E}$ and

$$\Omega_{(V_0,E_0)}=\{\omega\in\widehat{\Omega}_T:\ s_v=1\text{ for all }v\in V_0,\ \eta_{uv}=0\text{ for all }(u,v)\in E_0\}.$$

We say that (V_0,E_0) is *minimal for* $\widehat{\Sigma}$ if for every point ω in $\Omega_{(V_0,E_0)}$ and for every $i,j\in[n]$ $\kappa_{ij}(\omega)=0$, and furthermore, that (V_0,E_0) is minimal with such a property (with respect to inclusion on both coordinates). We now show that

$$\widehat{\Omega}_T=\bigcup_{(V_0,E_0)\text{ min.}}\Omega_{(V_0,E_0)}.$$

The first inclusion "$\subseteq$" follows from the fact that if $\omega\in\widehat{\Omega}_T$ then $\kappa_{ij}(\omega)=\hat{\kappa}_{ij}=0$ for all $i,j\in[n]$. Therefore $\omega\in\Omega_{(V_0,E_0)}$ for some minimal (V_0,E_0). The second inclusion is obvious.

Each $\Omega_{(V_0,E_0)}$ is an affine subspace in $\mathbb{R}^{|V|+|E|}$, denoted by $M_{(V_0,E_0)}$, constrained to Ω_T. Let $\mathcal{S}$ denote the intersection lattice of all $M_{(V_0,E_0)}$ for (V_0,E_0) minimal with ordering denoted by $\leq$. For each $i\in\mathcal{S}$ let $M^{(i)}$ denote the corresponding intersection and define

$$S_i=M^{(i)}\setminus\bigcup_{j<i}M^{(j)}.$$

In this way we obtain an *$\mathcal{S}$-induced decomposition* of $\mathbb{R}^{|V|+|E|}$ (cf. Section 3.1 in Goresky and MacPherson, 1988).

By (Lazarsfeld, 2004, Example 9.3.17) the function $\omega\mapsto\text{rlct}_\omega(\mathscr{J})$ is lower semicontinuous (the argument used there works over the real numbers). This means that for every $\omega_0\in\Omega_T$ and $\varepsilon>0$ there exists a neighborhood U of ω_0 such that $\text{rlct}_{\omega_0}(\mathscr{J})\leq\text{rlct}_\omega(\mathscr{J})+\varepsilon$ for all $\omega\in U$. Since the set of values of the real log-canonical threshold is discrete this means that for every $\omega_0\in\widehat{\Omega}_T$ and any sufficiently small neighborhood W_0 of ω_0, one has $\text{rlct}_{\omega_0}(\mathscr{J})\leq\text{rlct}_\omega(\mathscr{J})$ for all $\omega\in W_0$. Moreover, $\text{rlct}(\mathscr{J})$ is constant on each S_i. Since for any neighborhood W_0 of $\omega_0\in\widehat{\Omega}_{\text{deep}}$ we have $W_0\cap S_i\neq\emptyset$ for all $i\in\mathcal{S}$ then necessarily the minimum of the real log-canonical threshold is attained for a point in the deepest singularity. ■

Proposition 17 shows that in the singular case we can restrict our analysis to the neighborhood of $\widehat{\Omega}_{\text{deep}}$. Often however, we also consider points in a bigger set

$$\widehat{\Omega}_0=\{\omega\in\widehat{\Omega}_T:\eta_{uv}=0\text{ for all }(u,v)\in\widehat{E}\}.$$

Note that $\widehat{\Omega}_{\text{deep}}$ lies on the boundary of Ω_T (cf. (10)) but $\widehat{\Omega}_0$ also contains internal points of Ω_T which will be crucial for some of the arguments later.

We now formulate another technical lemma which enables us to reduce computations to the monomial case.

Lemma 18 *Assume that $q \in \mathcal{M}_T$ is such that $\hat{\kappa}_{ij} = 0$ for all $i, j \in [n]$. Let $\mathscr{J}(\omega_0)$ be the ideal $\mathscr{J}$ translated to the origin. Then for every $\omega_0 \in \widehat{\Omega}_0$*

$$\mathrm{RLCT}_0(\mathscr{J}(\omega_0)) = \mathrm{RLCT}_0(\mathscr{J}'), \tag{18}$$

where $\mathscr{J}'$ is a monomial ideal such that each $\kappa_{ij}(\omega+\omega_0)$ in the set of generators of $\mathscr{J}(\omega_0)$ is replaced either by

$$\begin{array}{ll} s_{r(ij)} \prod_{(u,v)\in E(ij)} \eta_{uv} & \text{if } s^0_{r(ij)} = 1, \text{ or by} \\ \prod_{(u,v)\in E(ij)} \eta_{uv} & \text{if } s^0_{r(ij)} \neq 1. \end{array}$$

Proof Let $i, j \in [n]$ and assume that $\omega_0 = ((s^0_v), (\eta^0_e)) \in \widehat{\Omega}_0$ so that $\eta^0_e = 0$ for all $e \in E$. Then, by Proposition 12:

$$\kappa_{ij}(\omega+\omega_0) = \frac{1}{4}(1-(s_{r(ij)}+s^0_{r(ij)})^2) \prod_{e\in E(ij)} \eta_e. \tag{19}$$

If $s^0_{r(ij)} \neq 1$ for a sufficiently small $\varepsilon > 0$ there exist positive constants $c(\varepsilon), C(\varepsilon)$ such that $c(\varepsilon) < 1-(s_{r(ij)}+s^0_{r(ij)})^2 < C(\varepsilon)$ for $s_{r(ij)} \in (-\varepsilon, \varepsilon)$. Therefore, by Proposition 10 (iv), we can replace this term in (19) with 1. If $s^0_{r(ij)} = 1$ rewrite $1-(1+s^2_{r(ij)})^2$ as $-s_{r(ij)}(2+s_{r(ij)})$. For a sufficiently small ε we can find two positive constants $c(\varepsilon), C(\varepsilon)$ such that $c < 2+s_{r(ij)} < C$ whenever $s_{r(ij)} \in (-\varepsilon, \varepsilon)$. Again, by Proposition 10 (iv), we can replace $2+s_{r(ij)}$ with 1. This proves Equation (18). ■

Since $\mathscr{J}'$ is a monomial ideal then, by (Lin, 2011, Proposition 4.11) and Theorem 20 below, we can compute $\mathrm{RLCT}_0(\mathscr{J}')$ using the method of Newton diagrams. We present this method in the following subsection.

6.2 Newton Diagram Method

Given an analytic function $f \in \mathcal{A}_0(\mathbb{R}^d)$ we pick local coordinates $x = (x_1, \ldots, x_d)$ in a neighborhood of the origin. This allows us to represent f as a power series in $x_1, \ldots, x_d$ such that $f(x) = \sum_\alpha c_\alpha x^\alpha$. The exponents of terms of the polynomial f are vectors in $\mathbb{N}^d$. The *Newton polyhedron* of f denoted by $\Gamma_+(f)$ is the convex hull of the subset

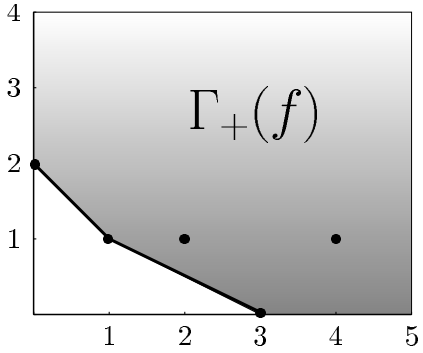
$$\{\alpha+\alpha' : c_\alpha \neq 0, \alpha' \in \mathbb{R}^d_{\geq 0}\}.$$

A subset $\gamma \subset \Gamma_+(f)$ is a *face* of $\Gamma_+(f)$ if there exists $\beta \in \mathbb{R}^d$ such that

$$\gamma = \{\alpha \in \Gamma_+(f) : \langle \alpha, \beta\rangle \leq \langle \alpha', \beta\rangle \text{ for all } \alpha' \in \Gamma_+(f)\}.$$

If γ is a subset of $\Gamma_+(f)$ then we define $f_\gamma(x) = \sum_{\alpha\in\gamma\cap\mathbb{N}^d} c_\alpha x^\alpha$. The *principal part* of f is, by definition, the sum of all terms of f supported on all compact faces of $\Gamma_+(f)$.

Example 4 *Let $f(x,y) = x^3 + 2xy + 6x^2y + 3x^4y + y^2$. Then the Newton diagram looks as follows:*



where the dots correspond to the terms of f. There are only two bounded facets of $\Gamma_+(f)$ and the principal part of f is equal to x^3+xy+y^2.

Definition 19 *The principal part of the power series f with real coefficients is $\mathbb{R}$-nondegenerate if for all compact faces γ of $\Gamma_+(f)$*

$$\left\{x\in\mathbb{R}^n:\ \frac{\partial f_\gamma}{\partial x_1}(x)=\cdots=\frac{\partial f_\gamma}{\partial x_n}(x)=0\right\}\subseteq\{\omega\in\mathbb{R}^n: x_1\cdots x_n=0\}.$$

From the geometric point of view this condition means that the singular locus of the hypersurface defined by $f_\gamma(x)=0$ lies outside of $(\mathbb{R}^*)^n$ for all compact faces γ of $\Gamma_+(f)$.

The following theorem shows that, if the principal part of f is $\mathbb{R}$-nondegenerate and $f\in\mathcal{A}_\Theta^{\geq}$, the computations are greatly facilitated. An example of an application of these methods in statistical analysis can be found in Yamazaki and Watanabe (2004).

Theorem 20 (Arnold et al., 1988) *Let $f\in\mathcal{A}_0^{\geq}(\mathbb{R}^d)$ and $f(0)=0$. If the principal part of f is $\mathbb{R}$-nondegenerate then $\mathrm{RLCT}_0(f)=(\frac{1}{t},c)$ where t is the smallest number such that the vector $(t,\ldots,t)$ hits the polyhedron $\Gamma_+(f)$ and c is the codimension of the face it hits.*

For a proof see Theorem 4.8, Lin (2011).

Let now $f\in\mathcal{A}_{\theta_0}^{\geq}$ such that $f(\theta_0)=0$. We can then center f at θ_0 obtaining a function in $\mathcal{A}_0^{\geq}$. Then we can use Theorem 20 to compute $\mathrm{RLCT}_{\theta_0}(f)$.

Remark 21 *Note that this theorem in general will not give us $\mathrm{RLCT}_{\Theta_0}(f)$ if 0 is a boundary point of Θ in which case we also need to resolve the defining inequalities. For a discussion see (Arnold et al., 1988, Section 8.3.4) and Example 2.7 in Lin (2011).*

6.3 Proof of Proposition 16

Let $n\geq 4$. For each $\omega_0\in\widehat{\Omega}_0$, let $\delta=\delta(\omega_0)\in\{0,1\}^V$ denote the indicator vector satisfying $\delta_v=1$ if $v\in V$ is such that $s_v^0=1$ and $\delta_v=0$ otherwise. In particular $\delta_i=0$ for all $i=1,\ldots,n$ because the leaves, by (A2), are assumed to be non-degenerate. Let $\mathcal{V}_\delta=\mathbb{R}^{n_e+|\delta|}=\mathbb{R}^{|\delta|}\times\mathbb{R}^{n_e}$, where $|\delta|=\sum_v\delta_v$, be the real space with variables representing the edges $(x_e)_{e\in E}$ and nodes (y_v) for all v such that $\delta_v=1$. With some arbitrary numbering of the nodes and edges we order the variables as follows: $y_1\prec\cdots\prec y_{|\delta|}\prec x_{e_1}\prec\cdots\prec x_{e_{n_e}}$. In Lemma 18, for each $\omega_0\in\widehat{\Omega}_0$, we reduced our computations to the analysis of $\mathrm{RLCT}_0(\mathscr{J}')$ where $\mathscr{J}'$ has a simple monomial form. Let Q_δ be a polynomial

function on Ω_T defined as a sum of squares of generators of $\mathscr{J}'$. In particular $\mathrm{RLCT}_0(\mathscr{J}') = \mathrm{RLCT}_0(Q_\delta)$. The exponents of terms of the polynomial $Q_\delta(\omega)$ are vectors in $\{0,2\}^{n_e+|\delta|}$. We have that

$$Q_\delta(\omega) = \sum_{i\neq j\in[n]} s_{r(ij)}^{2\delta_{r(ij)}} \prod_{(u,v)\in E(ij)} \eta_{uv}^2. \tag{20}$$

The convex hull, in $\mathcal{V}_\delta$, of the exponents of the terms in Q_δ is called the *Newton polytope* of Q_δ and denoted $\Gamma(Q_\delta)$. We now investigate this polytope which is needed to understand the polyhedron $\Gamma_+(Q_\delta)$, which is needed to use Theorem 20. Since each term of Q_δ corresponds to a path between two leaves then the construction of the Newton polytope $\Gamma(Q_\delta) \subset \mathcal{V}_\delta$ gives a direct relationship between paths in T and the points generating the polytope. Convex combinations of points corresponding to paths give rise to points in the polytope. Let $E_0 \subseteq E$ be the subset of edges of T such that one of the ends is in the set of leaves of T. We call these edges *terminal*. Note that each point generating $\Gamma(Q_\delta)$ satisfies $\sum_{e\in E_0} x_e = 4$. This follows from the fact that each of these points corresponds to a path between two leaves in T and every such a path need to cross exactly two terminal edges. Consequently each point of $\Gamma(Q_\delta)$ needs to satisfy this equation as well. The induced facet of the Newton polyhedron $\Gamma_+(Q_\delta)$ is given as

$$F_0 = \{(\mathbf{y},\mathbf{x}) \in \Gamma_+(Q_\delta) : \sum_{e\in E_0} x_e = 4\} \tag{21}$$

and each point of $\Gamma_+(Q_\delta)$ satisfies $\sum_{e\in E_0} x_e \geq 4$.

The following lemma proves one part of Proposition 16.

Lemma 22 (The real log-canonical threshold of $\mathscr{J}$) *Under assumptions of Proposition 16 we have that* $\mathrm{rclt}_0(\mathscr{J}') = \frac{n}{4}$.

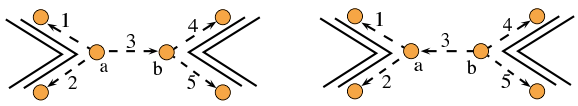
Proof If $n = 2$ then, since $s_1^0, s_2^0 \neq 1$, by Lemma 18 we have that

$$\mathrm{RLCT}_{\omega_0}(\mathscr{J}') = \mathrm{RLCT}_0(\eta_{12}^2) = \left(\frac{1}{2}, 1\right).$$

Therefore Proposition 16 holds in this case. Now assume that $n \geq 4$. By Theorem 20 we have to show that $t = \frac{4}{n}$ is the smallest t such that the vector $(t,\ldots,t)$ hits $\Gamma_+(Q_\delta)$. To show that $\frac{4}{n}\mathbf{1} \in \Gamma_+(Q_\delta)$ we construct a point $q \in \Gamma(Q_\delta)$ such that $q \leq \frac{4}{n}\mathbf{1}$ coordinatewise. The point is constructed as follows.

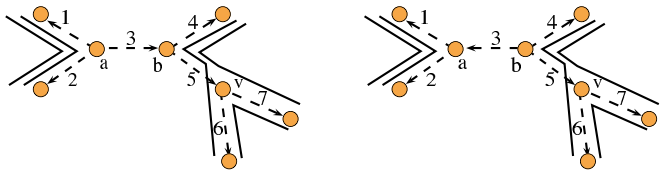
Construction 23 *Let $T = (V,E)$ be a trivalent tree with $n \geq 4$ leaves, rooted in r. We present two constructions of networks of paths between the leaves of T.*

The first construction is for the case when the root is degenerate, $\delta_r = 1$. In this case T is necessarily rooted in an inner node. If $n = 4$ then the network consists of the two paths within cherries counted with multiplicity two.



Each of the paths corresponds to a point in $\Gamma(Q_\delta)$. We order the coordinates of $\mathcal{V}_\delta = \mathbb{R}^{5+|\delta|}$ by $y_a \prec y_b \prec x_1 \prec \cdots \prec x_5$ where y_a, y_b are included only if $\delta_a, \delta_b = 1$. For example the point corresponding to the path involving edges e_1 and e_2 is $(2,0;2,2,0,0,0)$. The barycenter of the points corresponding to all the four paths in the network is $(1,1;1,1,0,1,1)$ both if T is rooted in a or b.

If $n > 4$ then we build the network recursively. Assume that T is rooted in an inner node a and pick an inner edge (a,b). Label the edges incident with a and b as for the quartet tree above and consider the subtree given by the quartet tree. Draw four paths as on the picture above. Let v be any leaf of the quartet subtree which is not a leaf of T and label the two additional edges incident with v by e_6 and e_7. Now we extend the network by adding e_6 to one of the paths terminating in v and e_7 to the other. Next we add an additional path involving only e_6 and e_7 like on the picture below. By construction v is the root of the additional path. We extend the network cherry by cherry until it covers all terminal edges.



Note that we have made some choices building up the network and hence the construction is not unique. However, each of the inner nodes is always a root of at least one and at most two paths. Moreover, each edge is covered at most twice and each terminating edge is covered exactly two times. We have n paths in the network, all representing points of $\Gamma(Q_\delta)$ denoted by $q_1,\dots,q_n$. Let $q = \frac{1}{n}\sum_{i=1}^n q_i$ then $q \in \Gamma(Q_\delta)$ is given by $x_{ab} = 0$, $x_e = \frac{4}{n}$ for all $e \in E \setminus (a,b)$. The other coordinates by construction satisfy $y_a = \frac{4}{n}$, $y_b = \frac{4}{n}$ if $\delta_b = 1$, and $y_v = \frac{2}{n}$ for all $v \in V \setminus \{a,b\}$ such that $\delta_v = 1$.

If $\delta_r = 0$ then we proceed as follows. For $n = 4$ consider a network of all the possible paths all counted with multiplicity one apart from the cherry paths (paths of length two) counted with multiplicity two. This makes eight paths and each edge is covered exactly four times. The coordinates of the point representing the barycenter of all paths in the network satisfy $x_e = 1$ for all $e \in E$ and $y_v = \frac{1}{2}$ for all v such that $\delta_v = 1$. This construction generalizes recursively in a similar way as the one for T rooted in an inner node. We always have $2n$ paths and each edge is covered exactly four times. The network induces a point $q \in \Gamma(Q_\delta)$ with coordinates given by $y_v = \frac{2}{n}$ for all $v \in V$ such that $\delta_v = 1$ and $x_e = \frac{4}{n}$ for $e \in E$. (This finishes the construction.)

The point $\frac{4}{n}\mathbf{1}$ lies in $\Gamma_+(Q_\delta)$, which follows from Construction 23 and the fact that the constructed point $q \in \Gamma(Q_\delta)$ satisfies $q \leq \frac{4}{n}\mathbf{1}$. Moreover, for any $s < \frac{4}{n}$ the point $s(1,\dots,1)$ does not satisfy $\sum_{e\in E_0} x_e \geq 4$ and hence it cannot be in $\Gamma_+(Q_\delta)$. It follows that $\frac{4}{n}$ is the smallest t such that $t\mathbf{1} \in \Gamma_+(Q_\delta)$ and therefore $\mathrm{rlct}_0(\mathscr{J}') = \frac{n}{4}$. Note that the result does not depend on δ.

■

To compute the multiplicity of the real log-canonical threshold of Q_δ we have to get a better understanding of the polyhedron $\Gamma_+(Q_\delta)$. According to Theorem 20 we need to find the codimension of the face of $\Gamma_+(Q_\delta)$ hit by the vector $\frac{4}{n}\mathbf{1}$. First we find the hyperplane representation of the Newton polytope $\Gamma(Q_\delta)$ reducing the problem to a simpler but equivalent one.

Definition 24 (A pair-edge incidence polytope) *Let $T = (V,E)$ be a trivalent tree with $n \geq 4$ leaves. We define a polytope $P_n \subset \mathbb{R}^{n_e}$, where $n_e = 2n-3$, as the convex combination of points $(q_{ij})_{i,j\in[n]}$ where k-th coordinate of q_{ij} is one if the k-th edge is in the path between i and j and there is zero otherwise. We call P_n a pair-edge incidence polytope by analogy to the pair-edge incidence matrix defined in (Mihaescu and Pachter, 2008, Definition 1).*

The reason to study the pair-edge incidence polytope is that its structure can be handled easily and it can be shown to be affinely equivalent to $\Gamma(Q_\delta)$. The latter is immediate if $\delta = (0,\ldots,0)$ since $Q_{\mathbf{0}} = 2P_n$. For an arbitrary δ fix a rooting r of T and define a linear map $f_r : \mathbb{R}^{n_e} \to \mathbb{R}^{|\delta|}$ as follows. For each $v \in V \setminus r$ such that $\delta_v = 1$ set

$$y_v = \frac{1}{2}(x_{v\mathrm{ch}_1(v)} + x_{v\mathrm{ch}_2(v)} - x_{\mathrm{pa}(v)v}),$$

where $\mathrm{ch}_1(v), \mathrm{ch}_2(v)$ denotes the two children of v. If $\delta_r = 1$ then set

$$y_r = \frac{1}{2}(x_{r\mathrm{ch}_1(r)} + x_{r\mathrm{ch}_2(r)} + x_{r\mathrm{ch}_3(r)}).$$

The map $(\mathrm{id} \times f_r) : \mathbb{R}^{n_e} \to \mathbb{R}^{n_e} \times \mathbb{R}^{|\delta|}$ satisfies $(\mathrm{id} \times f_r)(2P_n) = \Gamma(Q_\delta)$ because, for each point, $y_r = 2$ if and only if the path crosses r and for any other node $y_v = 2$ if and only if the path crosses v and v is the root of the path, that is if the path crosses both children of v.

Lemma 25 *Let $P_n \subset \mathbb{R}^{n_e}$ be the pair-edge incidence polytope for a trivalent tree with n leaves where $n \geq 4$. Then $\dim(P_n) = n_e - 1 = 2n-4$. The unique equation defining the affine span of P_n is $\sum_{e\in E_0} x_e = 2$. For each inner node $v \in V$ let $e_1(v)$, $e_2(v)$, $e_3(v)$ denote the three adjacent edges. Then exactly $3(n-2)$ facets define P_n and they are given by*

$$x_{e_1(v)} + x_{e_2(v)} - x_{e_3(v)} \geq 0, \quad x_{e_2(v)} + x_{e_3(v)} - x_{e_1(v)} \geq 0, \quad \text{and } x_{e_3(v)} + x_{e_1(v)} - x_{e_2(v)} \geq 0 \quad \text{for all } v \in V. \tag{22}$$

Proof Let M_n be the pair-edge incidence matrix, that is a $\binom{n}{2} \times n_e$ matrix with rows corresponding to the points defining P_n. By (Mihaescu and Pachter, 2008, Lemma 1) the matrix has full rank and hence P_n has codimension one in $\mathbb{R}^{n_e}$. Moreover since each path necessarily crosses two terminal edges then each point generating P_n satisfies the equation $\sum_{e\in E_0} x_e = 2$ and hence this is the equation defining the affine subspace containing P_n.

Now we show that the inequalities give a valid facet description for P_n. This can be checked directly for $n = 4$ using POLYMAKE Gawrilow and Joswig (2005). Assume this is true for all $k < n$. By Q_n we will denote the polyhedron defined by the equation $\sum_{e\in E_0} x_e = 2$ and $3(n-2)$ inequalities given by (22). We want to show that $P_n = Q_n$. It is obvious that $P_n \subseteq Q_n$ since all points generating P_n satisfy the equation and the inequalities. We show that the opposite inclusion also holds.

Consider any cherry $\{e_1, e_2\} \subset E$ in the tree given by two leaves, which we denote by $1, 2$, and the separating inner node a. Define a projection $\pi : \mathbb{R}^{n_e} \to \mathbb{R}^{n_e-2}$ on the coordinates related to all the edges apart from the two in the cherry. We now show that $\pi(Q_n) = \widehat{Q}_{n-1}$, where $\widehat{P} = \mathrm{conv}\{0, P\}$ is a cone with the base given by P. The projection $\pi(Q_n)$ is described by all the triples of inequalities for all the inner nodes apart from the one incident with the cherry and the defining equation becomes an inequality

$$\sum_{e\in E_0\setminus\{e_1,e_2\}} x_e \leq 2.$$

Denote the edge incident with e_1, e_2 by e_3 and the related coordinates of x by $x_{e_1}, x_{e_2}, x_{e_3}$. The three inequalities involving x_{e_1} and x_{e_2} do not affect the projection since they imply that

$$\max\{x_{e_1} - x_{e_2}, x_{e_2} - x_{e_1}\} \leq x_{e_3} \leq x_{e_1} + x_{e_2}$$

and hence in particular if $x_{e_1} = x_{e_2}$ the constraint becomes $[0, 2x_{e_1}]$. Consequently the set given by $x_{e_1} + x_{e_2} - x_{e_3} \geq 0$, $x_{e_1} + x_{e_3} - x_{e_2} \geq 0$, $x_{e_2} + x_{e_3} - x_{e_1} \geq 0$ projects down to $\mathbb{R}_{\geq 0}$. However, since $\widehat{Q}_{n-1}$ is contained in the nonnegative orthant, there are no additional constraints on x_{e_3}. Inequalities in (22) define a polyhedral cone and the equation $\sum_{e \in E_0 \setminus \{e_1,e_2\}} x_e = t$ for $t \geq 0$ cuts out a bounded slice of the cone which is equal to $t \cdot P_{n-1}$. The sum of all these for $t \in [0,2]$ is exactly $\widehat{Q}_{n-1}$.

Since $\widehat{Q}_{n-1} = \widehat{P}_{n-1}$ by induction, then each $\pi(x)$ for $x \in Q_n$ is a convex combination of the points generating P_{n-1} and zero, that is $\pi(x) = \sum c_{ij} p_{ij}$ where the sum is over all $i \neq j \in \{a, 3, \ldots, n\}$ and $c_{ij} \geq 0$, $\sum c_{ij} \leq 1$. Next, we lift this combination back to Q_n, and show, that any such a lift has to lie in P_n. This would imply that in particular $x \in P_n$. Let y denote a lift of $\pi(x)$ to Q_n. We have

$$y = \sum c_{ij} r_{ij} + \left(1 - \sum c_{ij}\right) r_0,$$

where r_{ij} is a lift of $\pi(p_{ij})$ and r_0 is a lift of the origin. It suffices to show that each r_{ij} and r_0 necessarily lie in P_n.

Consider the following three cases. First, if $p_{ij} \in P_{n-1}$ is such that $x_{e_3} = 0$. Since $P_{n-1} = Q_{n-1}$ and Q_{n-1} satisfy the equation $\sum_{e \in E_0 \setminus \{e_1,e_2\}} x_e + x_{e_3} = 2$, sum of all the other coordinates related to the terminal edges of the smaller tree is 2. Hence, if we lift $\pi(p_{ij})$ to Q_n, then $x_{e_3} = 0$ and

$$x_{e_1} + x_{e_2} \geq 0, \quad x_{e_1} - x_{e_2} \geq 0, \quad x_{e_2} - x_{e_1} \geq 0$$

by plugging $x_{e_3} = 0$ into the three inequalities for the node a. But since $r_{ij} \in Q_n$ must also satisfy the equation $\sum_{e \in E_0} x_e = 2$, and, since we already have

$$\sum_{e \in E_0 \setminus \{e_1,e_2\}} x_e = 2,$$

then $x_{e_1} + x_{e_2} = 0$ and hence $x_{e_1} = x_{e_2} = 0$. Consequently, r_{ij} is a vertex of P_n corresponding to the path between i and j. Second, if p_{ij} is a vertex of P_{n-1} such that $x_{e_3} = 1$, then the sum of all the other coordinates of p_{ij} related to the terminal edges of the smaller tree is 1. Because the lift lies in Q_n we have $x_{e_1} + x_{e_2} = 1$. The additional inequalities give that $x_{e_1}, x_{e_2} \geq 0$. Hence in this case r_{ij} is a convex combination of two points in P_n corresponding to paths terminating in either of the nodes 1 or 2. Finally, we can easily check that zero lifts uniquely to a point in P_n corresponding to the path $E(12)$ joining the leaves 1 and 2. Indeed, from the equation defining Q_n we have $x_{e_1} + x_{e_2} = 2$ and from the inequalities since $x_{e_3} = 0$ we have $x_{e_1} = x_{e_2} = 1$. Therefore every lift y of $\pi(x)$ to Q_n can be written as a convex combination of points generating P_n and hence $y \in P_n$. Consequently $x \in P_n$ and hence $Q_n \subseteq P_n$. ■

Lemma 25 shows that P_n has an extremely simple structure. The inequalities give a polyhedral cone and the equation cuts out the polytope P_n as a slice of this cone. The result gives us also the representation of $\Gamma(Q_\delta)$ in terms of the defining equations and inequalities.

Proposition 26 (Structure of $\Gamma(Q_\delta)$) *Polytope $\Gamma(Q_\delta) \subset \mathcal{V}_\delta$ is given as an intersection of the sets defined by the inequalities in (22) together with $|\delta| + 1$ equations given by*

$$\begin{array}{ll} 2y_v = x_{v\mathrm{ch}_1(v)} + x_{v\mathrm{ch}_2(v)} - x_{\mathrm{pa}(v)v} & \text{for all } v \neq r \text{ such that } \delta_v = 1, \\ 2y_r = x_{r\mathrm{ch}_1(r)} + x_{r\mathrm{ch}_2(r)} + x_{r\mathrm{ch}_3(r)} & \text{if } \delta_r = 1, \text{ and} \\ \sum_{e \in E_0} x_e = 4. & \end{array} \tag{23}$$

From this we can partially understand the structure of $\Gamma_+(Q_\delta)$. First note that $\Gamma_+(f) = \Gamma(f) + \mathbb{R}^d_{\geq 0}$, where the plus denotes the Minkowski sum. The *Minkowski sum* of two polyhedra is by definition

$$\Gamma_1 + \Gamma_2 = \{x+y \in \mathbb{R}^d : x \in \Gamma_1, y \in \Gamma_2\}.$$

Lemma 27 *Let $\Gamma \subset \mathbb{R}^n_{\geq 0}$ be a polytope and let Γ_+ be the Minkowski sum of Γ and the standard cone $\mathbb{R}^n_{\geq 0}$. Then all the facets of Γ_+ are of the form $\sum_i a_i x_i \geq c$, where $a_i \geq 0$ and $c \geq 0$.*

Now we are ready to compute multiplicities of the real log-canonical threshold $\mathrm{RLCT}_0(Q_\delta)$ at least in certain cases. This completes the proof of Proposition 16.

Lemma 28 (Computing multiplicities) *Let T be a trivalent tree with $n \geq 4$ leaves, rooted in r. Let $q \in \mathcal{M}_T$ be such that $\hat{\kappa}_{ij} = 0$ for all $i, j \in [n]$ and $\omega_0 \in \widehat{\Omega}_0$. Let $\delta = \delta(\omega_0)$ be such that $\delta_v = 1$ if $s^0_v = 1$ and it is zero otherwise. Define $Q_\delta(\omega)$ as in (20). If either: (i) $\delta_r = 0$ or (ii) $\delta_r = 1$ and $\delta_v = 1$ for all $(r,v) \in E$ then $\mathrm{mult}_0(Q_\delta) = 1$.*

Proof A standard result for Minkowski sums says that each face of a Minkowski sum of two polyhedra can be decomposed as a sum of two faces of the summands and this decomposition is unique. Each facet of $\Gamma_+(Q_\delta)$ is decomposed as a face of the standard cone $\mathbb{R}^{n_e+|\delta|}_{\geq 0} \subset \mathcal{V}_\delta$ plus a face of $\Gamma(Q_\delta)$. We say that a face of $\Gamma(Q_\delta)$ induces a facet of $\Gamma_+(Q_\delta)$ if there exists a face of the standard cone $\mathbb{R}^{n_e+|\delta|}_{\geq 0}$ such that the Minkowski sum of these two faces gives a facet of $\Gamma_+(Q_\delta)$. Since the dimension $\Gamma(Q_\delta)$ is lower than the dimension of the resulting polyhedron it turns out that one face of $\Gamma(Q_\delta)$ can induce more than one facet of $\Gamma_+(Q_\delta)$. In particular $\Gamma(Q_\delta)$ itself induces more than one facet where one of them is F_0 given by (21).

Every facet of $\Gamma_+(Q_\delta)$ containing the point $\frac{4}{n}\mathbf{1}$, after normalizing the coefficients to sum to n, that is $\sum_v \alpha_v + \sum_e \beta_e = n$, is of the form

$$\sum_v \alpha_v y_v + \sum_e \beta_e x_e \geq 4, \tag{24}$$

where by Lemma 27 we can assume that $\alpha_v, \beta_e \geq 0$. Our approach can be summarized as follows. Using Construction 23 we provide coordinates of a point $p \in \Gamma(Q_\delta)$ such that $\frac{4}{n}\mathbf{1}$ lies on the boundary of $p + \mathbb{R}^{n_e+|\delta|}_{\geq 0}$. Then $\frac{4}{n}\mathbf{1}$ can only lie on faces of $\Gamma_+(Q_\delta)$ induced by faces of $\Gamma(Q_\delta)$ containing p. To show that the multiplicity is exactly 1 we need to show that $\frac{4}{n}\mathbf{1}$ lies in the interior of F_0.

First, assume that $\delta_r = 0$ which corresponds to the case when the root r represents a non-degenerate random variable. Consider the point $p \in \Gamma(Q_\delta)$ induced by the network of $2n$ paths given in the second part of Construction 23. Since $x_e = \frac{4}{n}$ for all $e \in E$ then from the description of $\Gamma(Q_\delta)$ in Lemma 26 we can check that all defining inequalities are strict for this point. Therefore p lies in the interior of $\Gamma(Q_\delta)$ and the only facets of $\Gamma_+(Q_\delta)$ containing p are these induced by $\Gamma(Q_\delta)$ itself. The equation defining a facet induced by $\Gamma(Q_\delta)$ has to be obtained as a combination of the defining equations: $\sum_{e \in E_0} x_e = 4$ and $|\delta|$ equations

$$2y_v - x_{v\mathrm{ch}_1(v)} - x_{v\mathrm{ch}_2(v)} + x_{\mathrm{pa}(v)v} = 0 \tag{25}$$

for all $v \in V$ such that $\delta_v = 1$. We check possible combinations such that the form of the induced inequality in (24) is attained. The first inequality, defining F_0, is already of this form (cf. (21)). The sum of all the coefficients is n since there are n terminal edges. Any other facet has to be

obtained by adding to the first equation (since the right hand side in (24) is 4) a non-negative (since the coefficients in front of y_v need to be non-negative) combination of equations in (25). However, since the sum of the coefficients in (25) is $+1$, this contradicts the assumption that the sum of coefficients in the defining inequality is n. Consequently, if $\delta_r = 0$ the codimension of the face hit by $\frac{4}{n}\mathbf{1}$ is 1 and hence by Theorem 20 we have that $\mathrm{mult}_0(Q_\delta) = 1$.

Second, if $\delta_r = 1$ and $\delta_v = 1$ for all children of r in T then since all the nodes adjacent to r (denote them by a,b,c) are inner we have three different ways of conducting the construction of the n-path network in Construction 23 (by omitting each of the incident edges). Hence we get three different points and their barycenter satisfies $x_{ra} = x_{rb} = x_{rc} = \frac{8}{3n}$ and $x_e = \frac{4}{n}$ for all the other edges; $y_r = \frac{4}{n}$, $y_a = y_b = y_c = \frac{8}{3n}$ and $y_v = \frac{2}{n}$ for all the other inner nodes. Denote this point by p and note that $p \leq \frac{4}{n}\mathbf{1}$. By the facet description of $\Gamma(Q_\delta)$ derived in Proposition 26 we can check that this point cannot lie in any of the facets defining $\Gamma(Q_\delta)$ and hence it is an interior point of the polytope. As in the first case it means that the facets of $\Gamma_+(Q_\delta)$ containing p are induced by $\Gamma(Q_\delta)$. By Proposition 26 the affine span is given by (23). Since the sum of coefficients in the equation involving y_r is negative we cannot use the same argument as in the first case. Instead, we add to $\sum_{e\in E_0} x_e = 4$ a non-negative combination of equations in (25) each with coefficient $t_v \geq 0$ and then add the equation in (23) involving y_r with coefficient $\sum_{v\neq r} t_v$. The sum of coefficients in the resulting equation will be n by construction. The coefficient of x_{ra} is $t_a - \sum_{v\neq r} t_v = -\sum_{v\neq r,a} t_v$. Since it has to be non-negative it follows that $t_v = 0$ for all v apart from a. However, by checking the coefficient of x_{rb} one deduces that $t_v = 0$ for all inner nodes v. Consequently the only possible facet of $\Gamma_+(Q_\delta)$ containing $\frac{4}{n}\mathbf{1}$ is F_0 and hence again $\mathrm{mult}_0(Q_\delta) = 1$. ■

The following example shows that in certain cases $\mathrm{mult}_0(Q_\delta)$ can be strictly greater than 1.

Example 5 *Consider the quartet tree model with q such that $\hat{\kappa}_{ij} = 0$ for all $i,j = 1,2,3,4$. In this case $\Gamma(Q_\delta) \subseteq \mathbb{R}^7$ has six vertices $(2,0;2,2,0,0,0)$, $(2,0;2,0,2,2,0)$, $(2,0;2,0,2,0,2)$, $(2,0;2,0,2,2,0)$, $(2,0;0,2,2,0,2)$ and $(0,2;0,0,0,2,2)$. The facet description of the Newton polyhedron $\Gamma_+(Q_\delta)$ can be easily computed using* POLYMAKE *Gawrilow and Joswig (2005). From this description it is easily checked that the point $(1,1;1,1,1,1,1)$ lies on two facets of $\Gamma_+(Q_\delta)$. It follows that the codimension of the face hit by this vector is two, or equivalently,* $\mathrm{mult}_0(Q_\delta) = 2$.

7. Proof of Theorem 2

In this section we complete the proof of Theorem 2 using results from the previous sections. We split it into three steps.

7.1 Step 1

To analyze the asymptotic behavior of the stochastic complexity F_N, by Theorem 4, equivalently we can compute $\mathrm{RLCT}_{\Theta_T}(K;\varphi)$, where K is the Kullback-Leibler distance defined in (3) and φ is the prior distribution satisfying (A1). By Theorem 11 and Theorem 14 this real log-canonical threshold is equal to $\mathrm{RLCT}_{\Omega_T}(\mathscr{I})$, where $\mathscr{I}$ is the ideal defined by (11).

7.2 Step 2

We compute separately $\mathrm{RLCT}_{\Omega_T}(\mathscr{I})$ in the case when $n=3$. If T is rooted in the inner node the expansion for $\mathbb{E}F_N$ follows from Theorem 4 in Rusakov and Geiger (2005). Thus if $\widehat{E}=E$, which in Rusakov and Geiger (2005) corresponds to the type 2 singularity, then

$$\mathbb{E}F_N = NS + 2\log N + O(1) \quad \text{or} \qquad \mathrm{RLCT}_{\Omega_T}(\mathscr{I}) = (2,1). \tag{26}$$

Since all the neighbours of the root are leaves and hence, by (A2), they are non-degenerate we need only to make sure that the first equation in Theorem 2 gives (26). This follows from the fact that $l_2=0$ and $l_0=1$, where l_i $i=0,1,2,3$ defined in the introduction is the number of *inner* nodes of T whose degree in $\widehat{T}$ is i. In the case when $|\widehat{E}|=1$ (type 1 singularity) we have

$$\mathbb{E}F_N = NS + \frac{5}{2}\log N + O(1) \quad \text{or} \qquad \mathrm{RLCT}_{\Omega_T}(\mathscr{I}) = \left(\frac{5}{2},1\right).$$

The second equation in Theorem 2 holds since $l_2=1$, $l_0=0$ and $c=0$. If $\widehat{E}=\emptyset$ we have

$$\mathbb{E}F_N = NS + \frac{7}{2}\log N + O(1) \quad \text{or} \qquad \mathrm{RLCT}_{\Omega_T}(\mathscr{I}) = \left(\frac{7}{2},1\right),$$

which again is true since $l_2=0$, $l_0=0$ and $c=0$.

Now assume that T is rooted in a leaf, say 1. If there exists $i,j=1,2,3$ such that $\hat{\kappa}_{ij}\neq 0$ (or equivalently $|\widehat{E}|\leq 1$) then $\widehat{V}=\emptyset$ and by Proposition 8

$$\mathbb{E}F_N = NS + \frac{7-2l_2}{2} + O(1) \quad \text{or} \qquad \mathrm{RLCT}_{\Omega_T}(\mathscr{I}) = \left(\frac{7-2l_2}{2},1\right).$$

If $\widehat{E}=E$ then $\widehat{V}\neq\emptyset$ and by Theorem 14 for every $\omega_0\in\widehat{\Omega}_T$

$$\mathrm{RLCT}_{\omega_0}(\mathscr{I}) = \left(\frac{3}{2},0\right) + \mathrm{RLCT}_{\omega_0}(\mathscr{J}).$$

Moreover, by Lemma 18, for every $\omega_0\in\widehat{\Omega}_0$

$$\mathrm{RLCT}_{\omega_0}(\mathscr{J}) = \mathrm{RLCT}_0(\langle \eta_{1h}\eta_{h2}, \eta_{1h}\eta_{h3}, s_h^{\delta_h}\eta_{h2}\eta_{h3}\rangle),$$

where $\delta_h=1$ if $s_h^0=1$ and $\delta_h=1$ otherwise. It can be checked directly by using the Newton diagram method and Theorem 20 that $\mathrm{RLCT}_{\omega_0}(\mathscr{J}) = (\frac{3}{4},1)$ both if $\delta_h=0$ and $\delta_h=1$ and hence $\mathrm{RLCT}_{\omega_0}(\mathscr{I}) = (\frac{9}{4},1)$. Since the points in $\widehat{\Omega}_0$ such that $s_h^0\neq 1$ lie in the interior of Ω_T then for these points $\mathrm{RLCT}_{\omega_0}(\mathscr{I}) = \mathrm{RLCT}_{\Omega_0}(\mathscr{I})$ where Ω_0 is a neighborhood of ω_0 in Ω_T. Hence, by (6), we have that

$$\mathrm{RLCT}_{\Omega_T}(\mathscr{I}) = \min_{\omega_0\in\widehat{\Omega}_T} \mathrm{RLCT}_{\Omega_0}(\mathscr{I}) \leq \min_{\omega_0\in\widehat{\Omega}_0} \mathrm{RLCT}_{\Omega_0}(\mathscr{I}) = \left(\frac{9}{4},1\right).$$

On the other hand, by (5) and then Proposition 17, we obtain the following inequalities

$$\mathrm{RLCT}_{\Omega_T}(\mathscr{I}) \geq \min_{\omega_0\in\widehat{\Omega}_T} \mathrm{RLCT}_{\omega_0}(\mathscr{I}) \geq \min_{\omega_0\in\widehat{\Omega}_{\mathrm{deep}}} \mathrm{RLCT}_{\omega_0}(\mathscr{I}) = \left(\frac{9}{4},1\right).$$

It follows that

$$\mathbb{E}F_N = NS + \frac{9}{4}\log N + O(1) \quad \text{or} \qquad \mathrm{RLCT}_{\Omega_T}(\mathscr{I}) = \left(\frac{9}{4},1\right),$$

which gives the the second equation in Theorem 2 since in this case $l_2=c=0$ and $l_0=1$.

7.3 Step 3, Case 1

Assume now that $n \geq 4$ and $r \notin \widehat{V}$. In this case, using notation from Section 5, every T_i for $i = 1, \ldots, k$ is rooted in one of its leaves. Hence $\mathrm{RLCT}_{\omega_0}(\mathscr{J}(T_i)) = (\frac{|L_i|}{4}, 1)$ for every $i = 1, \ldots k$. If $|L_i| \neq 3$ this follows from Proposition 16. If $|L_i| = 3$ it follows from Case 2 above. By Lemma 15 and Equation (17), for every $\omega_0 \in \widehat{\Omega}_0$ we have that

$$\mathrm{rlct}_{\omega_0}(\mathscr{I}) = \frac{n}{2} + \sum_{i=1}^{m} \frac{n_v^i + n_e^i - n_i - 2l_2^i}{2} + \sum_{i=1}^{k} \frac{|L_i|}{4},$$

where n_v^i, n_e^i, l_2^i are respectively the number of vertices, edges and and degree two nodes in $\widehat{T}$ of S_i; and L_i is the set of leaves of T_i. Let m_i denote the number of nodes of $\widehat{T}$ whose degree is i. Note that $m_2 = l_2$ but m_0 does not necessarily equal l_0. We now use three simple formulas: $\sum_i n_v^i = m_1 + m_2 + m_3$ (that is only degree zero nodes of $\widehat{T}$ do not lie in the S_i's), $\sum_i n_e^i = |E \setminus \widehat{E}|$ (that is $E \setminus \widehat{E}$ is the set of all edges of all the S_i's) and $\sum_i |L_i| = m_2 + n - m_1$ (that is the leaves of all the T_i's are precisely the degree two nodes of $\widehat{T}$ and these leaves of T which have degree zero in $\widehat{T}$). Moreover, for any graph with the vertex set V and the edge set E, $\sum_{v \in V} \deg(v) = 2n_e$ (see Semple and Steel, 2003, Corollary 1.2.2). Therefore, with the formula applied for the forest $\widehat{T}$, we have $m_1 + 2m_2 + 3m_3 = 2|E \setminus \widehat{E}|$. Using these four formulas together we show that $\mathrm{rlct}_{\omega_0}(\mathscr{I}) = \frac{1}{4}(3n + m_2 + 5m_3)$. The final formula for the coefficient follows from the fact that $l_2 = m_2$ and $l_0 = n_v - n - m_2 - m_3$. Moreover, since $\delta_r = 0$ for all $\omega_0 \in \widehat{\Omega}_0$ then, by Lemma 28, $\mathrm{mult}_0(\mathscr{J}(T_i)) = 1$ for every $\omega_0 \in \widehat{\Omega}_0$. Therefore,

$$\mathrm{RLCT}_{\omega_0}(\mathscr{I}) = \left(\frac{n_v + n_e - 2l_2}{2} - \frac{5l_0}{4}, 1 \right). \tag{27}$$

Now we show that $\mathrm{RLCT}_{\Omega_T}(\mathscr{I})$ also has the same form. Let ω_2 be a point in $\widehat{\Omega}_0$ such that $s_v \neq 1$ for all $v \in V$ and let $\omega_1 \in \widehat{\Omega}_{\mathrm{deep}}$. Equation (27) is true both if $\omega_0 = \omega_1$ and $\omega_0 = \omega_2$ and hence $\mathrm{RLCT}_{\omega_1}(\mathscr{I}) = \mathrm{RLCT}_{\omega_2}(\mathscr{I})$. However, since ω_2 is an inner point of Ω_T, it follows from the definition of $\mathrm{RLCT}_{\Omega_T}(\mathscr{I})$ as the minimum over all points in Ω_T, that

$$\mathrm{RLCT}_{\Omega_T}(\mathscr{I}) \leq \mathrm{RLCT}_0(\mathscr{I}_{\omega_2}).$$

On the other hand by (5) and Proposition 17

$$\mathrm{RLCT}_0(\mathscr{I}_{\omega_1}) = \min_{\omega_0 \in \widehat{\Omega}_T} \mathrm{RLCT}_0(\mathscr{I}_{\omega_0}) \leq \min_{\omega_0 \in \widehat{\Omega}_T} \mathrm{RLCT}_{\Omega_0}(\mathscr{I}_{\omega_0}) = \mathrm{RLCT}_{\Omega_T}(\mathscr{I}).$$

Therefore, if $r \notin \widehat{V}$, then in fact $\mathrm{RLCT}_{\Omega_T}(\mathscr{I}) = (\lambda, 1)$, where λ is the coefficient in (27), and

$$\mathbb{E}F_N = NS + \lambda \log N + O(1).$$

The main formula in Theorem 2 is proved in this case because $c = 0$.

7.4 Step 3, Case 2

Let now $n \geq 4$ and $r \in \widehat{V}$. Let $1 \leq j \leq k$ be such that r is an inner node of T_j and $\omega_0 \in \widehat{\Omega}_0$. For all $i \neq j$, T_i is rooted in one of its leaves. Therefore, by Lemma 22, Lemma 28 and Step 2 above

for all $i \neq j$ we have that $\mathrm{RLCT}_{\omega_0}(\mathscr{J}(T_i)) = (|L_i|/4, 1)$. It remains to compute $\mathrm{RLCT}_{\omega_0}(\mathscr{J}(T_j))$. If $|L_j| = 3$ then $\mathrm{RLCT}_{\omega_0}(\mathscr{J}(T_j)) = (1/2, 1) = ((|L_j|-1)/4, 1)$ by the Step 2 above, (cf. (26)). In this case the computations are the same as in Step 3, Case 1 but with a difference of $\frac{1}{4}$ in the real log-canonical threshold. We obtain

$$\mathbb{E}F_N = NS + \left(\frac{n_v + n_e - 2l_2}{2} - \frac{5l_0+1}{4}\right)\log N + O(1).$$

However, if $|L_j| \geq 4$ then, by Lemma 22, $\mathrm{rlct}_0(\mathscr{J}(T_j)) = |L_j|/4$ and hence as in Step 3, Case 1 we have $\sum_{i=1}^{k} \mathrm{rlct}_0(\mathscr{J}_{\omega_0}(T_i)) = \frac{1}{4}(n - m_1 + m_2)$. Therefore $\mathrm{rlct}_{\Omega_T}(\mathscr{I}) = \lambda$. We compute the multiplicity by considering different subcases. If all the neighbours of r are degenerate then for all points $\omega_0 \in \widehat{\Omega}_{\mathrm{deep}}$ we have that $\delta_r = 1$ and $\delta_v = 1$ for all neighbours v or r. It follows from Lemma 28 that $\mathrm{mult}_{\omega_0}(\mathscr{J}(T_j)) = 1$ and hence $\mathrm{mult}_{\Omega_T}(\mathscr{I}) = 1$. Therefore,

$$\mathbb{E}F_N = NS + \frac{1}{4}(3n + l_2 + 5l_3)\log N + O(1).$$

Otherwise we do not have explicit bounds on the multiplicity. Since $\mathrm{mult}_{\Omega_T}(\mathscr{I}) \geq 1$ then

$$\mathbb{E}F_N = NS + \frac{1}{4}(3n + l_2 + 5l_3)\log N - (m-1)\log\log N + O(1),$$

where $m \geq 1$. This finishes the proof of Theorem 2. □

Remark 29 *Example 5 showed that* $\mathrm{mult}_{\omega_0}(\mathscr{I})$ *may be strictly greater than* 1 *for some boundary points of* Ω_T. *The analysis of how it affects the computation of* $\mathrm{mult}_{\Omega_T}(\mathscr{I})$ *is highly complicated as it involves resolution of the boundary constraints. Typically we are just able to provide upper bounds. For example, since in Example 5 we have* $\mathrm{mult}_{\Omega_0}(\mathscr{I}) = \mathrm{mult}_{\omega_0}(\mathscr{I})$ *then, by (5),* $\mathrm{mult}_{\Omega_0}(\mathscr{I}) \leq \mathrm{mult}_{\omega_0}(\mathscr{I}) = 2$.

Acknowledgments

I am especially grateful to Shaowei Lin for a number of illuminating discussions and introducing me to the theory of log-canonical thresholds. I also want to thank Diane Maclagan and the referees for helpful comments.

References

Vladimir I. Arnold, Sabir M. Guseĭn-Zade, and Aleksandr N. Varchenko. *Singularities of Differentiable Maps*, volume II. Birkhäuser, 1988.

Edward Bierstone and Pierre D. Milman. Semianalytic and subanalytic sets. *Publications Mathématiques de l'IHÉS*, 67(1):5–42, 1988.

David Maxwell Chickering and David Heckerman. Efficient approximations for the marginal likelihood of Bayesian networks with hidden variables. *Machine Learning*, 29(2):181–212, 1997.

Gregory F. Cooper and Edward Herskovits. A Bayesian method for the induction of probabilistic networks from data. *Machine Learning*, 9(4):309–347, 1992.

Ewgenij Gawrilow and Michael Joswig. Geometric reasoning with polymake. `arXiv:math.CO/0507273`, 2005.

Mark Goresky and Robert MacPherson. *Stratified Morse Theory*, volume 14 of *Ergebnisse der Mathematik und ihrer Grenzgebiete (3) [Results in Mathematics and Related Areas (3)]*. Springer-Verlag, Berlin, 1988. ISBN 3-540-17300-5.

Dominique Haughton. On the choice of a model to fit data from an exponential family. *Ann. Statist*, 16(1):342–355, 1988.

David Heckerman, Dan Geiger, and David Maxwell Chickering. Learning Bayesian networks: The combination of knowledge and statistical data. *Machine Learning*, 20(3):197–243, 1995.

Robert Lazarsfeld. *Positivity in Algebraic Geometry*. A Series of Modern Surveys in Mathematics. Springer Verlag, 2004.

Shaowei Lin. Asymptotic Approximation of Marginal Likelihood Integrals. `arXiv:1003.5338`, November 2011. submitted.

Radu Mihaescu and Lior Pachter. Combinatorics of least-squares trees. *Proceedings of the National Academy of Sciences of the United States of America*, 105(36):13206, 2008.

Judea Pearl and Michael Tarsi. Structuring causal trees. *J. Complexity*, 2(1):60–77, 1986. ISSN 0885-064X. Complexity of approximately solved problems (Morningside Heights, N.Y., 1985).

Dmitry Rusakov and Dan Geiger. Asymptotic model selection for naive Bayesian networks. *J. Mach. Learn. Res.*, 6:1–35 (electronic), 2005. ISSN 1532-4435.

Morihiko Saito. On real log canonical thresholds. `arXiv:0707.2308`, 2007.

Gideon Schwarz. Estimating the dimension of a model. *Annals of Statistics*, 6(2):461–464, 1978.

Charles Semple and Mike Steel. *Phylogenetics*, volume 24 of *Oxford Lecture Series in Mathematics and its Applications*. Oxford University Press, Oxford, 2003. ISBN 0-19-850942-1.

Sumio Watanabe. *Algebraic Geometry and Statistical Learning Theory*. Number 25 in Cambridge Monographs on Applied and Computational Mathematics. Cambridge University Press, 2009. ISBN-13: 9780521864671.

Keisuke Yamazaki and Sumio Watanabe. Newton diagram and stochastic complexity in mixture of binomial distributions. In *Algorithmic Learning Theory*, volume 3244 of *Lecture Notes in Comput. Sci.*, pages 350–364. Springer, Berlin, 2004.

Piotr Zwiernik and Jim Q. Smith. Implicit inequality constraints in a binary tree model. *Electron. J. Statist.*, 5:1276–1312, 2011a. ISSN 1935-7524. doi: 10.1214/11-EJS640.

Piotr Zwiernik and Jim Q. Smith. Tree-cumulants and the geometry of binary tree models. *to appear in Bernoulli*, 2011b.

Journal of Machine Learning Research 12 (2011) 3311-3370 Submitted 7/09; Revised 1/11; Published 11/11

Semi-Supervised Learning with Measure Propagation

Amarnag Subramanya ASUBRAM@EE.WASHINGTON.EDU
Jeff Bilmes BILMES@EE.WASHINGTON.EDU
Department of Electrical Engineering
University of Washington
Seattle, WA 98195, USA

Editor: Yoshua Bengio

Abstract

We describe a new objective for graph-based semi-supervised learning based on minimizing the Kullback-Leibler divergence between discrete probability measures that encode class membership probabilities. We show how the proposed objective can be efficiently optimized using alternating minimization. We prove that the alternating minimization procedure converges to the correct optimum and derive a simple test for convergence. In addition, we show how this approach can be scaled to solve the semi-supervised learning problem on very large data sets, for example, in one instance we use a data set with over 10^8 samples. In this context, we propose a graph node ordering algorithm that is also applicable to other graph-based semi-supervised learning approaches. We compare the proposed approach against other standard semi-supervised learning algorithms on the semi-supervised learning benchmark data sets (Chapelle et al., 2007), and other real-world tasks such as text classification on Reuters and WebKB, speech phone classification on TIMIT and Switchboard, and linguistic dialog-act tagging on Dihana and Switchboard. In each case, the proposed approach outperforms the state-of-the-art. Lastly, we show that our objective can be generalized into a form that includes the standard squared-error loss, and we prove a geometric rate of convergence in that case.

Keywords: graph-based semi-supervised learning, transductive inference, large-scale semi-supervised learning, non-parametric models

1. Introduction

In many applications, annotating training data is time-consuming, costly, tedious, and error-prone. For example, training an accurate speech recognizer requires large amounts of well annotated speech data (Evermann et al., 2005). In the case of document classification for Internet search, it is not feasible to accurately annotate sufficient number of web-pages for all categories of interest. The process of training classifiers with small amounts of labeled data and relatively large amounts of unlabeled data is known as semi-supervised learning (SSL). SSL lends itself as a useful technique in many machine learning applications as one only needs to annotate small amounts of data for training models.

While SSL may be used to solve a variety of learning problems, such as clustering and regression, in this paper we address only the semi-supervised classification problem—henceforth, SSL will refer to semi-supervised classification. Examples of SSL algorithms include self-training (Scudder, 1965) and co-training (Blum and Mitchell, 1998). A thorough survey of SSL algorithms is given in Seeger (2000), Zhu (2005b), Chapelle et al. (2007) and Blitzer and Zhu (2008). SSL

is also related to the problem of *transductive learning* (Vladimir, 1998). In general, a learner is transductive if it is designed only for a closed data set, where the test set is revealed at training time. In practice, however, transductive learners can be modified to handle unseen data (Sindhwani et al., 2005; Zhu, 2005b). Chapelle et al. (2007, Chapter 25) gives a nice discussion on the relationship between SSL and transductive learning.

Graph-based SSL algorithms are an important sub-class of SSL techniques that have received much attention in the recent past (Zhu, 2005b; Chapelle et al., 2007). Here one assumes that the data (both labeled and unlabeled) is embedded within a low-dimensional manifold that may be reasonably expressed by a graph. Each data sample is represented by a vertex in a weighted graph with the weights providing a measure of similarity between vertices. Most graph-based SSL algorithms fall under one of two categories – those that use the graph structure to spread labels from labeled to unlabeled samples (Szummer and Jaakkola, 2001; Zhu and Ghahramani, 2002a) and those that optimize a loss function based on smoothness constraints derived from the graph (Blum and Chawla, 2001; Zhu et al., 2003; Joachims, 2003; Belkin et al., 2005; Corduneanu and Jaakkola, 2003; Tsuda, 2005). In some cases, for example, label propagation (Zhu and Ghahramani, 2002a) and the harmonic functions algorithm (Zhu et al., 2003; Bengio et al., 2007), it can be shown that the two categories optimize a similar loss function (Zhu, 2005a; Bengio et al., 2007).

A large number of graph-based SSL algorithms attempt to minimize a loss function that is inherently based on squared-loss (Zhu et al., 2003; Bengio et al., 2007; Joachims, 2003). While squared-loss is optimal under a Gaussian noise model, it is not optimal in the case of classification problems. Another potential drawback in the case of some graph-based SSL algorithms (Blum and Chawla, 2001; Joachims, 2003) is that they assume binary classification tasks and thus require the use of sub-optimal (and often computationally expensive) approaches such as one vs. rest to solve multi-class problems. While it is often argued that the use of binary classifiers within a one vs. rest framework performs as well as true multi-class solutions (Rifkin and Klautau, 2004), our results on SSL problems suggest otherwise (see Section 7.2.2).

Further, there is a lack of principled approaches to incorporate label priors in graph-based SSL algorithms. Approaches such as *class mass normalization* (CMN) and *label bidding* are used as a post-processing step rather than being tightly integrated with the inference (Zhu and Ghahramani, 2002a). In this context, it is important to distinguish label priors from balance priors. Balance priors are used in some algorithms such as Joachims (2003) and discourage the scenario where all the unlabeled samples are classified as belonging to a single class (i.e., a degenerate solution). Balance priors impose selective pressure collectively on the entire set of resulting answers. Label priors, on the other hand, select the more desirable configuration for each answer individually without caring about properties of the overall set of resulting answers. In addition, many SSL algorithms, such as Joachims (2003) and Belkin et al. (2005), are unable to handle *label uncertainty*, where there may be insufficient evidence to justify only a single label for a labeled sample.

Another area for improvement over previous work in graph-based SSL (and SSL in general) is the lack of algorithms that scale to very large data sets. SSL is based on the premise that unlabeled data is easily obtained, and adding large quantities of unlabeled data leads to improved performance. Thus practical scalability (e.g., parallelization), is important to apply SSL algorithms on large real-world data sets. Collobert et al. (2006) and Sindhwani and Keerthi (2006) discuss the application of TSVMs to large-scale problems. Delalleau et al. (2005) suggests an algorithm for improving the induction speed in the case of graph-based algorithms. Karlen et al. (2008) solve a graph transduction problem with 650,000 samples. To the best of our knowledge, the largest graph-based problem

solved to date had about 900,000 samples (includes both labeled and unlabeled data) (Tsang and Kwok, 2006). Clearly, this is a fraction of the amount of unlabeled data at our disposal. For example, on the Internet alone, we create 1.6 billion blog posts, 60 billion emails, 2 million photos and 200,000 videos every day (Tomkins, 2008). In general, graph-based SSL algorithms that use matrix inversion (Zhu et al., 2003; Belkin et al., 2005) or eigen-based matrix decomposition (Joachims, 2003) do not scale very easily.

In Subramanya and Bilmes (2008), we proposed a new framework for graph-based SSL that involves optimizing a loss function based on Kullback-Leibler divergence (KLD) between probability measures defined for each graph vertex. These probability measures encode the class membership probabilities. The advantages of this new convex objective are: (a) it is naturally amenable to multi-class (> 2) problems; (b) it can handle label uncertainty; and (c) it can integrate priors. Furthermore, the use of probability measures allows the exploitation of other well-defined functions of measures, such as entropy, to improve system performance. Subramanya and Bilmes (2008) also showed how the proposed objective can be optimized using alternating minimization (AM) (Csiszar and Tusnady, 1984) leading to simple update equations. This new approach to graph-based SSL was shown to outperform other state-of-the-art SSL algorithms for the document and web page classification tasks. In this paper we extend the above work along the following lines –

1. We prove that AM on the proposed convex objective for graph-based SSL converges to the global optima. In addition we derive a test for convergence that does not require the computation of the objective.

2. We compare the performance of the proposed approach against other state-of-the-art SSL approaches, such as manifold regularization (Belkin et al., 2005), label propagation (Zhu and Ghahramani, 2002a), and spectral graph transduction (Joachims, 2003) on a variety of tasks ranging from synthetic data sets to SSL benchmark data sets (Chapelle et al., 2007) to real-world problems such as phone classification, text classification, web-page classification and dialog-act tagging.

3. We propose a graph node ordering algorithm that is cache cognizant and makes obtaining a linear speedup with a parallel symmetric multi-processor (SMP) implementation more likely. As a result, the algorithms are able to scale to very large data sets. The node ordering algorithm is quite general and can be applied to graph-based SSL algorithms such as Zhu and Ghahramani (2002a); Zhu et al. (2003). In one instance, we solve a SSL problem over a graph with 120 million vertices (which is quite a bit more than the previous largest size of 900,000 vertices). A useful byproduct of this experiment is the *semi-supervised switchboard transcription project* (S3TP) which consists of phone level annotations of the *Switchboard-I* corpus generated in a semi-supervised manner (see Section 8.1, Subramanya and Bilmes, 2009).

4. We propose a graph-based SSL objective using Bregman divergence in Section 9.1. This objective generalizes previously proposed approaches such as label propagation (Zhu and Ghahramani, 2002a), the harmonic functions algorithm (Zhu et al., 2003), the quadratic cost criterion (Bengio et al., 2007) and our proposed approach. This objective can potentially be optimized using AM which portends well for solving general learning problems over objects for which a Bregman divergence can be defined (Tsuda et al., 2005).

5. A specific case of the Bregman divergence form is the standard squared-loss based objective, and we prove a geometric rate of convergence in this case in Appendix F

6. We discuss a principled approach to integrating label priors into the proposed objective (see Section 9.2).

7. We also show how our proposed objective can be extended to directed graphs (see Section 9.3).

2. Graph Construction

Let $\mathcal{D}_l = \{(\mathbf{x}_i, r_i)\}_{i=1}^{l}$ be the set of labeled samples, $\mathcal{D}_u = \{\mathbf{x}_i\}_{i=l+1}^{l+u}$ the set of unlabeled samples and $\mathcal{D} \triangleq \{\mathcal{D}_l, \mathcal{D}_u\}$. Here r_i is an encoding of the labeled data and will be explained shortly. We are interested in solving the transductive learning problem, that is, given $\mathcal{D}$, the task is to predict the labels of the samples in $\mathcal{D}_u$ (for inductive see Section 7.4). We are given an undirected weighted graph $\mathcal{G} = (V, E)$, where the vertices (nodes) $V = \{1, \ldots, m\}$ ($m \triangleq l + u$) are the data points in $\mathcal{D}$ and the edges $E \subseteq V \times V$. Let $V = V_l \cup V_u$ where V_l is the set of labeled vertices and V_u the set of unlabeled vertices. $\mathcal{G}$ may be represented via a matrix $\mathbf{W}$ referred to as the weight or affinity matrix.

There are many ways of constructing the graph. In some applications, it might be a natural result of relationship between the samples in $\mathcal{D}$, for example, consider the case where each vertex represents a web-page and the edges represent the links between web-pages. In other cases, such as the work of Fei and Changshui (2006), the graph is generated by performing an operation similar to local linear embedding (LLE) but constraining the LLE weights to be non-negative. In a majority of the applications, including those considered in this paper, we use k-nearest neighbor (NN) graphs. In our case here, we make use of symmetric k-NN graphs and so the edge weight $w_{ij} = [\mathbf{W}]_{ij}$ is given by

$$w_{ij} = \begin{cases} \text{sim}(\mathbf{x}_i, \mathbf{x}_j) & \text{if } j \in \mathcal{K}(i) \text{ or } i \in \mathcal{K}(j) \\ 0 & \text{otherwise} \end{cases}$$

where $\mathcal{K}(i)$ is the set of k-NN of $\mathbf{x}_i$ ($|\mathcal{K}(i)| = \text{k}, \ \forall i$) and $\text{sim}(\mathbf{x}_i, \mathbf{x}_j)$ is a measure of similarity between $\mathbf{x}_i$ and $\mathbf{x}_j$ (which are represented by nodes i and j). It is assumed that the similarity measure is symmetric, that is, $\text{sim}(x, y) = \text{sim}(y, x)$. Further $\text{sim}(x, y) \geq 0$. Some popular similarity measures include

$$\text{sim}(\mathbf{x}_i, \mathbf{x}_j) = e^{-\frac{\|\mathbf{x}_i - \mathbf{x}_j\|_2^2}{2\sigma}} \text{ or } \text{sim}(\mathbf{x}_i, \mathbf{x}_j) = \cos(\mathbf{x}_i, \mathbf{x}_j) = \frac{\langle \mathbf{x}_i, \mathbf{x}_j \rangle}{\| \mathbf{x}_i \|_2 \| \mathbf{x}_j \|_2}$$

where $\| \mathbf{x}_i \|_2$ is the ℓ_2 norm, and $\langle \mathbf{x}_i, \mathbf{x}_j \rangle$ is the inner product of $\mathbf{x}_i$ and $\mathbf{x}_j$. The first similarity measure is a radial-basis function (RBF) kernel of width σ applied to the squared Euclidean distance while the second is cosine similarity. Choosing the correct similarity measure and k are crucial steps in the success of any graph-based SSL algorithm as it determines the graph. At this point, graph construction "is more of an art, than science" (Zhu, 2005a) and is an active research area (Alexandrescu and Kirchhoff, 2007b). The choice of $\mathbf{W}$ depends on a number of factors such as, whether $\mathbf{x}_i$ is continuous or discrete and characteristics of the problem at hand. We discuss more about the choice of $\mathbf{W}$ in the context of the appropriate problem in Section 7.

3. Proposed Approach for Graph-based Semi-Supervised Learning

For each $i \in V$ and $j \in V_l$, we define discrete probability measures p_i and r_j respectively over the measurable space $(\mathrm{Y}, \mathcal{Y})$. That is, for each vertex in the graph, we define a measure p_i and for all the labeled vertices, in addition to the p's we also define r_i (recall, $\mathcal{D}_l = \{(\mathbf{x}_i, r_i)\}_{i=1}^{l}$). Here $\mathcal{Y}$ is the σ-field of measurable subsets of Y and $\mathrm{Y} \subset \mathbb{N}$ (the set of natural numbers) is the discrete space of classifier outputs. Thus $|\mathrm{Y}| = 2$ yields binary classification while $|\mathrm{Y}| > 2$ yields multi-class. As we only consider classification problems here, p_i and r_i are in essence multinomial distributions and so $p_i(y)$ represents the probability that the sample represented by vertex i belongs to class y. We assume that there is at least one labeled sample for every class. Note that the objective we propose is actually more general and can be easily extended to other learning problems such as regression.

The $\{r_i\}_i$'s represent the labels of the supervised portion of the training data and are derived in one of the following ways: (a) if $\hat{y}_i$ is the single supervised label for input $\mathbf{x}_i$ then $r_i(y) = \delta(y = \hat{y}_i)$, which means that r_i gives unity probability for y equaling the label $\hat{y}_i$; (b) if $\hat{y}_i = \{\hat{y}_i^{(1)}, \ldots, \hat{y}_i^{(t)}\}$, $t \leq |\mathrm{Y}|$ is a *set* of possible outputs for input $\mathbf{x}_i$, meaning an object validly falls into all of the corresponding categories, we set $r_i(y) = (1/k)\delta(y \in \hat{y}_i)$ meaning that r_i is uniform over only the possible categories and zero otherwise; (c) if the labels are somehow provided in the form of a set of non-negative scores, or even a probability distribution itself, we just set r_i to be equal to those scores (possibly) normalized to become a valid probability distribution. As can be seen, the r_i's can handle a wide variety of inputs ranging from the most certain case where a single input yields a single output to cases where there is an *uncertainty* associated with the output for a given input. It is important to distinguish between the classical multi-label problem and the use of uncertainty in r_j. In our case, if there are two non-zero outputs during training as in $r_j(y_1), r_j(y_2) > 0, y_1, y_2 \in \mathrm{Y}$, it does not imply that the input $\mathbf{x}_j$ necessarily possesses the properties of the two corresponding classes. Rather, this means that there is uncertainty regarding truth, and we use a discrete probability measure over the labels to represent this uncertainty.

As p_i and r_i are discrete probability measures, we have that $\sum_y p_i(y) = 1$, $p_i(y) \geq 0$, $\sum_y r_i(y) = 1$, and $r_i(y) \geq 0$. In other words, p_i and r_i lie within a $|\mathrm{Y}|$-dimensional probability simplex which we represent using $\triangle_{|\mathrm{Y}|}$ and so $p_i, r_i \in \triangle_{|\mathrm{Y}|}$ (henceforth denoted as $\triangle$). Also $\mathrm{p} \triangleq (p_1, \ldots, p_m) \in \triangle^m$ denotes the set of measures to be learned, and $\mathrm{r} \triangleq (r_1, \ldots, r_l) \in \triangle^l$ are the set of measures that are given. Here, $\triangle^m \triangleq \triangle \times \ldots \times \triangle$ (m times). Finally let u be the uniform probability measure on $(\mathrm{Y}, \mathcal{Y})$, that is, $u(\mathrm{y}) = \frac{1}{|\mathrm{Y}|}\ \forall\ \mathrm{y} \in \mathrm{Y}$. In other words, u evenly distributes all the available probability mass across all possible assignments.

Consider the optimization problem $\mathcal{P}_{KL} : \min_{\mathrm{p} \in \triangle^m} \mathcal{C}_{KL}(\mathrm{p})$ where

$$\mathcal{C}_{KL}(\mathrm{p}) = \sum_{i=1}^{l} D_{KL}\big(r_i || p_i\big) + \mu \sum_{i=1}^{m} \sum_{j \in \mathcal{N}(i)} w_{ij} D_{KL}\big(p_i || p_j\big) - \nu \sum_{i=1}^{n} H(p_i).$$

Here $H(p) = -\sum_{\mathrm{y}} p(\mathrm{y}) \log p(\mathrm{y})$ is the Shannon entropy of p and $D_{KL}(p_i||q_j)$ is the KLD between measures p_i and q_j and is given by $D_{KL}(p||q) = \sum_{\mathrm{y}} p(\mathrm{y}) \log \frac{p(\mathrm{y})}{q(\mathrm{y})}$. (μ, ν) are hyper-parameters whose choice we discuss in Section 7. Given a vertex $i \in V$, $\mathcal{N}(i)$ denotes the set of neighbors of the vertex in the graph corresponding to w_{ij} and thus $|\mathcal{N}(i)|$ represents vertex i's degree.

Lemma 1 *If $\mu, \nu, w_{ij} \geq 0$, $\forall\ i, j$ then $\mathcal{C}_{KL}(\mathrm{p})$ is convex.*

Proof This follows as $D_{KL}(p_i||q_j)$ is convex in the pair (p_i, q_j), negative entropy is convex (Cover and Thomas, 1991), and we have a non-negative weighted combination of convex functions. ■

The goal of the above objective is to find the best set of measures p_i that attempt to: 1) agree with the labeled data r_j wherever it is available (the first term in $\mathcal{C}_{KL}$); 2) agree with each other when they are close according to a graph (the second graph-regularizer term in $\mathcal{C}_{KL}$); and 3) be smooth in some way (the last term in $\mathcal{C}_{KL}$). In essence, SSL on a graph consists of finding a labeling for $\mathcal{D}_u$ that is consistent with both the labels provided in $\mathcal{D}_l$ and the geometry of the data induced by the graph. In the following we discuss each of the above terms in detail.

The first term of $\mathcal{C}_{KL}$ will penalize the solution $p_i, i \in \{1, \ldots, l\}$, when it is far away from the labeled training data $\mathcal{D}_l$, but it does not insist that $p_i = r_i$, as allowing for deviations from r_i can help especially with noisy labels (Bengio et al., 2007) or when the graph is extremely dense in certain regions. As explained above, our framework allows for the case where supervised training is uncertain or ambiguous.

The second term of $\mathcal{C}_{KL}$ penalizes a lack of consistency with the geometry of the data and can be seen as a graph regularizer. If w_{ij} is large, we prefer a solution in which p_i and p_j are close in the KLD sense. One question about the objective relates to the asymmetric nature of KLD (i.e., $D_{KL}(p||q) \neq D_{KL}(q||p)$) (see Section 9.3 for a discussion about this issue in the directed graph case).

Lemma 2 *While KLD is asymmetric, given an undirected graph $\mathcal{G}$, the second term in the proposed objective, $\mathcal{C}_{KL}(\mathrm{p})$, is inherently symmetric.*

Proof As we have an undirected graph, $\mathbf{W}$ is symmetric, that is, $w_{ij} = w_{ji}$ and for every $w_{ij} D_{KL}(p_i||p_j)$, we also have $w_{ji} D_{KL}(p_j||p_i)$. ■

The last term encourages each p_i to be close to the uniform distribution (i.e., a maximum entropy configuration) if not preferred to the contrary by the first two terms. This acts as a guard against degenerate solutions commonly encountered in graph-based SSL (Blum and Chawla, 2001; Joachims, 2003). For example, consider the case where a part of the graph is almost completely disconnected from any labeled vertex—that is, a "pendant" graph component. This occurs sometimes in the case of k-NN graphs. In such situations the third term ensures that the nodes in this disconnected region are encouraged to yield a uniform distribution, validly expressing the fact that we do not know the labels of these nodes based on the nature of the graph. More generally, we conjecture that by maximizing the entropy of each p_i, the classifier has a better chance of producing high entropy results in graph regions of low confidence (e.g., close to the decision boundary and/or low density regions). This overcomes a common drawback of a large number of state-of-the-art classifiers (e.g., Gaussian mixture models, multi-layer perceptrons, Gaussian kernels) that tend to be confident even in regions far from the decision boundary.

Finally, while the second graph-regularizer term encourages high-entropy solutions for nodes that have high entropy neighbors, the graph regularizer alone is insufficient to yield high-entropy solutions in other cases where it may be desirable. For example, consider a connected pendant component that is "separated" from the rest of the graph by labeled nodes that have the same value. We can view this as a "lolly-pop" component, where the base of the stem is labeled, but the rest of the stem and the round portion of the lolly-pop are unlabeled. In such a configuration, the optimum configuration will set the label of all nodes to be equal to the labels of the stem. There can be cases,

however, where more uncertainty should be expressed about such a large mass of unlabeled nodes distantly situated from the nearest labeled node. The last term in the objective allows a solution where uncertainty is encouraged when a node is geodesically very distant from any label.

We conclude this section by summarizing some of the highlights and features of our framework:

1. *Manifold assumption:* $\mathcal{C}_{KL}$ uses the "manifold assumption" for SSL (see chapter 2 in Chapelle et al., 2007)—it assumes that the input data may be reasonably embedded within a low-dimensional manifold which in turn can be represented by a graph.

2. *Naturally multiclass:* As the objective is defined in terms of probability distributions over integers rather than just integers (or real-valued relaxations of integers Joachims, 2003; Zhu et al., 2003), the framework generalizes in a straightforward manner to multi-class problems. As a result, all the parameters are estimated jointly (compare to one vs. rest approaches which involve solving $|\mathrm{Y}|$ independent classification problems).

3. *Label uncertainty:* The objective is capable of handling uncertainty in the labels (encoded using r_i) (Pearl, 1990). We present an example of this in the scenario of text classification in Section 7.3.

4. *Ability to incorporate priors:* Priors can be incorporated by either

   (a) minimizing the KLD between an agglomerative measure and a prior, that is, $\mathcal{C}'_{KL}(\mathrm{p}) = \mathcal{C}_{KL}(\mathrm{p}) + \kappa D_{KL}(p_0||\tilde{p})$ where $\tilde{p}$ can for example be the arithmetic or geometric mean over p_i's or

   (b) minimizing the KLD between p_i and the prior p_0. First note that $\mathcal{C}_{KL}(\mathrm{p})$ may be re-written as $\mathcal{C}_{KL}(\mathrm{p}) = \sum_{i=1}^{l} D_{KL}\big(r_i||p_i\big) + \mu \sum_{i,j} w_{ij} D_{KL}\big(p_i||p_j\big) + \nu \sum_i D_{KL}\big(p_i||u\big)$ where u is uniform measure. This follows as $D_{KL}\big(p_i||u\big) = -H(p_i) + \text{const}$. Now if we replace the uniform measure, u, in the above by p_0 then we are asking for each p_i to be close to p_0. Even more generally, we may replace the uniform measure by a distinct fixed prior distribution for each vertex.

   While the former is more global, in the latter case, the prior effects each vertex individually. Also, the global prior is closer to the balance prior used in the case of algorithms like spectral graph transduction (Joachims, 2003). In both of the above cases, the resulting objective remains convex. It is also important to point out that using one of the above does not preclude us from using the other. We consider this to be a unique feature of our approach as we can incorporate both the balance and label priors simultaneously.

5. *Directed graphs:* The proposed objective can be used with directed graphs without any modification (see Section 9.3).

3.1 Solving $\mathcal{P}_{KL}$

As $\mathcal{C}_{KL}$ is convex and the constraints are linear, $\mathcal{P}_{KL}$ is a convex programming problem (Bertsekas, 1999). However, $\mathcal{P}_{KL}$ does not admit a closed form solution because the gradient of $\mathcal{C}_{KL}(\mathrm{p})$ w.r.t. $p_i(\mathrm{y})$ is of the form, $k_1 p_i(\mathrm{y}) \log p_i(\mathrm{y}) + k_2 p_i(\mathrm{y}) + k_3$ ($k_1,\ k_2,\ k_3$ are constants). Further, optimizing the dual of $\mathcal{P}_{KL}$ requires solving a similar equation. One of the reasons that $\mathcal{P}_{KL}$ does not admit a closed form solution is because we are optimizing w.r.t. to both variables in a KLD. Thus, we

are forced to use one of the numerical convex optimization techniques (Boyd and Vandenberghe, 2006) such as barrier methods (a type of interior point method, or IPM) or penalty methods (e.g., the method of multipliers (Bertsekas, 1999)). In the following we explain how method of multipliers (MOM) with quadratic penalty may be used to solve $\mathcal{P}_{KL}$. We choose a MOM based solver as it has been shown to be more numerically stable and has similar rates of convergence as other gradient based convex solvers (Bertsekas, 1999).

It can be shown that the update equations for $p_i(y)$ for solving $\mathcal{P}_{KL}$ using MOM are given by (see appendix A for details)

$$p_i^{(n)}(y) = \left[p_i^{(n-1)}(y) - \alpha^{(n-1)} \left(\frac{\partial \mathcal{L}_{\mathcal{C}_{KL}}(\mathrm{p}, \Lambda)}{\partial p_i(y)} \right)_{\{\mathrm{p}=\mathrm{p}^{(n-1)}, \Lambda=\Lambda^{(n-1)}, c=c^{(n-1)}\}} \right]^+$$

where $n = 1, \ldots,$ is the iteration index, $\alpha^{(n-1)}$ is the learning rate which is determined using the Armijo rule (Bertsekas, 1999), $[x]^+ = \max(x, 0)$ and

$$\frac{\partial \mathcal{L}_{\mathcal{C}_{KL}}(\mathrm{p}, \Lambda)}{\partial p_i(y)} = \mu \sum_{j \in \mathcal{N}(i)} \left[w_{ej} \big(1 + \log p_i(y) - \log p_j(y)\big) - \frac{w_{je} p_j(y)}{p_i(y)} \right] - \frac{r_i(y)}{p_i(y)} \delta(e \leq l) + \nu(\log p_i(y) + 1) + \lambda_i + 2c\big(1 - \sum_y p_i(y)\big).$$

In the above $\Lambda = \{\lambda_i\}$ are the Lagrange multipliers and c is the MOM coefficient (see appendix A).

While the MOM-based approach to solving $\mathcal{P}_{KL}$ is simple to derive, it has a number of drawbacks:

1. *Hyper(Extraneous)-Parameters:* Solving $\mathcal{P}_{KL}$ using MOM requires the careful tuning of a number of extraneous parameters including, the learning rate (α) which is obtained using the Armijo rule which has 3 other parameters, MOM penalty parameter (c), stopping criteria (ζ), and penalty update parameters (γ and β). In general, in the interest of scalability, it is advantageous to have as few tuning parameters in an algorithm as possible, especially in the case of SSL where there is relatively little labeled data available to "hold out" for use in cross validation tuning. The success of MOM based optimization depends on the careful tuning of all the 7 extraneous parameters (this is in addition to μ and ν, the hyper-parameters in the original objective). This is problematic as settings of these parameters that yield good performance on a particular data set have no generalization guarantees. In Section 7.2.1, we present an analysis that shows sensitivity of MOM to the settings of these parameters.

2. *Convergence guarantees:* For most problems, MOM lacks convergence guarantees. Bertsekas (1999) only provides a proof of convergence for cases when $c^{(n)} \to \infty$, a condition rarely satisfied in practice.

3. *Computational cost:* The termination criteria for the MOM based solver for $\mathcal{P}_{KL}$ requires that one compute the value of the objective function for every iteration leading to increased computational complexity.

4. *Lack of intuition in update equations:* While the update equations for $p_i(y)$ are easy to obtain, they lack an intuitive explanation.

As stated above, there are other alternatives for numerical optimization of convex functions. In particular, we could use an IPM for solving $\mathcal{P}_{KL}$, but barrier methods also have their own drawbacks (e.g., each step involves solving n linear equations). It is important to point out that we are not arguing against the use of gradient based approaches in general as they been quite successful for training multi-layer perceptrons, hidden conditional random fields, and so on where the objective is inherently non-convex. Sometimes even when the objective is convex, we need to rely on MOM or IPM for optimization like in our case in Section 9.2. However, as $\mathcal{P}_{KL}$ is a convex optimization problem, in this paper we explore and prefer other techniques for its optimization which do not have the aforementioned drawbacks.

4. Alternating Minimization (AM)

Given a distance function $d(p,q)$ between objects $p \in \mathcal{P}, q \in \mathcal{Q}$ where $\mathcal{P}, \mathcal{Q}$ are sets, consider the problem finding the p,q that minimizes $d(p,q)$. Sometimes solving this problem directly is hard, and in such cases the method of alternating minimization (AM) lends itself as a valuable tool for efficient optimization. AM refers to the case where we alternately minimize $d(p,q)$ with respect to p while q is held fixed and then vice-versa, that is,

$$p^{(n)} = \operatorname*{argmin}_{p \in \mathcal{P}} d(p, q^{(n-1)}) \text{ and } q^{(n)} = \operatorname*{argmin}_{q \in \mathcal{Q}} d(p^{(n)}, q).$$

Figure 1 illustrates the two steps of AM over two convex sets. We start with an initial arbitrary $Q_0 \in \mathcal{Q}$ which is held fixed while we minimize w.r.t. $P \in \mathcal{P}$ which leads to P_1. The objective is then held fixed w.r.t. P at $P = P_1$ and minimized over $Q \in \mathcal{Q}$ and this leads to Q_1. The above is then repeated with Q_1 playing the role of Q_0 and so on until (in the best of cases) convergence. The Expectation-Maximization (EM) (Dempster et al., 1977) algorithm is an example of AM. Moreover, the above objective over two variables can be extended to an objective over n variables. In such cases $n-1$ variables are held fixed while the objective is optimized with respect to the one remaining variable and the procedure iterates in a similar round-robin fashion.

An AM procedure might or might not have the following properties: 1) a closed-form solution to each of the alternating minimization steps of AM; 2) convergence to a final solution, and 3) convergence to a correct minimum of $d(p,q)$. In some cases, even when there is no closed-form solution to the direct minimization of $d(p,q)$, each step of AM has a closed form solution. In other cases, however (see Corduneanu and Jaakkola, 2003), one or both the steps of AM do not have closed form solutions.

Depending on $d(p,q)$ and on the nature of $\mathcal{P}, \mathcal{Q}$, an AM procedure might never converge. Even when AM does converge, it might not converge to the true correct minimum of $d(p,q)$. In general, certain conditions need to hold for AM to converge to the correct solution. Some approaches, such as Cheney and Goldstien (1959), Zangwill (1969) and Wu (1983), rely on the topological properties of the objective and the space over which it is optimized, while others such as Csiszar and Tusnady (1984) use geometrical arguments. Still others (Gunawardena, 2001) use a combination of the above.

In this paper, we take the *information geometry* approach proposed by Csiszar and Tusnady (1984) where the so-called *5-points property* (5-pp) is fundamental to determining whether AM on an objective converges to the global optima. 5-pp is defined as follows:

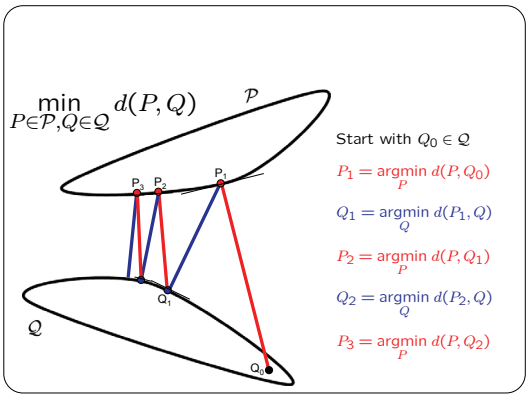


Figure 1: Alternating Minimization

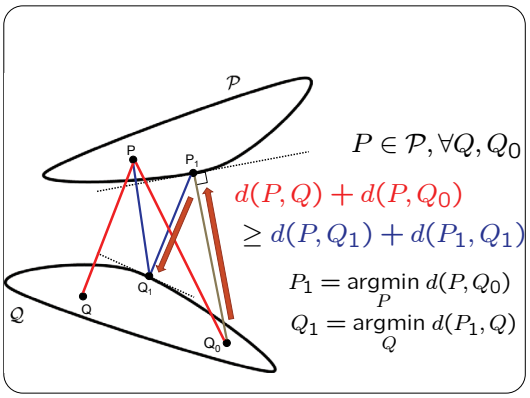


Figure 2: Illustration of the 5-point property

Definition 3 *If $\mathcal{P}$, $\mathcal{Q}$ are convex sets of finite measures, given a divergence $d(p,q)$, $p \in \mathcal{P}$, $q \in \mathcal{Q}$, then the 5-pp is said to hold for $p \in \mathcal{P}$ if $\forall\, q, q_0 \in \mathcal{Q}$ we have*

$$d(p,q) + d(p,q_0) \geq d(p,q_1) + d(p_1,q_1)$$

where $p_1 \in \operatorname{argmin}_{p\in\mathcal{P}} d(p,q_0)$ *and* $q_1 \in \operatorname{argmin}_{q\in\mathcal{Q}} d(p_1,q)$.

Figure 2 shows an illustration of 5-pp. Here we start with some $Q_0 \in \mathcal{Q}$, $P_1 = \operatorname{argmin}_{P\in\mathcal{P}} d(P,Q_0)$ and $Q_1 = \operatorname{argmin}_{Q\in\mathcal{Q}} d(P_1,Q)$. 5-pp is said hold for $d(P,Q)$ if for any $P \in \mathcal{P}$ and any $Q \in \mathcal{Q}$, the sum of the lengths of the red lines is greater than or equal to the sum of the lengths of the blue lines in Figure 2. Here the lengths are measured using the objective $d(P,Q)$. Csiszar and Tusnady (1984) have shown that the 5-pp holds for all p when $d(p,q) = D_{KL}(p||q)$.

So now the question is whether our proposed objective $\mathcal{C}_{KL}(\mathrm{p})$ can be optimized using AM and whether it converges to the correct optimum. This is the topic of discussion in the next section.

4.1 Graph-based SSL using AM

$\mathcal{P}_{KL}$ cannot be solved using AM and so we reformulate it in a manner amenable to AM. The following are the desired properties of such a reformulation –

1. The new (reformulated) objective should be a valid graph-based SSL criterion.
2. AM on the reformulated objective should converge to the global optimum of this objective.
3. The optimal solution in the case of the original ($\mathcal{P}_{KL}$) and reformulated problem should be identical.
4. Each step of the AM process should have a closed form and easily computable solution.
5. The resulting algorithm should scale to large data sets.

In this section, we formulate an objective that satisfies all of these properties. Consider the following reformulated objective –

$$\mathcal{P}_{MP} : \min_{\mathrm{p,q}\in\triangle^m} \mathcal{C}_{MP}(\mathrm{p},\mathrm{q}) \text{ where}$$

$$\mathcal{C}_{MP}(\mathrm{p},\mathrm{q}) = \sum_{i=1}^{l} D_{KL}\big(r_i||q_i\big) + \mu \sum_{i=1}^{m} \sum_{j\in\mathcal{N}'(i)} w'_{ij} D_{KL}\big(p_i||q_j\big) - \nu \sum_{i=1}^{m} H(p_i)$$

where for each vertex i in $\mathcal{G}$, we define a third discrete probability measure q_i over the measurable space $(\mathrm{Y},\mathcal{Y})$, $w'_{ij} = \left[\mathbf{W}'\right]_{ij}$, $\mathbf{W}' = \mathbf{W} + \alpha\mathbf{I}_n$, $\mathcal{N}'(i) = \{i\} \cup \mathcal{N}(i)$ and $\alpha \geq 0$. Here the q_i's play a similar role as the p_i's and can potentially be used to obtain a final classification result ($\mathrm{argmax}_y\, q_i(y)$). Thus, it would seem that we now have two classification results for each sample – one the most likely assignment according to p_i and another given by q_i. However, $\mathcal{C}_{MP}$ includes terms of the form $(w_{ii}+\alpha)D_{KL}(p_i||q_i)$ which encourage p_i and q_i to be close to each other. Thus α, which is a hyper-parameter, plays an important role in ensuring that $p_i = q_i,\ \forall\ i$. It should be clear that

$$\operatorname*{argmin}_{\mathrm{p}\in\triangle^n} \mathcal{C}_{KL}(\mathrm{p}) = \lim_{\alpha\to\infty} \operatorname*{argmin}_{\mathrm{p,q}\in\triangle^n} \mathcal{C}_{MP}(\mathrm{p},\mathrm{q}).$$

In the following we will show that there exists a finite α such that at a minima, $p_i(y) = q_i(y)\ \forall\ i,y$ (henceforth we will denote this as either $p_i = q_i\ \forall\ i$ or $\mathrm{p} = \mathrm{q}$).

We note that the new objective $\mathcal{C}_{MP}(\mathrm{p},\mathrm{q})$ can itself be seen as an intrinsically valid SSL criterion. While the first term encourages q_i for the labeled vertices to be close to the labels, r_i, the last term encourages higher entropy p's. The second term, in addition to acting as a graph regularizer, also acts as a glue between the p's and q's.

A natural question that arises at this point is why we choose this particular form for $\mathcal{C}_{MP}$ and not other alternatives. First note that $-H(p_i) = D_{KL}(p_i||u) + \text{const}$ where u is the uniform measure. KLD is a function of two variables (say the left and the right). In $\mathcal{C}_{MP}$, the p's always occur on the left hand side while the q's occur on the right. Recall that the reason $\mathcal{C}_{KL}$ did not admit a closed form solution is because we were attempting to optimize w.r.t. both the variables in a KLD. Thus going from $\mathcal{C}_{KL}$ to $\mathcal{C}_{MP}$ accomplishes two goals – (a) it makes optimization via AM possible, and (b) as we see shortly, it leads to closed form updates. Next we address the question of whether AM on $\mathcal{C}_{MP}$ converges to the correct optimum.

Lemma 4 *If $\mu, \nu, w'_{ij} \geq 0 \; \forall \; i, j$ then $\mathcal{C}_{MP}(\mathrm{p}, \mathrm{q})$ is convex.*

Proof This follows as $D_{KL}(p||q)$ is convex in the pair, and we have a weighted sum of convex functions with non-negative weights. ■

The previous lemma guarantees that any local minimum is a global minimum. The next theorem gives the powerful result that the AM procedure on our objective $\mathcal{C}_{MP}$ is guaranteed to converge to the true global minimum of $\mathcal{C}_{MP}$.

Theorem 5 (Convergence of AM on $\mathcal{C}_{MP}$, see appendix B) *If*

$$\mathrm{p}^{(n)} = \operatorname*{argmin}_{\mathrm{p} \in \triangle^m} \mathcal{C}_{MP}(\mathrm{p}, \mathrm{q}^{(n-1)}), \;\; \mathrm{q}^{(n)} = \operatorname*{argmin}_{\mathrm{q} \in \triangle^m} \mathcal{C}_{MP}(\mathrm{p}^{(n)}, \mathrm{q}) \text{ and } q_i^{(0)}(y) > 0 \; \forall \; y \in \mathrm{Y}, \; \forall i \text{ then}$$

(a) $\mathcal{C}_{MP}(\mathrm{p}, \mathrm{q}) + \mathcal{C}_{MP}(\mathrm{p}, \mathrm{p}^{(0)}) \geq \mathcal{C}_{MP}(\mathrm{p}, \mathrm{q}^{(1)}) + \mathcal{C}_{MP}(\mathrm{p}^{(1)}, \mathrm{q}^{(1)})$ *for all* $\mathrm{p}, \mathrm{q} \in \triangle^m$, *and*

(b) $\lim_{n \to \infty} \mathcal{C}_{MP}(\mathrm{p}^{(n)}, \mathrm{q}^{(n)}) = \inf_{\mathrm{p}, \mathrm{q} \in \triangle^m} \mathcal{C}_{MP}(\mathrm{p}, \mathrm{q})$.

Next we address the issue of showing that the solutions obtained in the case of the original and reformulated objectives are the same. We already know that if $\alpha \to \infty$ then we have equality, but we are interested in obtaining a finite lower-bound on α for which this is still the case. In the below, we let $\mathcal{C}_{MP}(\mathrm{p}, \mathrm{q}; \{w'_{ii} = 0\}_i)$ be the objective $\mathcal{C}_{MP}$ shown with the weight matrix parameterized with $w'_{ii} = 0$ for all i, and we let $\mathcal{C}_{MP}(\mathrm{p}, \mathrm{q}; \alpha)$ be the objective function shown with a particular parameterized value of α. For the proof of the next lemma and the two theorems that follow, see appendix C.

Lemma 6 *We have that*

$$\min_{\mathrm{p}, \mathrm{q} \in \triangle^m} \mathcal{C}_{MP}(\mathrm{p}, \mathrm{q}; w'_{ii} = 0) \leq \min_{\mathrm{p} \in \triangle^m} \mathcal{C}_{KL}(\mathrm{p}).$$

Theorem 7 *Given any $A, B, S \in \triangle^m$ (i.e., $A = [a_1, \ldots, a_n]$, $B = [b_1, \ldots, b_n]$, $S = [s_1, \ldots, s_n]$) such that $a_i(y), b_i(y), s_i(y) > 0, \; \forall \; i, y$ and $A \neq B$ (i.e., not all $a_i(y) = b_i(y)$) then there exists a finite α such that*

$$\mathcal{C}_{MP}(A, B) \geq \mathcal{C}_{MP}(S, S) = \mathcal{C}_{KL}(S).$$

The above theorem states that there exists a finite α that ensures $\mathcal{C}_{MP}(p, q)$ evaluated on any positive $p \neq q$ will be larger than any $\mathcal{C}_{KL}(\cdot)$. This is a stronger statement than we need, since we are interested only in what happens at the objective functions' minima. The following theorem does just this.

Theorem 8 (Equality of Solutions of $\mathcal{C}_{KL}$ and $\mathcal{C}_{MP}$) *Let*

$$\hat{\mathrm{p}} = \operatorname*{argmin}_{\mathrm{p} \in \triangle^m} \mathcal{C}_{KL}(\mathrm{p}) \text{ and } (\mathrm{p}^*_{\tilde{\alpha}}, \mathrm{q}^*_{\tilde{\alpha}}) = \operatorname*{argmin}_{\mathrm{p}, \mathrm{q} \in \triangle^m} \mathcal{C}_{MP}(\mathrm{p}, \mathrm{q}; \tilde{\alpha})$$

for an arbitrary $\tilde{\alpha} > 0$ where $\mathrm{p}^*_{\tilde{\alpha}} = (p^*_{1;\tilde{\alpha}}, \cdots, p^*_{m;\tilde{\alpha}})$ and $\mathrm{q}^*_{\tilde{\alpha}} = (q^*_{1;\tilde{\alpha}}, \cdots, q^*_{m;\tilde{\alpha}})$. Then there exists a finite $\hat{\alpha}$ such that at convergence of AM, we have that $\hat{\mathrm{p}} = \mathrm{p}^*_{\hat{\alpha}} = \mathrm{q}^*_{\hat{\alpha}}$. Further, it is the case that if $\mathrm{p}^*_{\tilde{\alpha}} \neq \mathrm{q}^*_{\tilde{\alpha}}$, then

$$\hat{\alpha} \geq \frac{\mathcal{C}_{KL}(\hat{\mathrm{p}}) - \mathcal{C}_{MP}(\mathrm{p}^*_{\tilde{\alpha}}, \mathrm{q}^*_{\tilde{\alpha}}; \alpha = 0)}{\mu \sum_{i=1}^n D_{KL}(p^*_{i;\tilde{\alpha}} || q^*_{i;\tilde{\alpha}})}$$

and if $\mathrm{p}^*_{\tilde{\alpha}} = \mathrm{q}^*_{\tilde{\alpha}}$ then $\hat{\alpha} \geq \tilde{\alpha}$.

We note that the above theorem guarantees the existence of a finite α that equates the minimum of $\mathcal{C}_{KL}$ and $\mathcal{C}_{MP}$ but it does not say how to find it since we do not know the true optimum of $\mathcal{C}_{MP}$. Nevertheless, if we use an α such that we end up with $p^* = q^*$ (or in practice, approximately so) then we are assured that this is the true optimum for $\mathcal{C}_{KL}$.

As mentioned above, AM is not always guaranteed to have closed form updates at each step, but in our case closed form updates may be achieved. The AM updates (see Appendix E for the derivation) are given by

$$p_i^{(n)}(y) = \frac{\exp\{\frac{\mu}{\gamma_i}\sum_j w'_{ij}\log q_j^{(n-1)}(y)\}}{\sum_y \exp\{\frac{\mu}{\gamma_i}\sum_j w'_{ij}\log q_j^{(n-1)}(y)\}} \quad \text{and}$$

$$q_i^{(n)}(y) = \frac{r_i(y)\delta(i \le l) + \mu\sum_j w'_{ji} p_j^{(n)}(y)}{\delta(i \le l) + \mu\sum_j w'_{ji}}$$

where $\gamma_i = \nu + \mu\sum_j w'_{ij}$.

Thus, $\mathcal{C}_{MP}$ satisfies all the desired properties of the reformulation. In addition, it is also possible to derive a test for convergence that does not require that one compute the value of $\mathcal{C}_{MP}(\mathrm{p},\mathrm{q})$ (i.e., evaluate the objective).

Theorem 9 (Test for convergence, see Appendix D) *If* $\{(\mathrm{p}^{(n)},\mathrm{q}^{(n)})\}_{n=1}^{\infty}$ *is generated by AM of* $\mathcal{C}_{MP}(\mathrm{p},\mathrm{q})$ *and* $\mathcal{C}_{MP}(\mathrm{p}^*,\mathrm{q}^*) \triangleq \inf_{\mathrm{p},\mathrm{q}\in\triangle^n} \mathcal{C}_{MP}(\mathrm{p},\mathrm{q})$ *then*

$$\mathcal{C}_{MP}(\mathrm{p}^{(n)},\mathrm{q}^{(n)}) - \mathcal{C}_{MP}(\mathrm{p}^*,\mathrm{q}^*) \le \sum_{i=1}^{n}\big(\delta(i \le l) + d_i\big)\beta_i,$$

$$\beta_i \triangleq \log\sup_y \frac{q_i^{(n)}(y)}{q_i^{(n-1)}(y)}, \quad d_j = \sum_i w_{ij}.$$

While a large number of optimization procedures resort to computing the change in the objective function with n (iteration index), in this case we have a simple check for convergence. This test does not require that one compute the value of the objective function which can be computationally expensive especially in the case of large graphs. Table 1 summarizes the advantages of the proposed AM approach to solving $\mathcal{P}_{MP}$ over that of using MOM to directly solve $\mathcal{P}_{KL}$. We also provide an empirical comparison of these approaches in Section 7.2.1. Henceforth, we refer to the process of using AM to solve $\mathcal{P}_{MP}$ as *measure propagation* (MP).

5. Squared-Loss Formulation

In this section, we show how the popular squared-loss objective may be formulated over measures. We then discuss its relationship to the proposed objective. Consider the optimization problem $\mathcal{P}_{SQ}$: $\min_{\mathrm{p}\in\triangle^m} \mathcal{C}_{SQ}(\mathrm{p})$ where

$$\mathcal{C}_{SQ}(\mathrm{p}) = \sum_{i=1}^{l} \| r_i - p_i \|^2 + \sum_{i=1}^{m}\sum_{j\in\mathcal{N}(i)} w_{ij} \| p_i - p_j \|^2 + \nu\sum_{i=1}^{m} \| p_i - u \|^2$$

and $\| p \|^2 = \sum_y p^2(y)$. $\mathcal{P}_{SQ}$ can also be seen as a multi-class extension of the *quadratic cost criterion* (Bengio et al., 2007) or as a variant of one of the objectives in Zhu and Ghahramani (2002b).

Criteria	MOM	AM
Iterative	YES	YES
Learning Rate	Armijo Rule	None
Number of Hyper-parameters	7	1 (α)
Test for Convergence	Requires Tuning	Automatic
Update Equations	Not Intuitive	Intuitive and easily Parallelized

Table 1: There are two ways to solving the proposed objective, namely, the popular numerical optimization tool method of multipliers (MOM), and the proposed approach based on alternating minimization (AM). This table compares the two approaches on various fronts.

Lemma 10 (Relationship between $\mathcal{C}_{KL}$ and $\mathcal{C}_{SQ}$) *We have that*

$$\mathcal{C}_{KL}(\mathrm{p}) \geq \frac{\mathcal{C}_{SQ}(\mathrm{p})}{\log 4} - m\nu \log |\mathrm{Y}|.$$

Proof By Pinsker's inequality we have that $D_{KL}(p||q) \geq (1/\log 4)\big(\sum_y |p(y)-q(y)|\big)^2 \geq (1/\log 4)\sum_y |p(y)-q(y)|^2$. As a result

$$\begin{aligned}
\mathcal{C}_{KL}(\mathrm{p}) &= \sum_{i=1}^{l} D_{KL}\big(r_i||p_i\big) + \mu \sum_{i=1}^{m} \sum_{j\in\mathcal{N}(i)} w_{ij} D_{KL}\big(p_i||p_j\big) - \nu \sum_{i=1}^{m} H(p_i) \\
&= \sum_{i=1}^{l} D_{KL}\big(r_i||p_i\big) + \mu \sum_{i=1}^{m} \sum_{j\in\mathcal{N}(i)} w_{ij} D_{KL}\big(p_i||p_j\big) + \nu \sum_{i=1}^{m} D_{KL}(p_i||u) - m\nu \log |\mathrm{Y}| \\
&\geq \frac{1}{\log 4}\left[\sum_{i=1}^{l} \| r_i - p_i \|^2 + \sum_{i=1}^{m} \sum_{j\in\mathcal{N}(i)} w_{ij} \| p_i - p_j \|^2 + \nu \sum_{i=1}^{m} \| p_i - u \|^2\right] - m\nu \log |\mathrm{Y}| \\
&= \frac{\mathcal{C}_{SQ}(\mathrm{p})}{\log 4} - m\nu \log |\mathrm{Y}|.
\end{aligned}$$

■

$\mathcal{P}_{SQ}$ can be reformulated as the following equivalent optimization problem $\mathcal{P}_{SQ}: \min_{\mathrm{p}\in\triangle^m} \mathcal{C}_{SQ}(\mathrm{p})$ where

$$\begin{aligned}
\mathcal{C}_{SQ}(\mathrm{p}) &= \mathrm{Tr}\big((S\mathrm{p}-\mathrm{r}')(S\mathrm{p}-\mathrm{r}')^T\big) + 2\mu \mathrm{Tr}(\mathbf{L}\mathrm{p}\mathrm{p}^T) + \nu \mathrm{Tr}((\mathrm{p}-\mathrm{u})(\mathrm{p}-\mathrm{u})^T), \\
S &\triangleq \begin{pmatrix} \mathbf{I}_l & 0 \\ 0 & 0 \end{pmatrix}, \quad \mathrm{r}' \triangleq \begin{pmatrix} \mathrm{r} & 0 \\ 0 & 0 \end{pmatrix}, \quad \mathrm{u} \triangleq (u,\ldots,u) \in \triangle^m,
\end{aligned}$$

$\mathbf{1}_m \in \mathbb{R}^m$ is a column vector of 1's, and $\mathbf{I}_l$ is the $l \times l$ identity matrix. Here $\mathbf{L} \triangleq \mathbf{D} - \mathbf{W}$ is the unnormalized graph Laplacian, $\mathbf{D}$ is a diagonal matrix given by $d_i = [\mathbf{D}]_{ii} = \sum_j w_{ij}$. $\mathcal{C}_{SQ}$ is convex if $\mu, \nu \geq 0$ and, as the constraints that ensure $\mathrm{p} \in \triangle$ are linear, we can make use of the KKT conditions (Bertsekas, 1999) to show that the solution to $\mathcal{P}_{SQ}$ is given by

$$\hat{\mathrm{p}} = (S + 2\mu\mathbf{L} + \nu\mathbf{I}_n)^{-1}\left[S\mathrm{r} + \nu\mathrm{u} + \frac{2\mu}{|\mathrm{Y}|}\mathbf{L}\mathbf{1}_n\mathbf{1}_c^T\right].$$

The above closed-form solution involves inverting a matrix of size $m \times m$. Henceforth we refer to the above closed form solution of $\mathcal{P}_{SQ}$ as *SQ-Loss-C* (C stands for closed form). Returning to the original formulation, using Lagrange multipliers, setting the gradient to zero and solving for the multipliers we get the update for p_i's to be

$$p_i^{(n)}(y) = \frac{r_i(y)\delta(i \leq l) + \nu u(y) + \mu \sum_j w_{ij} p_j^{(n-1)}(y)}{\delta(i \leq l) + \nu + \mu \sum_j w_{ij}}. \tag{1}$$

Here n is the iteration index. It can be shown that $\mathrm{p}^{(n)} \rightarrow \hat{\mathrm{p}}$ (Bengio et al., 2007). In the following we refer to the iterative method of solving $\mathcal{P}_{SQ}$ as *SQ-Loss-I*. There has not been any work in the past addressing the rate at which $\mathrm{p}^{(n)} \rightarrow \hat{\mathrm{p}}$ in the case of SQ-Loss-I. We address this issue in the following but first we define the rate of convergence of a sequence.

Definition 11 (Rate of Convergence Bertsekas, 1999) *Let $\{x_n\}$ be a convergent sequence such that $x_n \rightarrow 0$. It is said to have a linear rate of convergence if either*

$$x_n \leq q\eta^n \ \forall \ n \ \textit{or} \ \limsup_{n\rightarrow\infty} \frac{x_n}{x_{n-1}} \leq \eta$$

where $\eta \in (0,1)$ and $q > 0$.

As "geometric" rate of convergence is a more appropriate description of linear convergence, we use this term in the paper.

Theorem 12 (Rate of Convergence for SQ-Loss, see Appendix D) *If*

(a) $\nu > 0$, *and*

(b) $\mathbf{W}$ *has at least one non-zero off-diagonal element in every row (i.e.,* $\mathbf{W}$ *is irreducible)*

then the sequence of updates given in Equation 1 has a geometric rate of convergence for all i and y.

Thus we have that $\mathrm{p}^{(n)} \rightarrow \hat{\mathrm{p}}$ very quickly. It is interesting to consider a reformulation of $\mathcal{C}_{SQ}$ in a manner similar to $\mathcal{C}_{MP}$ (see Section 4.1), as we do next.

5.1 AM Amenable Formulation of $\mathcal{P}_{SQ}$

Consider the following reformulation of $\mathcal{C}_{SQ}$

$$\mathcal{C}'_{SQ}(\mathrm{p},\mathrm{q}) = \sum_{i=1}^{l} \parallel r_i - q_i \parallel^2 + \sum_{i=1}^{n} \sum_{j \in \mathcal{N}(i)} w'_{ij} \parallel p_i - q_j \parallel^2 + \nu \sum_{i=1}^{n} \parallel p_i - u \parallel^2 .$$

This form is amenable to AM and can be shown to satisfy 5-pp. Further the updates for two steps of AM have a closed form solution and are given by

$$p_i^{(n)}(y) = \frac{\nu u(y) + \mu \sum_j w'_{ij} q_j^{(n-1)}(y)}{\nu + \sum_j w'_{ij}},$$

$$q_i^{(n)}(y) = \frac{r_i(y)\delta(i \leq l) + \mu \sum_j w'_{ji} p_j^{(n)}(y)}{\delta(i \leq l) + \mu \sum_j w'_{ji}}.$$

We call this method *SQ-Loss-AM*. It is important to point out that for solving $\mathcal{P}_{SQ}$, one always resorts to either SQ-Loss-I or SQ-Loss-C depending on the nature of the problem. We will be using SQ-Loss-AM in the next section to provide more insights into the relationship between $\mathcal{P}_{KL}$ and $\mathcal{P}_{SQ}$.

6. Connections to Other Approaches

In this section we explore connections between our proposed approach and other previously proposed SSL algorithms.

6.1 Squared-Loss Based Algorithms

A majority of previously proposed graph-based SSL algorithms (Zhu et al., 2003; Joachims, 2003; Belkin et al., 2005; Bengio et al., 2007) are based on minimizing squared-loss. In the following we refer to the squared-loss based SSL algorithm proposed in Zhu and Ghahramani (2002a) as label propagation (LP) (this is the standard version of label propagation, see Table 2), the algorithm in Zhu et al. (2003) as the harmonic functions algorithms (HF). Also QC denotes the quadratic cost criterion (Bengio et al., 2007). While the objectives used in the case of LP, HF and QC are similar in spirit to our $\mathcal{C}_{SQ}$, there are some important differences. In the case of both HF and QC, the objective is defined over the reals whereas in our case $\mathcal{C}_{SQ}$ is defined over discrete probability measures. This leads to two important benefits – (a) it allows easy generalization to multi-class problems, (b) it allows us to exploit well-defined functions of measures in order to improve performance. Further, both the HF and LP algorithms do not have guards against degenerate solutions (i.e., the third term in $\mathcal{C}_{SQ}$). QC, on the other hand, employs a regularizer similar to the third term in $\mathcal{C}_{SQ}$ but QC is limited to only two-class problems (for multi-class problems one resorts to one vs. rest). Both the LP and HF algorithms optimize the same objective but LP uses a iterative solution while HF employs the closed form solution (it has been shown that LP converges to the solution given by HF Zhu, 2005a). QC is a generalization of HF and has been shown to outperform it (Bengio et al., 2007). Our squared-loss formulation, $\mathcal{C}_{SQ}$, is a generalization of QC for multi-class problems and as we show in Section 7.2.2, it outperforms QC. Thus, to compare against squared-loss based objectives, we simply use our formulation $\mathcal{C}_{SQ}$.

Table 2 summarizes the update equations in the case of some of the graph-based SSL algorithms. It is interesting to compare the update equations for SQ-Loss-AM and MP. It can be seen that the update equations for $q_i(y)$ in the case of SQ-Loss-AM and MP are the same. In the case of MP, the $p_i(y)$ update may be re-written as

$$p_i^{(n)}(y) = \frac{\prod_j \left(q_j^{(n-1)}(y)\right)^{\mu w'_{ij}}}{\sum_y \prod_j \left(q_j^{(n-1)}(y)\right)^{\mu w'_{ij}}}.$$

Thus, while squared loss makes use of a weighted arithmetic-mean, MP uses a weighted geometric-mean to update $p_i(y)$. In other words, while squared-error leads to additive updates, the use of KLD leads to multiplicative updates.

Algorithm	Update Equation(s)
MP	$p_i^{(n)}(y) = \frac{\exp\{\frac{\mu}{\gamma_i}\sum_j w'_{ij}\log q_j^{(n-1)}(y)\}}{\sum_y \exp\{\frac{\mu}{\gamma_i}\sum_j w'_{ij}\log q_j^{(n-1)}(y)\}}$ $q_i^{(n)}(y) = \frac{r_i(y)\delta(i\le l)+\mu\sum_j w'_{ji}p_j^{(n)}(y)}{\delta(i\le l)+\mu\sum_j w'_{ji}}$ $\gamma_i = \nu+\mu\sum_j w'_{ij}$
SQ-Loss-C	$\hat{\mathrm{p}} = (S+2\mu\mathbf{L}+\nu\mathbf{I}_m)^{-1}\left[S\mathrm{r}+\nu\mathrm{u}+\frac{2\mu}{\lvert\mathrm{Y}\rvert}\mathbf{L}\mathbf{1}_m\mathbf{1}_c^T\right]$ $\mathbf{L}\triangleq\mathbf{D}-\mathbf{W}, [\mathbf{D}]_{ii}=\sum_j w_{ij}$
SQ-Loss-I	$p_i^{(n)}(y) = \frac{r_i(y)\delta(i\le l)+\nu u(y)+\mu\sum_j w_{ij}p_j^{(n-1)}(y)}{\delta(i\le l)+\nu+\mu\sum_j w_{ij}}$
SQ-Loss-AM	$p_i^{(n)}(y) = \frac{\nu u(y)+\mu\sum_j w'_{ij}q_j^{(n-1)}(y)}{\nu+\sum_j w'_{ij}}$ $q_i^{(n)}(y) = \frac{r_i(y)\delta(i\le l)+\mu\sum_j w'_{ji}p_j^{(n)}(y)}{\delta(i\le l)+\mu\sum_j w'_{ji}}$
LP	$p_i^{(n)}(y) = \frac{r_i(y)\delta(i\le l)+\delta(i>l)\sum_j w_{ij}p_j^{(n-1)}(y)}{\delta(i\le l)+\delta(i>l)\sum_j w_{ij}}$

Table 2: A summary of update equations for various graph-based SSL algorithms. MP stands for our proposed measure propagation approach, SQ-Loss-C, SQ-Loss-I and SQ-Loss-AM represent the closed-form, iterative and alternative-minimization based solutions for the objective based on squared-error. LP is label propagation (Zhu and Ghahramani, 2002a). In all cases μ and ν are hyper-parameters.

Spectral graph transduction (SGT) (Joachims, 2003) is an approximate solution to the NP-hard norm-cut problem. The use of norm-cut instead of a mincut (as in Blum and Chawla, 2001) ensures that the number of unlabeled samples in each of the cuts is more balanced. SGT requires that one compute the eigen-decomposition of a $m \times m$ matrix which can be challenging for very large data sets. Manifold regularization (Belkin et al., 2005) proposes a general framework in which a parametric loss function that is defined over the labeled samples and is regularized by graph smoothness term defined over both the labeled and unlabeled samples. When the loss function satisfies certain conditions, it can be shown that the representer theorem applies and so the solution is a weighted sum over kernel computations. Thus the goal of the learning process is to discover these weights. When the parametric loss function is based on least squares, the approach is referred to as *Laplacian regularized least squares* (LapRLS) (Belkin et al., 2005) and when the loss function is based on hinge loss, the approach is called *Laplacian support vector machines* (LapSVM)) (Belkin et al., 2005). In the case of LapRLS, the weights have a closed form solution which involves inverting a $m \times m$ matrix while in the case of LapSVM, optimization techniques used for SVM training may be used to solve for the weights. In general, it has been observed that LapRLS and LapSVM give similar performance (see Chapter 11 in Chapelle et al., 2007). It very important to point out here that while LapSVM minimizes hinge loss (over the labeled samples) which is considered more appropriate than squared loss for classification, the graph regularizer is still based on squared error.

So is there a reason to prefer KLD based loss over squared-error? In this context we quote two relevant statements from Bishop (1995)

1. Page 226: " *In fact, the sum-of-squares error function is not the most appropriate for classification problems. It was derived from maximum likelihood on the assumption of Gaussian distributed target data. However, the target values for a l-of-c coding scheme are binary, and hence far from having a Gaussian distribution*."

2. Page 235: "*Minimization of the cross-entropy error function tends to result in similar relative errors on both small and large target values. By contrast, the sum-of-squares error function tends to give similar absolute errors for each pattern, and will therefore give large relative errors for small output values. This suggests that the cross-entropy error function is likely to perform better than sum-of-squares at estimating small probabilities*."

While the above quotes were made in the context of a multi-layered perceptron (MLP), they apply to learning in general. While squared-error has worked well in the case of regression problems (Bishop, 1995),[1] for classification, it is often argued that squared-loss is not the optimal criterion and alternative loss functions such as the cross-entropy (Bishop, 1995), logistic (Ng and Jordan, 2002), hinge-loss (Vladimir, 1998) have been proposed. When attempting to measure the dissimilarity between measures, KLD is said to be asymptotically consistent w.r.t. the underlying probability distributions (Bishop, 1995). The second quote above furthers the case in favor of adopting KLD based loss as it is based on relative error rather absolute error as in the case of squared-error. In addition, KLD is an ideal measure for divergence of probability distributions as it has description-length consequences (coding with the wrong distribution will lead to longer description bit length than necessary). Most importantly, as we will show in Section 7, MP outperforms the squared-error based $\mathcal{P}_{SQ}$ on a number of tasks. We also present further empirical comparison of these two objectives in Section 7.2.4.

We would like to note that Wang et al. (2008) proposed a graph-based SSL algorithm that also employs alternating minimization style optimization. However, it is inherently squared-loss based which MP outperforms (see Section 7). Further, they do not provide or state convergence guarantees and one side of their updates is not only not in the closed-form, but also it approximates an NP-complete optimization problem.

6.2 Information Regularization (Corduneanu and Jaakkola, 2003)

The information regularization (IR) (Corduneanu and Jaakkola, 2003) algorithm also makes use of a KLD based loss for SSL but is different from our proposed approach in following ways

1. IR is motivated from a different perspective. Here the input space is divided into regions $\{R_i\}$ which may or may not overlap. For a given point $x_j \in R_i$, IR attempts to minimize the KLD between $p_j(y|x_j)$ and $\hat{p}_{R_i}(y)$, the agglomerative distribution for region R_i. The intuition behind this is that, if a particular sample is a member of a region, then its posterior must be similar to the posterior of the other members. Given a graph, one can define a region to be a vertex and its neighbors thus making IR amenable to graph-based SSL. In Corduneanu and Jaakkola (2003), the agglomeration is performed by a simple averaging (arithmetic mean).

1. Assuming a Gaussian noise model in a regression problem leads to an objective based on squared-loss.

2. While IR suggests (without proof of convergence) the use of AM for optimization, one of the steps of the optimization does not admit a closed-form solution. This is a serious practical drawback especially in the case of large data sets.

3. It does not make use of a entropy regularizer. But as our results show, the entropy regularizer leads to much improved performance.

Tsuda (2005) (hereafter referred to as PD) is an extension of the IR algorithm to hyper-graphs where the agglomeration is performed using the geometric mean. This leads to closed form solutions in both steps of the AM procedure. However, like IR, PD does not make use of a entropy regularizer. Further, the update equation for one of the steps of the optimization in the case of PD (Equation 13 in Tsuda, 2005) is actually a special case of our update equation for $p_i(y)$ and may be obtained by setting $w_{ij} = 1/2$. Further, our work here can be easily extended to hyper-graphs (see Section 9.3).

7. Results

Table 3 lists the data sets that we use in this paper. These corpora come from a diverse set of domains, including image processing (handwritten digit recognition), natural language processing (document classification, webpage classification, dialog-act tagging), and speech processing (phone classification). The sizes vary from $m = 400$ (BCI) to the largest data set, Switchboard, which has 120 million samples. The number of classes vary from $|\mathrm{Y}| = 2$ to $|\mathrm{Y}| = 72$ in the case of Dihana. The goal is to show that the proposed approach performs well on both small and large data sets, for binary and multi-class problems. Further, in each case we compare the performance of MP against the state-of-the-art algorithm for that task. Each data set is described in detail in the relevant sections.

7.1 Synthetic 2D Two-Moon Data Set

In order to understand the advantages of MP over other state-of-the-art SSL algorithms, we evaluated their performance on the synthetic 2D two-moon data set. This is a binary classification problem. We compare against SQ-Loss-I (see Section 5), LapRLS (Belkin et al., 2005), and SGT (Joachims, 2003). For all approaches, we constructed a symmetrized 10-NN graph using an RBF kernel. In the case of LapRLS and SGT, the hyper-parameter values were set in accordance to the recipe in Belkin et al. (2005) and Joachims (2003) respectively. In the case of MP, we set $\mu = 0.2$, $\nu = 0.001$ and $\alpha = 1.0$. For SQ-Loss-I, we set $\mu = 0.2$ and $\nu = 0.001$. These values were found to give reasonable performance for most data sets.

We used three different types of labelings: (a) two labeled samples from each class, (b) 4 samples from one class and 1 sample from the other class, and (c) 10 samples from one class and 1 sample from the other class. While the first represents the 'balanced' case, that is, equal number of labeled samples from the two classes, the others are 'imbalanced' conditions. In other words, (b) and (c) are representative of cases where the distribution over the labeled samples is not reflective of the underlying distribution over the classes (there are equal number of samples in each class). The results for each of the different labeling are shown in Figure 3. The first column shows the results obtained using SQ-Loss-I, the second column shows the results of LapRLS, the third is SGT and the fourth (last) column is MP. The following observations can be made from these results

Data Set	m	$\|\mathrm{Y}\|$	$H_N(p_0)$	Task
2D Two-Moon	500	2	1	Synthetic
BCI	400	2	1	Brain Computer Interface
USPS	1500	2	0.7	HandWritten Digits
Digit1	1500	2	1	Synthetic
COIL	1500	6	1	Image Recognition
Text	1500	2	1	Newsgroups Newswires
OPT-Digits	1797	10	1	HandWritten Digits
Reuters-21578	9603	10	0.8	Document Classification
WebKB	8282	4	0.9	Webpage Classification
Dihana	23,500	72	0.8	Dialog-Act Tagging
Switchboard-DA	185,000	18	0.6	Dialog-Act Tagging
TIMIT	1.4 million	48	0.9	Phone Classification
Switchboard	120 million	53	0.8	Phone Classification

Table 3: List of Data Sets we used to compare the performance of various SSL algorithms. $H_N(p_0) = H(p_0)/\log|\mathrm{Y}|$ is the normalized entropy of the prior and a value of 1 indicates a perfectly balanced data set while values closer to 0 imply imbalance. In the case of the Switchboard data set, $H_N(p_0)$ was computed over the STP data (see Section 8.1).

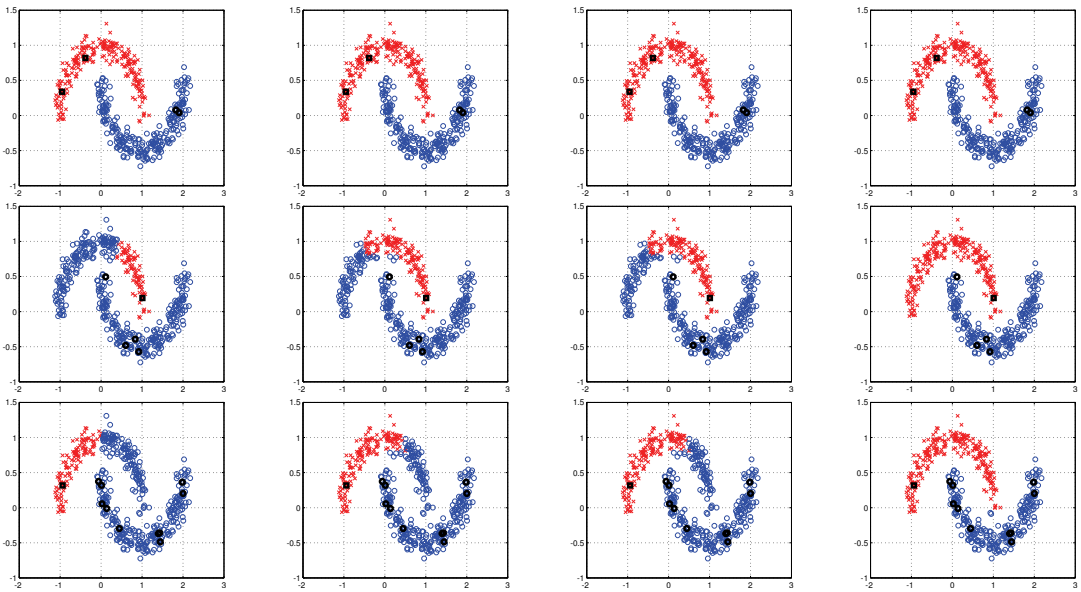


Figure 3: Results on the 2D two-moon data set. Each row shows results for different labelings and in each case the labeled points are shown in "black". The first column shows results obtained using SQ-Loss-I, the second column results were obtained using LapRLS, SGT was used for the third column and the last column shows the results in the case of MP.

1. MP is able to achieve perfect classification in the first two cases, and essentially perfect (2 errors) in the third case.

2. In the balanced case (first row), all approaches achieve perfect classification. Here, all approaches are able to correctly learn the nature of the manifold.

3. In the imbalanced cases (second and third rows), all three other approaches (SQ-Loss-I, LapRLS, and SGT) fail to correctly classify a significant portion of samples. This is not surprising and has been observed by others in the past (see Figure 1 in Wang et al., 2008).

4. Finally, in the case of SQ-Loss-I, we tried using class mass normalization (CMN) (Zhu and Ghahramani, 2002a) as a post-processing step. While the results did not change in the balanced case, CMN in fact resulted in worse error rate performance in the imbalanced cases. Note that Figure 3 for SQ-Loss-I does not include CMN.

7.2 Results on Benchmark SSL Data Sets

We also evaluated the performance of MP on a number of benchmark SSL data sets including, USPS, Text, Digit1, BCI, COIL and Opt-Digits. All the above data sets, with the exception of Opt-Digits (obtained from the UCI machine learning repository), came from `http://www.kyb.tuebingen.mpg.de/ssl-book`. Digit1 is a synthetic data set, USPS is a handwritten digit recognition task, BCI involves classifying signals obtained from a brain computer interface, COIL is a part of the Columbia object image recognition library and involves classifying objects using images taken at different orientations. Text involves classifying IBM vs. the rest for documents taken from the top 5 categories in comp.* newswire. Opt-Digits is also a handwritten digit recognition task. We note that most of these data sets are perfectly balanced (see Table 3)—further details may be found in Chapelle et al. (2007).

We compare MP against four other algorithms: 1) k-nearest neighbors; 2) Spectral Graph Transduction (SGT) (Joachims, 2003); 3) Laplacian Regularized Least Squares (LapRLS) (Belkin et al., 2005); and 4) $\mathcal{P}_{SQ}$ solved using SQ-Loss-I. Here k-nearest neighbors is the fully-supervised approach, while others are graph-based SSL approaches. We used the standard features supplied with the corpora without any further processing. For the graph-based approaches we constructed symmetrized k-NN graphs using an RBF kernel. We discuss the choice of k and the width of the kernel shortly. For each data set, we generated transduction sets with different number of labeled samples, $l \in \{10, 20, 50, 80, 100, 150\}$. In each case, we generated 11 different transduction sets. The first set was used to tune the hyper-parameters which were then held fixed over the remaining sets. In the case of the k-nearest neighbors approach, we tried $k \in \{1, 2, 4, 5, 10, 20, 30, 40, 50, 70, 90, 100, 120, 140, 150, 160, 180, 200\}$. For the graph-based approaches, k (for the k-NN graph) was tuned on the first transduction set over the following values $k \in \{2, 5, 10, 50, 100, 200, m\}$. The optimal width of the RBF kernel, σ, in the case of SQ-Loss-I, SGT and MP was determined over the following set $\sigma \in \{g_a/3 : a \in \{2, 3, \cdots, 10\}\}$ where g_a is the average distance between each sample and its a^{th} nearest neighbor over the entire data set (Bengio et al., 2007).

In the case of LapRLS, we followed the setup described in Section 21.2.5 of Chapelle et al. (2007). Here, as per the recipe in Joachims (2003), the optimal σ was determined in a slightly different manner—we tried $\sigma \in \{\frac{\sigma_0}{8}, \frac{\sigma_0}{4}, \frac{\sigma_0}{2}, \sigma_0, 2\sigma_0, 4\sigma_0, 8\sigma_0\}$ where σ_0 is the average norm of the feature vectors. In addition the hyper-parameters γ_A, r (see Belkin et al., 2005) associated with LapRLS were tuned over the following values: $\gamma_A \in \{$1e–6, 1e–4, 1e–2, 1, 100$\}$, $r \in \{0,$ 1e–4, 1e–2, 1, 100,

	USPS						**Digit1**					
l	10	20	50	80	100	150	10	20	50	80	100	150
k-NN	80.0	80.4	90.7	92.7	93.6	94.9	67.6	79.5	90.2	93.2	91.2	95.2
SGT	86.2	87.9	**94.0**	**95.7**	**96.0**	**97.0**	**92.1**	93.6	96.2	**97.1**	**97.4**	**97.7**
LapRLS	83.9	86.9	**93.7**	94.7	95.4	95.9	**92.4**	**95.3**	95.7	96.2	97.1	97.4
SQ-Loss-I	81.4	82.0	**93.6**	**95.8**	95.2	95.2	91.2	94.9	**96.9**	96.6	97.2	97.1
MP	**88.1**	**90.4**	**93.9**	95.0	**96.2**	**96.8**	**92.1**	**95.1**	96.1	**97.4**	**97.4**	**97.8**

	BCI						**Text**					
l	10	20	50	80	100	150	10	20	50	80	100	150
k-NN	48.5	52.4	53.3	50.6	53.1	53.5	60.2	64.2	71.6	72.4	72.3	74.5
SGT	49.7	50.4	**52.2**	52.4	**53.6**	54.5	**70.4**	70.9	**73.1**	**76.9**	**77.0**	**78.1**
LapRLS	**53.3**	**53.4**	**52.7**	**53.6**	**53.9**	56.1	68.2	69.1	71.2	73.4	74.2	76.2
SQ-Loss-I	51.0	51.3	50.7	**53.2**	53.3	53.1	67.9	**72.0**	**74.1**	**76.8**	**76.8**	76.6
MP	**53.0**	**53.2**	**52.8**	**53.9**	**54.0**	**57.0**	**70.3**	**72.6**	**73.0**	75.9	75.4	**77.9**

	COIL						**OPT**					
l	10	20	50	80	100	150	10	20	50	80	100	150
k-NN	34.5	53.9	66.9	77.9	79.2	83.5	79.6	83.9	85.5	90.5	92.0	93.8
SGT	40.1	61.2	78.0	88.5	89.0	89.9	90.4	90.6	91.4	94.7	**97.4**	**97.4**
LapRLS	**49.2**	61.4	78.4	80.1	84.5	87.8	89.7	**91.2**	92.3	96.1	**97.6**	**97.3**
SQ-Loss-I	**48.9**	63.0	**81.0**	87.5	89.0	90.9	**92.2**	90.2	**95.9**	**97.2**	**97.3**	**97.7**
MP	47.7	**65.7**	78.5	**89.6**	**90.2**	**91.1**	90.6	**90.8**	94.7	**96.6**	**97.0**	**97.1**

Table 4: Comparison of accuracies for different number of labeled samples (l) across USPS, Digit1, BCI, Text, COIL and Opt-Digits data sets. In each column, the best performing system and all approaches that are not significantly different at the 0.001 level (according to the difference of proportions significance test) are shown bold-faced.

1e4, 1e6}. Also, as per Belkin et al. (2005), we set $p = 5$ in the case of Text data set and $p = 2$ for all the other data sets. In the case of SGT, the search was over $c \in \{3000, 3200, 3400, 3800, 5000, 100000\}$ (Joachims, 2003). Finally, the trade-off parameters, μ and ν (associated with both MP and SQ-Loss-I) were tuned over the following sets: $\mu \in \{1e{-}8, 1e{-}6, 1e{-}4, 1e{-}2, 0.1, 1, 10\}$ and $\nu \in \{1e{-}8, 1e{-}6, 1e{-}4, 1e{-}2, 0.1\}$. In the case of SQ-Loss-I, the results were obtained after the application of CMN as a post-processing step as this has been shown to be beneficial to the performance on benchmark data sets (Chapelle et al., 2007). For MP, we initialized $\mathrm{p}^{(0)}$ such that all assignments had non-zero probability mass as this is a required condition for convergence and set $\alpha = 1$. As LapRLS and SGT assume binary classification problems, results for the multi-class data sets (COIL and OPT) were obtained using one vs. rest.

The mean accuracies over the 10 transduction sets (i.e., excluding the set used for tuning the hyper-parameters) for each corpora is shown in Table 4. The following observations may be made from these results

1. As expected, for all approaches, an increase in number of labeled samples leads to increased accuracy.

	USPS	Text	Digit1	BCI	COIL	Opt-Digits
LP-3	77.2	65.1	70.1	51.5	31.3	81.2
MOM	88.1	70.3	91.4	53.0	46.1	91.2
MP	88.2	70.3	92.1	53.0	47.7	93.4
MOM′	81.1	67.6	79.4	51.7	41.2	90.4

Table 5: Comparison of performance of MOM and MP. Results are in accuracies for the $l = 10$ case. We also show the results obtained after three iterations of LP (LP-3) (Zhu and Ghahramani, 2002a) as this was used to initialize MOM. MOM′ are the results obtained using the MOM setup with a small change in the setting of the hyper-parameters.

2. MP performs best in 15 out of the 36 cases, SQ-Loss is best in 10 out of the 36 cases, SGT in 8 out of the 36 cases and LapRLS in 7 out of the 36 cases. In 13 of cases in which MP was not the best, it was not significantly different compared to the winner (we characterize an improvement as being significant if it is significant at the 0.001 level according to a difference of proportions significance test).

3. It can be seen that SGT does best in the case of the Text corpus for a majority of the values of l, while MP is the best in a majority of the cases in the COIL and BCI data sets. SQ-Loss does best in the case of OPT. Thus in the case of the two multi-class data sets, the two *true* multi-class approaches perform better than the SSL approaches that use one vs. rest.

4. We also tried SQ-Loss-C and SQ-Loss-AM for solving the squared-loss based objective and in a majority of the cases the performance was the same as SQ-Loss-I. In other cases, the difference was insignificant. It should however be noted that using SQ-Loss-C to solve large problems can be rather difficult.

5. While there are no silver bullets in SSL (Zhu, 2005b), our MP algorithm outperforms other approaches in a majority of the cases. We would like to point out the diversity of the data sets used in the above experiment.

6. Finally note that while we have a used a simple approach to hyper-parameter selection, there are other ways of choosing them such as Goldberg and Zhu (2009)

7.2.1 MP VS. MOM

In this section we compare the results obtained from using MP against results obtained by directly optimizing the original objective, C_{KL} (henceforth we refer to this as MOM). As explained in Section 3.1, implementing MOM requires the careful tuning of a number optimization related hyper-parameters (in addition to μ and ν). After extensive experimentation, we found that setting, $\gamma = 0.25$, $\beta = 5$ and $\zeta =$1e–6 gave reasonable results. Further, as MOM is gradient based, we initialized $p^{(0)}$ (see Section 3.1) to the distributions obtained after 3 iterations of the label propagation algorithm described in Zhu and Ghahramani (2002a) (henceforth referred to as LP-3).

Table 5 shows average accuracies over all transduction sets for $l = 10$ (the trends were similar for other values of l) in the case of the corpora described in the previous section for (a) LP-3, (b) MOM (c) MP, and (d) MOM′. In the case of MOM′, we changed the values of the optimization

related hyper-parameters to $\gamma = 0.2$ and $\beta = 3$. The goal here is to show the sensitivity of MOM$'$ to the exact settings of the hyper-parameter values. The following observations can be made from these results

1. MOM outperforms LP-3. This implies MOM is able to learn over and beyond the set of distributions that result from 3 iterations of LP.

2. In the case of USPS, Digit1, COIL, Opt-Digits, MP outperforms MOM. Further, the performance gap between MP and MOM grows with the size of the data set. MP significantly outperforms MOM at the 0.0001 level in the case of the Opt-Digits. This might seem surprising because when we have that $p^* = q^*$ in the case of MP, the results obtained using MOM cannot be any worse than those obtained using MP (because the objective is convex). We conjecture that this is because MOM involves using a penalty parameter $c^{(n)}$ that tends to increase with n leading to slow convergence. This is more likely to happen in the neighborhood of p^* (Bertsekas, 1999). As a result MOM is terminated when the rate of the change of $p^{(n)}$ falls below some ζ and so it is possible that the objective has not attained the optimal value. In the case of MP, on the other hand, no such issues exist. Further we have a test for convergence (see Theorem 9).

3. The results obtained in the case of MOM$'$ show that this approach can be very sensitive to the settings of the hyper-parameters. While it may be possible to tune the various MOM related hyper-parameters in the case of small data sets, it is much less feasible in the case of large data sets.

7.2.2 ONE VS. REST AGAINST TRUE MULTI-CLASS

It is often argued binary classifiers when used within a one vs. rest framework perform at least as well as true multi-class solutions (Rifkin and Klautau, 2004). In this section, we test this claim in the context of SSL. We make use of the two multi-class data sets, COIL and OPT-Digits. Figure 4 shows a comparison of the performance of $\mathcal{P}_{SQ}$ (solved using SQ-Loss-C) and QC (Bengio et al., 2007). Even though SQ-Loss-I converges to SQ-Loss-C, in this case we used SQ-Loss-C as the size of the data set is small. As QC can handle only binary classification problems, the results for QC were generated using one vs. rest. Note that SQ-Loss-C is simply the closed form solution of $\mathcal{P}_{SQ}$ which is the multi-class extension of the QC objective. In the case of both the approaches, (a) the graph was generated by using an RBF kernel over the Euclidean distance, (b) we used the closed form solution, and (c) hyper-parameter search was done over exactly the same set of values. It can be seen that SQ-Loss-C outperforms QC in all cases. As the objectives are both inherently based on squared-error, the performance improvement in going from QC to $\mathcal{P}_{SQ}$ is likely because $\mathcal{P}_{SQ}$ is a true multi-class objective, that is, all the parameters are estimated jointly.

7.2.3 EFFECTS OF ENTROPY REGULARIZATION

We also wish to explore the effects of the entropy regularizer. We ran MP using the same setup described in Section 7.2 but with $\nu = 0$. The results in the $l = 10$ case are shown in Table 6. Similar trends were observed in the case of other values of l. It can be seen that entropy regularization leads to improved performance in the case of all data sets. We moreover have seen this trend in the other data sets (results not reported herein). The entropy regularizer encourages solutions closer to

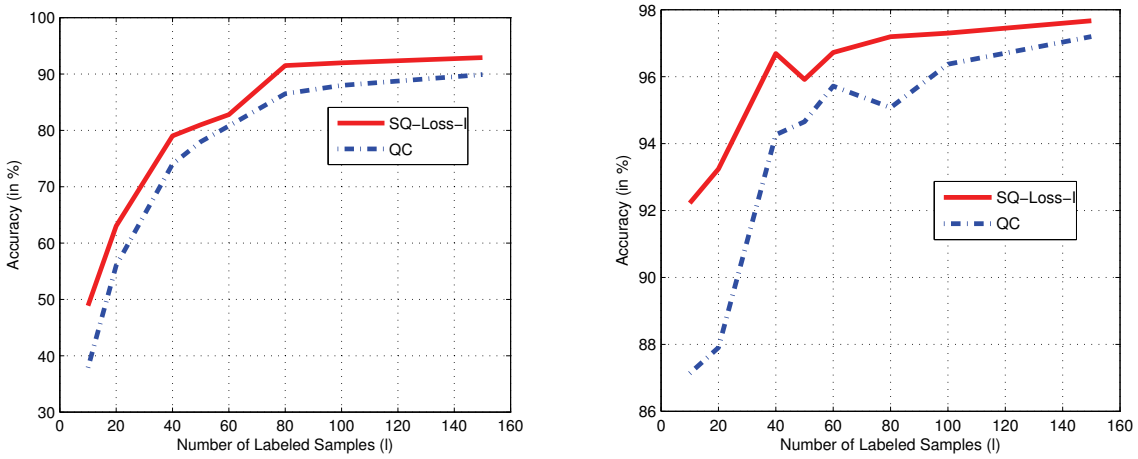


Figure 4: Comparison of the one vs. rest approach against true multi-class classifier. Figures show accuracy (in %) vs. Number of Labeled Samples (l) for (a)-left COIL and (b)-right OPT-DIGITS data sets. SQ-Loss-I is the solution to a true multi-class objective while QC makes use of one vs. rest approach for multi-class problems.

	USPS	Text	Digit1	BCI	COIL	Opt-Digits
MP ($\nu = 0$)	85.7	70.0	91.7	51.1	45.2	89.5
MP	88.2	70.3	92.1	53.0	47.7	93.4

Table 6: Comparison of performance of MP with and without ($\nu = 0$) entropy regularization. Results are in accuracies for the $l = 10$ case.

the uniform distribution, and we mentioned above that this helps to retain uncertainty in portions of graph very isolated from label information. To explain why this could lead to actual *improved* performance, however, we speculate that the entropy term is beneficial for the same reason as that of maximum entropy estimation—except for evidence to the contrary, we should prefer solutions that are as indifferent as possible.

7.2.4 SENSITIVITY OF MP AND SQ-LOSS-I TO σ

In this section, we examine the effects of change in hyper-parameters settings on the performance of $\mathcal{P}_{SQ}$ (solved using SQ-Loss-I) and MP. In particular, we look at the effects of varying the width of the RBF kernel used to generate the weighted graph. Figure 5 shows results obtained for the $l = 50$ case in the USPS and Opt-Digits data sets (in each case the value of σ at the mode of each curve is its optimal value). It can be seen that in the case of both the data sets, the performance variation is larger in the case of SQ-Loss-I while MP is more robust to the value of σ. Note that in the case of Opt-Digits, at the optimal value for σ, SQ-Loss-I outperforms MP. Similar trends were observed in the case of other data sets. As the choice of hyper-parameters in an issue in SSL, we prefer approaches that are more robust to the value of the hyper-parameters. We believe the robustness

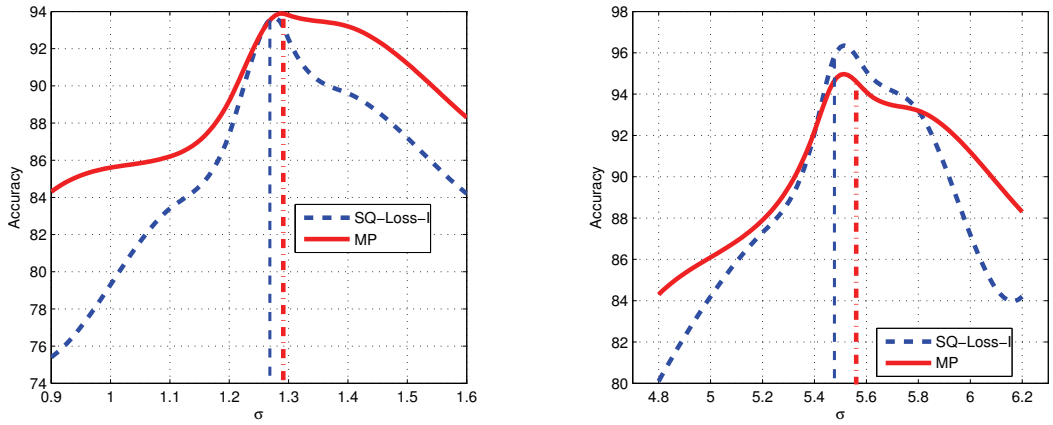


Figure 5: Figures showing the variation of accuracy with change in the width (σ) of the RBF kernel. The left figure was generated using the USPS data set for the $l = 50$ case while the right figure was generated using the Opt-Digits data set for the $l = 50$ case. The vertical lines (blue for SQ-Loss-I and red for MP) depict the σ given by the algorithm described in the previous section.

of MP is due to the fact that it is inherently based on KLD which is more suited for classification compared to squared error.

7.3 Text Classification

Text classification involves automatically assigning a given document to a fixed number of semantic categories. Each document may belong to one, many, or none of the categories. In general, text classification is a *multi-class* problem (more than 2 categories). Training fully-supervised text classifiers requires large amounts of labeled data whose annotation can be expensive (Dumais et al., 1998). As a result there has been interest is using SSL techniques for text classification (Joachims, 1999, 2003). However past work in semi-supervised text classification has relied primarily on one vs. rest approaches to overcome the inherent multi-class nature of this problem. We compare our algorithm (MP) with other state-of-the-art text categorization algorithms, namely: (a) SVM (Joachims, 1999); (b) Transductive-SVM (TSVM) (Joachims, 1999); (c) Spectral Graph Transduction (SGT) (Joachims, 2003); and (d) $\mathcal{P}_{SQ}$ solved using SQ-Loss-I. Apart from MP, SGT and SQ-Loss-I are graph-based algorithms, while SVM is fully-supervised (i.e., it does not make use of any of the unlabeled data). As shown by the results in Joachims (2003), SGT outperforms other SSL algorithms for this task. Thus we choose to compare against SGT. We implemented SVM and TSVM using *SVM Light* (Joachims, 2002) and SGT using *SGT Light* (Joachims, 2004). In the case of SVM, TSVM and SGT we trained $|\mathrm{Y}|$ classifiers (one for each class) in a one vs. rest manner precisely following Joachims (2003). We used two real-world data sets: (a) Reuters-21578 and (b) WebKB. In the following we discuss the application of the above algorithms to these data sets.

7.3.1 REUTERS-21578

We used the "ModApte" split of the Reuters-21578 data set collected from the Reuters newswire in 1987 (Lewis et al., 1987). The corpus has 9,603 training (not to be confused with $\mathcal{D}$) and 3,299 test documents (which represents $\mathcal{D}_u$). Of the 135 potential topic categories only the 10 most frequent categories are used (Joachims, 1999). Categories outside the 10 most frequent were collapsed into one class and assigned a label "other". For each document i in the data set, we extract features $\mathbf{x}_i$ in the following manner: stop-words are removed followed by the removal of case and information about inflection (i.e., stemming) (Porter, 1980). We then compute TFIDF features for each document (Salton and Buckley, 1987). We constructed symmetrized k-NN graphs with weights generated using cosine similarity between TFIDF features generated as explained above.

For this task $\mathrm{Y} = \{\ earn, acq, money, grain, crude, trade, interest, ship, wheat, corn, average\}$. For SQ-Loss-I and MP, we use the output space $\mathrm{Y}' = \mathrm{Y} \cup \{\ other\ \}$. For documents in $\mathcal{D}_l$ that are labeled with multiple categories, we initialize r_i to have equal non-zero probability for each such category. For example, if document i is annotated as belonging to classes $\{\ acq, grain, wheat\}$, then $r_i(acq) = r_i(grain) = r_i(wheat) = 1/3$. Note that there might be other (non-uniform) ways of initializing r_i (e.g., using word counts).

We created 21 transduction sets by randomly sampling l documents from the standard Reuters training set with the constraint that each of 11 categories (top 10 categories and the class *other*) are represented at least once in each set. These samples constitute $\mathcal{D}_l$. All algorithms used the same transduction sets. In the case of SGT, SQ-Loss-I and MP, the first transduction set was used to tune the hyper-parameters which we then held fixed for all the remaining 20 transduction sets. For all the graph-based approaches, we ran a search over $k \in \{2, 10, 50, 100, 250, 500, 1000, 2000, m\}$ (note $k = m$ represents a fully connected graph, i.e., a clique). In addition, in the case of MP, we set $\alpha = 2$ for all experiments, and we ran a search over $\mu \in \{1\text{e–}8, 1\text{e–}4, 0.01, 0.1, 1, 10, 100\}$ and $\nu \in \{1\text{e–}8, 1\text{e–}6, 1\text{e–}4, 0.01, 0.1\}$. In the case of SGT, the search was over $c \in \{3000, 3200, 3400, 3800, 5000, 100000\}$ (Joachims, 2003).

We report precision-recall break even point (PRBEP) results on the 3,299 test documents in Table 7. PRBEP has been a popular measure in information retrieval (see, e.g., Raghavan et al., 1989). It is defined as that value for which precision and recall are equal. Results for each category in Table 7 were obtained by averaging the PRBEP over the 20 transduction sets. The final row "average" was obtained by macro-averaging (average of averages). The optimal value of the hyper-parameters in case of SQ-Loss-I was $k = 100$; in case of MP, $k = 1000$, $\mu = 1\text{e–}4$, $\nu = 1\text{e–}4$; and in the case of SGT, $k = 100$, $c = 3400$. The results show that MP outperforms the state-of-the-art on 6 out of 10 categories and is competitive in 3 of the remaining 4 categories. Further it significantly outperforms all other approaches in case of the macro-averages. MP is significantly better at the 0.001 level over its nearest competitor (SGT) according to a difference of proportions significance test.

Figure 6 shows the variation of "average" PRBEP (last row in Table 7) against the number of labeled documents (l). For each value of l, we tuned the hyper-parameters over the first transduction set and used these values for all the other 20 sets. Figure 6 also shows error-bars ($\pm$ standard deviation) for all the experiments. As expected, the performance of all the approaches improves with increasing number of labeled documents. Once again in this case, MP, outperforms the other approaches for all values of l.

Category	SVM	TSVM	SGT	SQ-Loss-I	MP
earn	91.3	95.4	90.4	96.3	**97.9**
acq	67.8	76.6	91.9	90.8	**97.2**
money	41.3	60.0	65.6	57.1	**73.9**
grain	56.2	**68.5**	43.1	33.6	41.3
crude	40.9	**83.6**	65.9	74.8	55.5
trade	29.5	34.0	36.0	**56.0**	47.0
interest	35.6	50.8	50.7	47.9	**78.0**
ship	32.5	46.3	**49.0**	26.4	39.6
wheat	47.9	44.4	59.1	58.2	**64.3**
corn	41.3	33.7	51.2	55.9	**68.3**
average	48.9	59.3	60.3	59.7	**66.3**

Table 7: P/R Break Even Points (PRBEP) for the top 10 categories in the Reuters data set with $l = 20$ and $u = 3299$. All results are averages over 20 randomly generated transduction sets. The last row is the macro-average over all the categories.

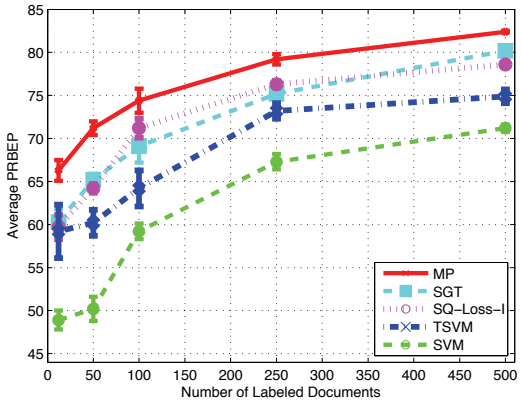


Figure 6: Average PRBEP over all classes vs. number of labeled documents (l) for Reuters data set

7.3.2 WEBKB COLLECTION

World Wide Knowledge Base (WebKB) is a collection of 8282 web pages obtained from four academic domains. The web pages in the WebKB set are labeled using two different polychotomies. The first is according to topic and the second is according to web domain. In our experiments we only considered the first polychotomy, which consists of 7 categories: *course*, *department*, *faculty*, *project*, *staff*, *student*, and *other*. Following Nigam et al. (1998) we only use documents from categories *course*, *department*, *faculty*, *project* which gives 4199 documents for the four categories. Each of the documents is in HTML format containing text as well as other information such as HTML tags, links, etc. We used both textual and non-textual information to construct the feature vectors. In this case we did not use either stop-word removal or stemming as this has been found to hurt performance on this task (Nigam et al., 1998). As in the case of the Reuters data set we extracted TFIDF features for each document and constructed the graph using cosine similarity.

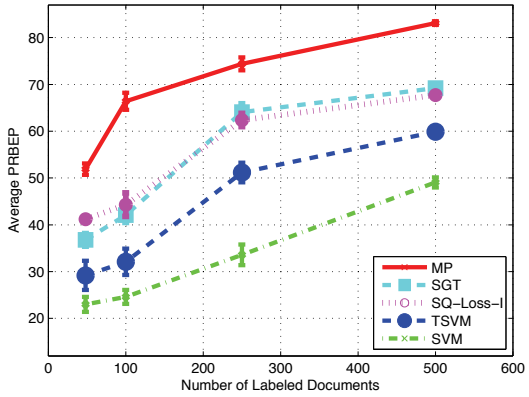


Figure 7: Average PRBEP over all classes vs. number of labeled documents (l) for WebKB collection.

Class	SVM	TSVM	SGT	SQ-Loss-I	MP
course	46.5	43.9	29.9	45.0	**67.6**
faculty	14.5	31.2	**42.9**	40.3	42.5
project	15.8	17.2	17.5	27.8	**42.3**
student	15.0	24.5	56.6	51.8	**55.0**
average	23.0	29.2	36.8	41.2	**51.9**

Table 8: P/R Break Even Points (PRBEP) for the WebKB data set with $l = 48$ and $u = 3148$. All results are averages over 20 randomly generated transduction sets. The last row is the macro-average over all the classes

As in Bekkerman et al. (2003), we created four roughly-equal random partitions of the data set. In order to obtain $\mathcal{D}_l$, we first randomly choose a split and then sampled l documents from that split. The other three splits constitute $\mathcal{D}_u$. We believe this is more realistic than sampling the labeled web-pages from a single university and testing web-pages from the other universities (Joachims, 1999). This method of creating transduction sets allows us to better evaluate the generalization performance of the various algorithms. Once again we create 21 transduction sets and the first set was used to tune the hyper-parameters. Further, we ran a search over the same grid as used in the case of Reuters. We report precision-recall break even point (PRBEP) results on the 3,148 test documents in Table 8. For this task, we found that the optimal value of the hyper-parameter were: in the case of SQ-Loss-I, $k = 1000$; in case of AM, $k = 1000$, $\mu =$ 1e–2, $\nu =$ 1e–4; and in case of SGT, $k = 100$, $c = 3200$. Once again, MP significantly outperforms the state-of-the-art (results are significant at the 0.0001 level). Figure 7 shows the variation of PRBEP with number of labeled documents (l) and was generated in a similar fashion as in the case of the Reuters data set.

7.4 TIMIT Phone Recognition

The TIMIT corpus of read speech was designed to provide speech data for acoustic-phonetic studies and for the development and evaluation of automatic speech recognition systems (Zue et al., 1990). TIMIT contains broadband recordings of 630 speakers of eight major dialects of American English, each reading ten phonetically rich sentences. The corpus includes time-aligned phonetic transcriptions and has standard training (3896 utterances) and test (196 utterances) sets. For hyper-parameter tuning, as TIMIT does not define a development set, we created one with 50 TIMIT utterances (independent of the training and test sets). In the past, TIMIT has been used almost exclusively to evaluate the performance of supervised learning algorithms (Halberstadt and Glass, 1997; Somervuo, 2003). Here, we use it to evaluate SSL algorithms by using fractions of the standard TIMIT training set obtained by random sampling. This simulates the case when only small amounts of data are labeled. We compare the performance of MP against that of

(a) ℓ_2 regularized 2-layer multi-layered perceptron (MLP) (Bishop, 1995), and

(b) $\mathcal{P}_{SQ}$ solved using SQ-Loss-I.

Recall that, while MLPs are fully-supervised, SQ-Loss-I and MP are both graph-based SSL algorithms. We choose ℓ_2 regularized MLPs as they have been shown to beat SVMs for the phone classification task (Li and Bilmes, 2006).

To obtain the acoustic observations, $\mathbf{x}_i$, the signal was first pre-emphasized ($\alpha = 0.97$) and then windowed using a Hamming window of size 25ms at 100Hz. We then extracted 13 mel-frequency cepstral coefficients (MFCCs) (Lee and Hon, 1989) from these windowed features. Deltas were appended to the above resulting in 26 dimensional features. As phone classification performance is improved by context information, we appended each frame with 3 frames from the immediate left and right contexts and used these 182 dimensional feature vectors as inputs to the classifier. These features were used to construct a symmetrized 10-NN graph over the entire training and development sets. This graph had 1,382,342 vertices. The weights are given by

$$w_{ij} = \text{sim}(\mathbf{x}_i, \mathbf{x}_j) = \exp\{-(\mathbf{x}_i - \mathbf{x}_j)^T \Sigma^{-1} (\mathbf{x}_i - \mathbf{x}_j)\}$$

where Σ is the covariance matrix computed over the entire TIMIT training set. We follow the standard practice of mapping the original set of 61 phones in TIMIT down to 48 phones for modeling ($|\mathrm{Y}| = 48$) and then a further mapping to 39 phones for scoring (Lee and Hon, 1989).

For each approach the hyper-parameters were tuned on the development set by running an extensive search. In the case of the MLP, the hyper-parameters include the number of hidden units and the regularization coefficient. For MP and SQ-Loss-I, the hyper-parameters were tuned over the following sets $\mu \in \{$1e–8, 1e–4, 0.01, 0.1$\}$ and $\nu \in \{$1e–8, 1e–6, 1e–4, 0.01, 0.1$\}$. We found that setting $\alpha = 1$ in the case of MP ensured that p = q at convergence. As both MP and SQ-Loss-I are transductive, in order to measure performance on an independent test set, we induce the labels using the Nadaraya-Watson estimator, that is, given an input sample, $\hat{\mathbf{x}}$, that we wish to classify, the output is given by

$$\hat{y} = \operatorname*{argmax}_{y \in \mathrm{Y}} \hat{p}(y) \text{ where } \hat{p}(y) = \frac{\sum_{j \in \mathcal{N}(\hat{\mathbf{x}})} \text{sim}(\hat{\mathbf{x}}, \mathbf{x}_j) p^*_j(y)}{\sum_{j \in \mathcal{N}(\hat{\mathbf{x}})} \text{sim}(\hat{\mathbf{x}}, \mathbf{x}_j)},$$

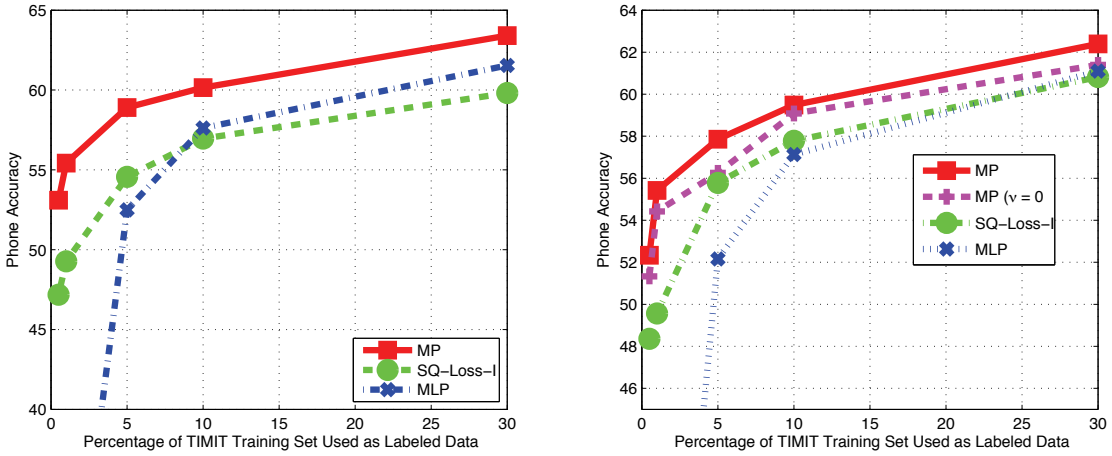


Figure 8: Phone Accuracy (PA) on the TIMIT development set (left) and TIMIT NIST core evaluation/test set (right). The x-axis shows the percentage of standard TIMIT training data that was treated as being labeled.

$\mathcal{N}(\hat{\mathbf{x}})$ is the set of nearest neighbors of $\hat{\mathbf{x}}$ in the training data (i.e., all the samples over which the graph was constructed) and p_j^* is the converged value of p_j. In our experiments we have that $|\mathcal{N}(\hat{\mathbf{x}})| = 50$.

The left plot in Figure 8 shows the phone classification results on the TIMIT development set while the right plot shows the results on the NIST Core test set. The y-axis shows phone accuracy (PA) which represents the percentage of frames correctly classified and the x-axis shows the fraction f of the training set that was treated as being labeled. We show results for $f \in \{0.005, 0.05, 0.1, 0.25, 0.3\}$. Note that in each case we use the same graph, that is, only the set of labeled vertices V_l changes depending on f. The following observations may be made from these results:

1. MP outperforms the SQ-Loss-I objective for all cases of f. This lends further weight to the claim that KLD based loss is more suitable for classification problems.

2. When little labeled training data is available, both SQ-Loss-I and MP significantly outperform the MLP. For example when 0.5% of the training set is labeled, the PA in the case of MP was 52.3% while in the MLP gave a PA of 19.6%. This is not surprising as the MLP does not make use of the unlabeled data. It remains to be tested if semi-supervised MLP training (Malkin et al., 2009) would reduce or reverse this difference.

3. Even when 10% of the original TIMIT training set is used, MP gives a PA of about 60% and outperforms both the MLP and SQ-Loss-I.

4. It is interesting to note that an MLP trained using the entire training set (i.e., it had 100% of the labeled samples) resulted in a PA of 63.1%. But using about 30% of this data, MP gives a PA of about 62.4%.

	Bigram	Trigram
SQ-Loss-I	75.6%	76.9%
MP	81.0%	81.9%

Table 9: Dialog-Act Tagging Accuracy results on the Dihana Corpus. The results are for the case of classifying user turns. The baseline DA accuracy was 76.4% (Martínez-Hinarejos et al., 2008)

5. We also found that for larger values of f (e.g., at $f = 1$), the performances of MLP and MP did not differ significantly. But those are more representative of the supervised training scenarios which is not the focus here.

6. A comparison of the curves for MP with and without entropy regularization illustrates the importance of the graph-regularizer (second term in C_{KL} and C_{MP}).

7.5 Dialog-Act Tagging

Discourse patterns in natural conversations and meetings are well known indicators of interesting and useful information about human conversational behavior. Dialog acts (DA) which reflect the functions that utterances serve in discourse are one type of such patterns. Detecting and understanding dialog act patterns can provide benefit to systems such as automatic speech recognition, machine translation and general natural language processing (NLP). In this section we present dialog-act tagging results on two tasks: (a) Dihana, and (b) SWB.

7.5.1 DIHANA DA TAGGING

Dihana is a Spanish dialog corpus. It is composed of 900 task-oriented computer-human spoken dialogs collected via a train reservation system. Typical topics include timetables, fares, and services offered on trains. The size of the vocabulary is 823 words. Dihana was acquired from 225 different speakers (153 male and 72 female). On average, each dialog consisted of 7 user turns and 10 system turns, with an average of 7.7 words per user turn. The corpus has three tasks which include classifying the DAs of the (a) user turns, (b) system turns, and (c) both user and system turns. Each of these tasks has training, test and development sets setup for 5-fold cross validation. As the system turns are more structured compared to the user turns, the task of classifying user turns is more challenging. For more information, see Martínez-Hinarejos et al. (2008).

Here we compare the performance of MP against that of SQ-Loss-I and a HMM-based DA tagging system described in Martínez-Hinarejos et al. (2008). We extracted two sets of features from the text: (a) bigram TFIDF and (b) trigram TFIDF (Salton and Buckley, 1987). We constructed symmetrized k-NN graphs using each of the above features making use of cosine similarity. The graphs were defined over the training, test and development sets for the task that involved classifying user turns. The hyper-parameters were tuned over $k \in \{2, 10, 20, 50, 100\}$, $\mu \in \{$1e–8, 1e–4, 0.01, 0.1, 1, 10, 100$\}$ and $\nu \in \{$1e–8, 1e–6, 1e–4, 0.01, 0.1$\}$ on the development set. In the case of MP, we found that setting $\alpha = 2$ gave p = q at convergence.

The DA tagging results averaged over the 5-folds for the Dihana corpus are shown in Table 9. Unlike previous experiments, in this case, we treat the entire training set as being labeled, whereas

	Bigram	Trigram
SQ-Loss-I	79.1%	81.3%
MP	83.2%	85.6%

Table 10: Dialog-Act Tagging Accuracy results on the Switchboard DA Corpus. The baseline DA accuracy was 84.2% (Ji and Bilmes, 2005)

the test set is unlabeled. This simulates the case when SSL algorithms are used for supervised learning but in a transductive manner (i.e., the test set is assumed to be given). The HMM-based DA tagger which was trained on the same set gave an accuracy of 76.4%. It can be seen from Table 9 that MP outperforms both SQ-Loss-I and the HMM based tagger in both the bigram-TFIDF and trigram-TFIDF cases. We conjecture that the performance improvement of MP over HMM is due to two reasons: (a) MP is a discriminative model while the HMM was trained in a generative fashion, (b) as MP is transductive, it is able to exploit the knowledge of the graph over the test set.

7.5.2 SWITCHBOARD DA TAGGING

The goal of the Switchboard discourse language modeling project was to annotate the utterances in the Switchboard-I (SWB) training set with their corresponding discourse acts (Jurafsky and Ess-Dykema, 1997). SWB is a collection of telephone conversations (see Section 8.1). Every utterance in a each conversation was given one of the 42 different dialog act tags (see Table 2 in Jurafsky and Ess-Dykema, 1997). For our work here we only use the 11 most frequent tags. This covers more than 86% of all the utterances in SWB. These utterances were split into training, development and test sets containing 180314, 5192 and 4832 utterances respectively.

As in the case of Dihana, we generated both bigram and trigram TFIDF features and constructed graphs in the manner described above. Here we compare the performance of MP and SQ-Loss-I against the performance of a parametric dynamic Bayesian Network (DBN) that makes use of a hidden back-off model (Ji and Bilmes, 2005). The DBN, however, made use of only bigram features. The hyper-parameters were tuned over $k \in \{2, 10, 20\}$, $\mu \in \{$1e–8, 1e–4, 0.01, 0.1, 1, 10, 100$\}$ and $\nu \in \{$1e–8, 1e–6, 1e–4, 0.01, 0.1$\}$ on the development set. In the case of MP, we found that setting $\alpha = 2$ ensured that $p = q$ at convergence.

The test set DA tagging accuracy is shown in Table 10. We see that when we use trigram TFIDF features, MP outperforms the bigram DBN. More importantly, it performs better than SQ-Loss-I in all cases.

8. Parallelism and Scalability to Large Data Sets

In this section we discuss how MP can be scaled to very large data sets. We use the Switchboard I (SWB) data set which is a collection of about 2,400 two-sided telephone conversations among 543 speakers (302 male, 241 female) from all areas of the United States (Godfrey et al., 1992). A computer-driven system handled the calls, giving the caller appropriate recorded prompts, selecting and dialing another person (the callee) to take part in a conversation, introducing a topic for discussion and recording the speech from the two subjects into separate channels until the conversation was finished. SWB is very popular in the speech recognition community and is used almost ubiq-

uitously for the training of large vocabulary conversational speech recognition systems (Evermann et al., 2005; Subramanya et al., 2007) and consists of about 300 hours of speech data.

In order to construct a graph using the SWB data, we exact features $\mathbf{x}_i$ in the following manner—the wave files were first segmented and then windowed using a Hamming window of size 25ms at 100Hz. We then extracted 13 perceptual linear prediction (PLP) coefficients from these windowed features and appended both deltas and double-deltas resulting in a 39 dimensional feature vector. As phone classification performance is improved by context, we used a 7 frame context window (3 frames in the past and 3 in the future) yielding a 273 dimensional $\mathbf{x}_i$. This procedure resulted in 120 million samples.

Due to the large size $m = 120M$ of the SWB data set, it is not currently feasible to generate the graph using the conventional brute-force search which is $O(m^2)$. Nearest neighbor search is a well researched problem with many approximate solutions. A large number of solutions to this problem are based on variations of the classic *kd-tree* data structure (Friedman et al., 1977). Here we make use of the Approximate Nearest Neighbor (ANN) library (see `http://www.cs.umd.edu/~mount/ANN/`) (Arya and Mount, 1993; Arya et al., 1998). It constructs a modified version of the kd-tree data structure which is then used to query the NNs. The query process requires that one specify an error term, ε, and guarantees that

$$\frac{d(\mathbf{x}_i, \mathcal{N}(\mathbf{x}_i))}{d(\mathbf{x}_i, \mathcal{N}_\varepsilon(\mathbf{x}_i))} \leq 1 + \varepsilon$$

where $\mathcal{N}(\mathbf{x}_i)$ is a function that returns the actual NN of $\mathbf{x}_i$ while $\mathcal{N}_\varepsilon(\mathbf{x}_i)$ returns the approximate NN. Larger values of ε improve the speed of the nearest neighbor search at the cost of accuracy. For more details about the algorithm, see Arya and Mount (1993); Arya et al. (1998). In our case we constructed a symmetrized 10-NN graph with $\varepsilon = 2.0$.

Next we describe how MP can be parallelized on a shared-memory symmetric multiprocessor (SMP). The update equations in the case of MP are amenable to a parallel implementation and also to further optimizations that lead to a near linear speedup. In the MP update equations (see Section 3), we see that one set of measures is held fixed while the other set is updated without any required communication amongst set members, so there is no write contention. This immediately yields a T-threaded implementation where the graph is evenly T-partitioned and each thread operates over only a size $m/T = (l+u)/T$ subset of the graph nodes.

We used the graph constructed using the SWB data above and ran a timing test on a 16 core symmetric multiprocessor with 128GB of RAM, each core operating at 1.6GHz. We varied the number T of threads from 1 (single-threaded) up to 16, in each case running 3 iterations of MP (i.e., 3 each of p and q updates). Each experiment was repeated 10 times, and we measured the minimum CPU time over these 10 runs. CPU time does not include the time taken to load data-structures from disk. The speedup for T threads is typically defined as the ratio of time taken for single thread to time taken for T threads. The solid (black) line in Figure 9(a) represents the ideal case (a linear speedup), that is, when using T threads results in a speedup of T. The pointed (green) line shows the actual speedup of the above procedure, typically less than ideal due to inter-process communication and poor shared L1 and/or L2 microprocessor cache interaction. When $T \leq 4$, the speedup (green) is close to ideal, but for increasing T the algorithm increasingly falls away from the ideal case. Note that in the figure (and henceforth) we refer to the green pointed line as 'speech temporal ordering' as the nodes in the graph are ordered based on the sequence in which they occur in the utterance.

Our contention is that the sub-linear speedup is due to the poor cache cognizance of the algorithm. At a given point in time, suppose thread $t \in \{1,\ldots,T\}$ is operating on node i_t. The collective set of neighbors that are being used by these T threads are $\{\cup_{t=1}^{T}\mathcal{N}(i_t)\}$ and this, along with nodes $\cup_{t=1}^{T}\{i_t\}$ (and all memory for the associated measures), constitute the current *working set*. The working set should be made as small as possible to increase the chance it will fit in any shared machine caches, but this becomes decreasingly likely as T increases since the working set is monotonically increasing with T. Our goal, therefore, is for the nodes that are being simultaneously operated on to have a large amount of neighbor overlap thus minimizing the working set size. Viewed as the optimization problem, we must find a partition $(V_1,V_2,\ldots,V_{m/T})$ of V that minimizes $\max_{j\in\{1,\ldots,m/T\}}|\cup_{v\in V_j}\mathcal{N}(v)|$. With such a partition, we may also order the subsets so that the neighbors of V_i would have maximal overlap with the neighbors of V_{i+1}. We then schedule the T nodes in V_j to run simultaneously, and schedule the V_j sets successively.

Algorithm 1: Graph Node Ordering Algorithm Pseudocode, SMP Case

Input: A Graph $G = (V,E)$
Result: A node ordering, by when they are marked.
Select an arbitrary node v ;
while *There are unselected nodes remaining* **do**
  Select an unselected $v' \in \mathcal{N}^2(v)$ that maximizes $|\mathcal{N}(v)\cap\mathcal{N}(v')|$. If the intersection is empty, select an arbitrary unselected v'. ;
  Mark v' as selected.; `// v' is next node in the order`
  $v \leftarrow v'$. ;

Of course, the time to produce such a partition cannot dominate the time to run the algorithm itself. Therefore, we propose a simple fast node ordering procedure (Algorithm 1) that can be run once before the parallelization begins. The algorithm orders the nodes such that successive nodes are likely to have a high amount of neighbor overlap with each other and, by transitivity, with nearby nodes in the ordering. It does this by, given a node v, choosing another node v' (from amongst v's neighbors' neighbors, meaning the neighbors of v's neighbors) that has the highest neighbor overlap. We need not search all V nodes for this, since anything other than v's neighbors' neighbors has no overlap with the neighbors of v. Given such an ordering, the t^{th} thread operates on nodes $\{t,t+m/T,t+2m/T,\ldots\}$. If the threads proceed synchronously (which we do not enforce) the set of nodes being processed at any time instant are $\{1+jm/T,2+jm/T,\ldots,T+jm/T\}$. This assignment is beneficial not only for maximizing the set of neighbors being simultaneously used, but also for successive chunks of T nodes since once a chunk of T nodes have been processed, it is likely that many of the neighbors of the next chunk of T nodes will already have been pre-fetched into the caches. With the graph represented as an adjacency list, and sets of neighbor indices sorted, our algorithm is $O(mk^3)$ in time and linear in memory since the intersection between two sorted lists may be computed in $O(k)$ time. This is typically even better than $O(m\log m)$ since $k^3 < \log m$ for large m.

We ordered the SWB graph nodes, and ran timing tests as explained above. The CPU time required for the node ordering step is included in each run along with the time for MP. The results are shown in Figure 9(a) (pointed red line) where the results are much closer to ideal, and there are no obvious diminishing returns like in the unordered case. Running times are given in Figure 9(b).

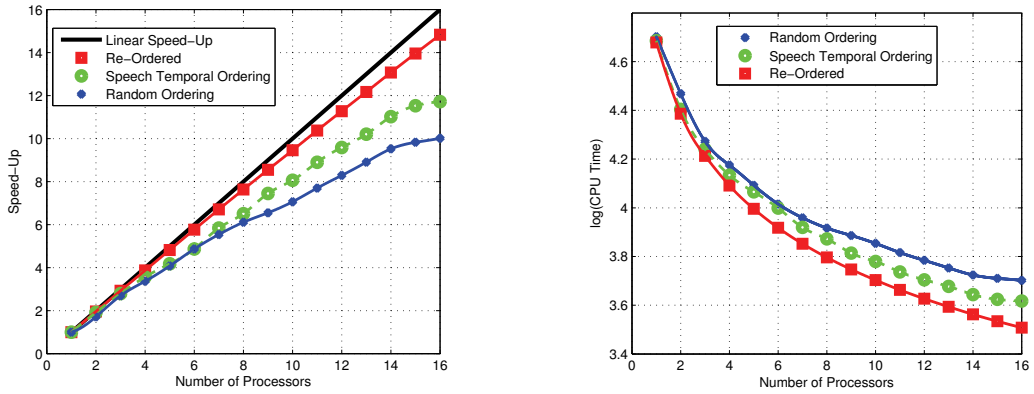


Figure 9: (a) speedup vs. number of threads for the SWB graph (see Section 7). The process was run on a 128GB, 16 core machine with each core at 1.6GHz. (b) The actual CPU times in seconds on a *log scale* vs. number of threads for with and without ordering cases. "Random" corresponds to the case where we choose a random unselected node rather than the one with maximum overlap (see Algorithm 1).

Moreover, the ordered case showed better performance even for a single thread $T = 1$. Note that since we made use of speech data to generate the graph, it is already naturally well-ordered by time. This is because human speech is a slowly changing signal, so the nodes corresponding to consecutive frames are similar, and can be expected to have similar neighbors. Therefore, we expect our "baseline" speech graph to be better than an arbitrary order, one that might be encountered in a different application domain. In order to measure performance for such arbitrarily ordered graphs, we took the original graph and reordered uniformly at random (a uniform node shuffle). We ran timing experiments on the resulting graph and the results are shown in Figure 9 as "Random". As can be seen, there is indeed a benefit from the speech order, and relative to this random baseline, our node ordering heuristic improves machine efficiency quite significantly.

We conclude this section by noting that (a) re-ordering may be considered a pre-processing (offline) step, (b) the SQ-Loss algorithm may also be implemented in a multi-threaded manner and this is supported by our implementation, (c) our re-ordering algorithm is general and fast and can be used for any graph-based algorithm where the iterative updates for a given node are a function of its neighbors (i.e., the updates are harmonic w.r.t. the graph Zhu et al., 2003), and (d) while the focus here was on parallelization across different processors on a SMP, a similar approach also applies for distributed processing across a network with a shared disk (Bilmes and Subramanya, 2011).

8.1 Switchboard Phonetic Annotation

In this section we consider how MP can be used to annotate the SWB data set. Recall that SWB consists of 300 hours of speech with word-level transcriptions. In addition, less reliable phone level annotations generated in an automatic manner by a speech recognizer with a non-zero error rate are also available (Deshmukh et al., 1998). The *Switchboard Transcription Project* (STP) (Greenberg, 1995) was undertaken to accurately annotate SWB at the phonetic and syllable levels. One of the goals was that such data could then be used to improve the performance of conversational speech

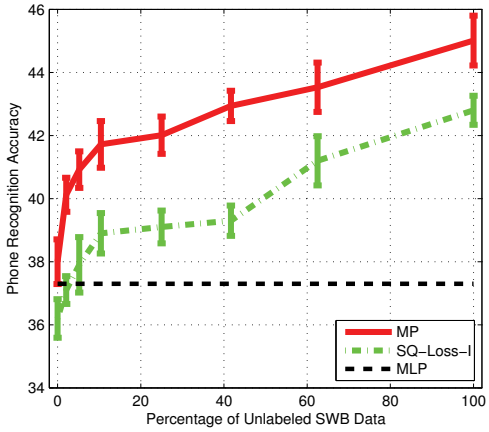


Figure 10: Phone Accuracy vs. Percentage of switchboard (SWB) I training data. STP portion of SWB was excluded. Phone Accuracy was measured on the STP data. Note that when all the Switchboard I data was added, the resulting graph had **120 million** vertices. The dashed black line shows the performance of a MLP measured using the $s = 0\%$ case over the same training, development and test sets as MP and LP.

recognition systems. As the task was time-consuming, costly, and error-prone, only 75 minutes of speech segments selected from different SWB conversations were annotated at the phone level and about 150 minutes annotated at the syllable level. Having access to such annotations for all of SWB could be useful for large vocabulary speech recognition research and speech science research in general. Thus, this an ideal real-world task for SSL.

For our experiments here we only make use of the phonetic labels ignoring the syllable annotations. Our goal here is two-fold: (a) treat the phonetically annotated portion of STP as labeled data and use it to annotate all of SWB in STP style, that is, at the phonetic level, yielding the S3TP corpus and (b) show that our approach scales to very large data sets.

We randomly split the 75 minute phonetically annotated part of STP into three sets, one each for training, development and testing containing 70%, 10% and 20% of the data respectively (the size of the development set is considerably smaller than the size of the training set). This procedure was repeated 10 times (i.e., we generated 10 different training, development and test sets by random sampling). In each case, we trained a phone classifier using the training set, tuned the hyper-parameters on the development set and evaluated the performance on the test set. In the following, we refer to SWB that is not a part of STP as *SWB-STP*. We added the unlabeled SWB-STP data in stages. The percentage, s, of unlabeled data included, 0%, 2%, 5%, 10%, 25%, 40%, 60%, and 100% of SWB-STP. We ran both MP and SQ-Loss-I in each case. When $s = 100\%$, there were about 120 million nodes in the graph. As far as we know, this is by far the largest (by about two orders of magnitude) size graph ever reported for an SSL procedure.

We constructed graphs using the STP data and $s\%$ of (unlabeled) SWB-STP data following the recipe described in the previous section. For all the experiments here we used a symmetrized 10-NN graph and $\varepsilon = 2.0$. The labeled and unlabeled points in the graph changed based on training, development and test sets used. In each case, we ran both the MP and SQ-Loss-I objectives. For each

set, we ran a search over $\mu \in \{1e{-}8, 1e{-}4, 0.01, 0.1\}$ and $\nu \in \{1e{-}8, 1e{-}6, 1e{-}4, 0.01, 0.1\}$ for both the approaches. The best value of the hyper-parameters were chosen based on the performance on the development set and the same value was used to measure the accuracy on the test set. The mean phone accuracy over the different test sets (and the standard deviations) are shown in Figure 10 for the different values of s. We would like to point out that our results at $s = 0\%$ outperform the state-of-the-art. As a reference, at $s = 0\%$, an ℓ_2 regularized MLP with a 9 frame context window gave a mean phone accuracy of 37.2% and standard deviation of 0.83 (note that this MLP was trained fully-supervised). Phone classification in the case of conversational speech is a much harder task compared to phone classification of read speech (Morgan, 2009). It can be seen that MP outperforms SQ-Loss-I in all cases. More importantly, we see that the performance on the STP data improves with the addition of increasing amounts of unlabeled data, and MP seems to get a better benefit with this additional unlabeled data, although even SQ-Loss-I has not reached the point where unlabeled data starts becoming harmful (Nadler et al., 2010).

9. Discussion

In this section, we discuss possible extensions of the proposed approach.

9.1 Generalizing Graph-based Learning via Bregman Divergence

Given a strictly convex real-valued function $\phi : \triangle \rightarrow \mathbb{R}$, the Bregman divergence $\mathbf{B}_\phi(\psi_1 || \psi_2)$ between two measures $\psi_1, \psi_2 \in \triangle$ is given by Lafferty et al. (1997)

$$\mathbf{B}_\phi(\psi_1 || \psi_2) \triangleq \phi(\psi_1) - \phi(\psi_2) - \langle \nabla\phi(\psi_2), \psi_1 - \psi_2 \rangle.$$

It can be shown that a number of popular distance measures, such as Euclidean distance, KLD, Itakura-Satio distance are special cases of Bregman divergence (Banerjee et al., 2005). Consider the optimization problem $\mathcal{P}_{BR} : \min_{\mathrm{p} \in \triangle^m} \mathcal{C}_{BR}(\mathrm{p})$ where

$$\mathcal{C}_{BR}(\mathrm{p}) = \sum_{i=1}^{l} \mathbf{B}_\phi\big(r_i || p_i\big) + \mu \sum_{i=1}^{m} \sum_{j \in \mathcal{N}(i)} w_{ij} \mathbf{B}_\phi\big(p_i || p_j\big) + \nu \sum_{i=1}^{m} \mathbf{B}_\phi(p_i || u).$$

When $\mathbf{B}_\phi(p||q)$ is convex in the pair (p,q) (Banerjee et al., 2005), $\mathcal{C}_{BR}$ is also convex. Clearly $\mathcal{C}_{BR}$ is a valid graph-based learning objective and it can be seen that it generalizes objectives based on both squared loss ($\phi = \sum_y p^2(y)$) and KLD based loss ($\phi = \sum_y p(y) \log p(y)$). While in the case graph Laplacian-based techniques, one can generate a large family of regularizers by iterating the Laplacian or taking various transformations of its spectrum to create new ways of measuring smoothness on the graph, here in the Bregman case, the same can be achieved by using different ϕ's.

The graph regularizer is central to any graph-based SSL algorithm, and there are two factors that effect this regularizer: (a) the graph weights and (b) the loss function used to measure the disparity between the distributions. In the cases we have discussed thus far, the loss function has been either based on squared-error or KLD. Further, while there have been efforts in the past to learn the graph (and thus the graph weights) (Zhu and Ghahramani, 2002a; Zhang and Lee, 2006; Zhu et al., 2005; Alexandrescu and Kirchhoff, 2007a), to the best of our knowledge, there has been no efforts directed towards learning the loss function. So the natural question is whether it is possible to learn ϕ jointly with p?

One simple idea would be to set $\phi = \sum_y \left(\lambda p^2(y) + (1-\lambda) p(y) \log p(y)\right)$ which leads to a combination of the popular squared loss and proposed KLD based loss objectives (henceforth we refer to this as $\mathcal{C}'_{BR}(\mathrm{p}, \lambda)$). We then need to learn λ jointly with p. However, directly minimizing $\mathcal{C}'_{BR}$ w.r.t. both p and λ will always leads to $\lambda^* = 1$ as KLD is lower bounded by squared loss (by Pinsker's inequality). Thus other criteria such as those based on minimizing the leave-one-out error (Zhang and Lee, 2006) or minimum description length may be required. There might also be other convex parametrization of ϕ. This would amount to learning the loss function while the actual graph weights are held fixed.

While we have defined Bregman divergence over simplices, they are actually quite general and can be defined over other general sets of objects such as vectors or matrices (Tsuda et al., 2005). This can be used to solve general learning problems using alternating-minimization using a reformulation similar to the one suggested in Section 4. We believe that this is another contribution of our work here as our proposed objective, and the use of alternating-minimization to efficiently optimize it are in fact very general and can be used to solve other learning problems (Tsuda et al., 2005).

9.2 Incorporating Priors

As discussed in Section 1, there are two types of priors in SSL—label priors and balance priors. They are useful in the case of imbalanced data sets. We have seen that MP is less sensitive to imbalance compared to other graph-based SSL approaches (see the results in the cases of two-moon, USPS, Reuters, TIMIT and SWB data sets). However, in cases of extreme imbalance, even the performance of MP might suffer and so we show how to modify our proposed objective to handle both the above priors in a principled manner. Label priors are useful when the underlying data set is imbalanced. For example, in the case of phone classification, as a result of the nature of human speech and language production, some classes of sounds tend to occur at a higher rate compared to others. Clearly ignoring such domain knowledge can hurt performance particularly in the case of SSL where labeled data is sparse. On the other hand, balance priors are useful to prevent degenerate solutions. An extreme example of a degenerate solution would be all unlabeled samples being classified as belonging to the same class when the underlying data set has a uniform prior. This can occur due to a number of reasons such as, (a) improper graph construction, (b) improperly sampled labeled data, that is, the case where a majority of the labeled samples come from one class (similar to the scenario discussed in the case of the 2D two-moon data set).

Label Priors: This is more akin to the classical integration of priors within a Bayesian learning setting. There has been some work in the past directed towards integrating priors for parametric (non-graph-based) SSL (Mann and McCallum, 2007). In the case of graph-based SSL, class mass normalization (CMN) (Zhu and Ghahramani, 2002a; Bengio et al., 2007) and label bidding (Zhu and Ghahramani, 2002a), are the two approaches that have been used to-date. However, these are applicable only after the inference process has converged. In other words, they represent ways in which the posteriors may be influenced so that the average probability mass over all the posteriors for a given class matches that given by the prior. Ideally, like in general Bayesian learning, it is imperative that the priors are tightly integrated in to the inference process rather than influencing the results at a later point. Our proposed objective can be extended to incorporate label priors. We first remind the reader that $\mathcal{C}_{KL}(\mathrm{p})$ may be re-written as $\mathcal{C}_{KL}(\mathrm{p}) = \sum_{i=1}^{l} D_{KL}\big(r_i||p_i\big) + \mu \sum_{i,j} w_{ij} D_{KL}\big(p_i||p_j\big) + \nu \sum_i D_{KL}\big(p_i||u\big)$ where u is uniform measure.

Now consider minimizing over $\mathrm{p} \in \triangle^m$

$$C'_{KL}(\mathrm{p}) = \sum_{i=1}^{l} D_{KL}\big(r_i||p_i\big) + \mu \sum_{i,j} w_{ij} D_{KL}\big(p_i||p_j\big) + \nu \sum_{i} D_{KL}\big(p_i||p_0\big).$$

The above objective is convex and the last term encourages each p_i to be close to p_0 without actually insisting that $p_i(y) = p_0(y)\ \forall\ i, y$. It is possible to reformulate the above objective as

$$C'_{MP}(\mathrm{p},\mathrm{q}) = \sum_{i=1}^{l} D_{KL}\big(r_i||q_i\big) + \mu \sum_{i,j} w'_{ij} D_{KL}\big(p_i||q_j\big) + \nu \sum_{i} D_{KL}\big(p_i||p_0\big).$$

which can be easily solved using AM. Further each of the update equations has a closed form solution. This represents the case where the prior effects each vertex directly (i.e., a more local influence) .

Balance Priors: There has been some work in graph-based SSL for incorporating balance priors. SGT (Joachims, 2003) which is an approximation to the NP-hard norm cut problem attempts to incorporate priors by influencing the nature of the final cut. But there are other drawbacks associated with SGT such as computational complexity. We can incorporate a balance term in our objective by first defining $\tilde{p}(y)$ as the agglomerative measure over all the p's and then minimizing

$$C'_1(\mathrm{p}) = C_{KL}(\mathrm{p}) + \kappa D_{KL}(p_0||\tilde{p})$$

where $p_0(\mathrm{y})$ is the prior probability that $\mathrm{Y} = \mathrm{y}$. The above retains the nice convexity properties of the original objective. There are many ways of defining $\tilde{p}$, such as,

$$\tilde{p}(y) = \frac{1}{n} \sum_{i=1}^{n} p_i(y) \text{ or } \tilde{p}(y) \propto \prod_{i=1}^{n} (p_i(y) + \varepsilon).$$

The first case above represents the arithmetic mean while the second one is the geometric mean. Here the prior only indirectly influences the individual p's, that is, via $\tilde{p}$. Unfortunately, this form cannot be optimized in the closed form using alternating-minimization. However, the MOM approach proposed in Section 3.1 or IPMs or any other numerical convex optimization approach may be used to solve the above problem.

9.3 Directed Graphs

In some applications, the graphs are directed in nature. Examples include the Internet (a vertex might represent a web-page and directed links for hyper-links between pages), or a graph representing the routes taken by a delivery system. In such applications there is useful information that is expressed by the direction of the connection between two vertices. While we could convert any given directed graph into an undirected one, SSL algorithms in this case should exploit the information in the directed links. Thus far we have been using symmetrized k-NN graphs, but without the symmetrization step, k-NN graphs are not necessarily symmetric.

As KLD is an asymmetric measure of dissimilarity between measures, our proposed objective can very easily be extended to work for directed graphs. Note that, as in the case of an undirected graph, a directed graph can also be represented as a matrix $\mathbf{W}$, but here the matrix is asymmetric. There has been some work on graph-based SSL using directed graphs. For example, Zhou et al.

(2005), use a squared-loss based objective on directed graphs. We believe that this may not be ideal, as squared error is symmetric and as a result it might be difficult to fully exploit the information encoded by the directed links. An asymmetric measure of dissimilarity would have a better chance of correctly representing the problem of SSL on directed graphs.

It turns our that $\mathcal{C}_{MP}$ may be modified for directed graphs. We assume that if j is a NN of i then there is a directed arrow from i to j. There are in fact two scenarios that one needs to consider. Given a node $i \in V$, let $\mathcal{N}^{(in)}(i)$ be the set of nodes that have directed edges that lead into vertex i. Consider the following objective

$$\mathcal{C}_{MP}^{(D1)}(\mathrm{p},\mathrm{q}) = \sum_{i=1}^{l} D_{KL}\big(r_i||p_i\big) + \mu \sum_{i=1}^{m} \sum_{j \in \mathcal{N}^{(in)}(i)} w_{ij} D_{KL}\big(p_i||q_j\big) - \nu \sum_{i=1}^{m} H(p_i).$$

In this case, for node i, the second term in the above objective encourages p_i to be close to the q's of all its neighbors, $\mathcal{N}^{(in)}(i)$. In other words, the above form expresses the rule "each vertex should resemble its neighbors but not necessarily vice-versa." In a similar manner we can define a complementary form—let $\mathcal{N}^{(out)}(i)$ be the set of nodes which are on the other end of out-going links from node $i \in V$. Consider minimizing

$$\mathcal{C}_{MP}^{(D2)}(\mathrm{p},\mathrm{q}) = \sum_{i=1}^{l} D_{KL}\big(r_i||p_i\big) + \mu \sum_{i=1}^{m} \sum_{j \in \mathcal{N}^{(out)}(i)} w_{ij} D_{KL}\big(p_i||q_j\big) - \nu \sum_{i=1}^{m} H(p_i).$$

This form encourages, "the neighbors of a vertex should resemble it but not necessarily vice-versa."

Both $\mathcal{C}_{MP}^{(D1)}(\mathrm{p},\mathrm{q})$ and $\mathcal{C}_{MP}^{(D2)}(\mathrm{p},\mathrm{q})$ can be efficiently optimized using our alternating-minimization (the update equations are similar to MP). In a similar manner as the above, our objective can also be easily extended to hyper-graphs.

9.4 Connections to Entropy Minimization (Grandvalet and Bengio, 2005)

Entropy Minimization uses the entropy of the unlabeled data as a regularizer while optimizing a parametric loss function over the labeled data. The loss function here is given by

$$C(\Theta) = -\sum_{i=1}^{l} \log p(y_i|x_i;\Theta) + \nu \sum_{i=l+1}^{l+u} H(Y_i|X_i;\Theta)$$

where $H(Y_i|X_i;\Theta)$ is the Shannon entropy of the probability distribution $p(y_i|x_i;\Theta)$. While both our proposed approach and entropy minimization make use of the Shannon entropy as a regularizer, there are several important differences between the two approaches:

1. entropy minimization is not graph-based,
2. entropy minimization is parametric whereas our proposed approach is non-parametric
3. the objective in case of entropy minimization is not convex, whereas in our case we have a convex formulation with simple update equations and convergence guarantees.
4. most importantly, entropy minimization attempts to minimize entropy while the proposed approach aims to maximize entropy. While this may seem a triviality, it has significant consequences on the optimization problem.

It is however possible to derive an interesting relationship between the proposed objective and entropy minimization. Consider

$$\begin{aligned}\mathcal{C}_{KL}(\mathrm{p}) &= \sum_{i=1}^{l} D_{KL}\big(r_i||p_i)\big) + \mu \sum_{i,j=1}^{n} w_{ij} D_{KL}\big(p_i||p_j\big) - \nu \sum_{i=1}^{n} H(p_i) \\ &\leq \sum_{i=1}^{l} D_{KL}\big(r_i||p_i)\big) - \mu \sum_{i,j=1}^{n} w_{ij} \sum_{y} p_i(y) \log p_j(y)\end{aligned}$$

as $w_{ij}, \nu, H(p_i) \geq 0$. Consider a degenerate graph in which $w_{ij} = \delta(i = j \wedge i > l)$ then

$$\begin{aligned}\mathcal{C}_{KL}(\mathrm{p}) &\leq \sum_{i=1}^{l} D_{KL}\big(r_i||p_i)\big) - \mu \sum_{i=l+1}^{n} \sum_{y} p_i(y) \log p_i(y) \\ &= \sum_{i=1}^{l} \sum_{y} \Big(r_i(y) \log r_i(y) - r_i(y) \log p_i(y) \Big) + \mu \sum_{i=l+1}^{n} H(p_i) \\ &\leq -\sum_{i=1}^{l} \sum_{y} r_i(y) \log p_i(y) + \mu \sum_{i=l+1}^{n} H(p_i).\end{aligned}$$

Setting $w_{ij} = \delta(i = j \wedge i > l)$ amounts to not using a graph regularizer. If we assume hard labels (i.e., $H(r_i) = 0$) and that each p_i is parameterized by, say θ_i, then we can rewrite the above as

$$\mathcal{C}_{KL}(\mathrm{p}) \leq -\sum_{i=1}^{l} \log p_i(y_i; \theta_i) + \mu \sum_{i=l+1}^{n} H(p_i; \theta_i).$$

Now if all the θ_i were tied to a single θ then we have that

$$\mathcal{C}_{KL}(\mathrm{p}) \leq -\sum_{i=1}^{l} \log p_i(y_i; \theta) + \mu \sum_{i=l+1}^{n} H(p_i; \theta)$$

which is equal to the entropy minimization objective. Thus entropy minimization minimizes a non-convex upper bound on a special case of our proposed loss function. This is perhaps one of the reasons why graph-based approaches outperform entropy minimization on manifold-like data sets (see chapter 21 in Chapelle et al., 2007).

9.5 Rate of Convergence of MP

Recall that in Section 5 we showed that the rate of convergence of SQ-Loss-I is geometric (linear). Here we empirically compare the rate of convergence of MP and SQ-Loss-I. While we have so far been unable to derive theoretical bounds on the convergence rate of MP, our empirical analysis shows that MP convergences faster than SQ-Loss-I. The difficulties associated with analyzing the rate of convergence of MP are mostly due to the non-linear nature of the update equation for $p_i^{(n)}(y)$.

We ran both MP and SQ-Loss-I to convergence on a number of data sets taken from a variety of domains (see Table 3). For both algorithms we measured

$$f^{(n)} = \frac{C^{(n)} - C^*}{C^{(n-1)} - C^*} \tag{2}$$

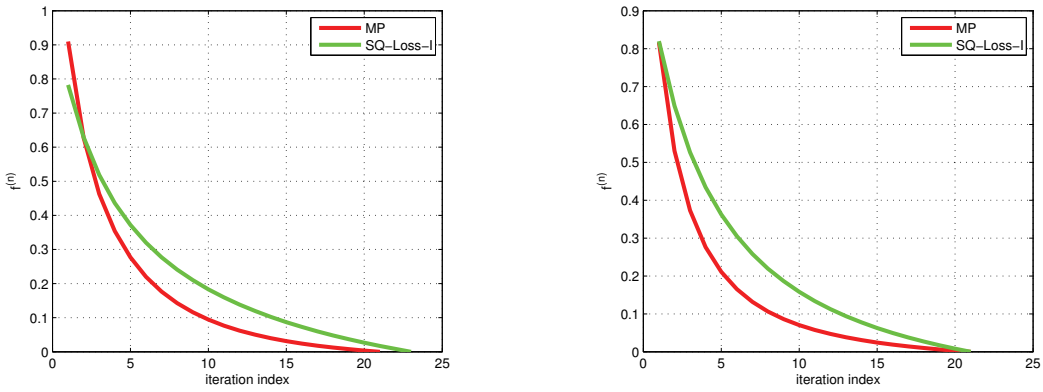


Figure 11: Plots showing the rate of convergence of MP and SQ-Loss-I in the case of the Text and USPS corpora. The x-axis represents iteration index and the y-axis rate of convergence, $f^{(n)}$ (see Equation 2).

and the plots of these quantities are shown in Figure 11 (similar trends were observed in the case of other data sets). In the above, C is the appropriate objective (i.e., $\mathcal{C}_{MP}$ in case of MP and $\mathcal{C}_{SQ}$ in the case of SQ-Loss-I) and C^* is the *corresponding* optimum value. While we have a standard test for convergence in the case of MP (see Theorem 9), for the purposes of comparison against SQ-Loss-I here, we use the following criteria: in either case we say that the algorithm has converged if the rate of change of the parameters falls below 0.5%. Figure 11 shows that MP converges faster in comparison to SQ-Loss-I. Based on these results, we make the following conjecture:

Conjecture 13 *MP has a geometric convergence rate, if not better.*

Finally a note on how to set α. Recall α is the hyperparameter that ensures that $p = q$ in the final solution in the case of MP. Recall that in theorem 8, we have shown that there exists a finite value of α such that $p^* = q^*$. In practice, we found that setting $\alpha = 2$ ensures the equality of p and q at convergence. As expected, we also found that increasing α leads to a slower rate of convergence in practice.

10. Conclusions

In this paper we presented a objective based on KLD for graph-based SSL. We have shown how the objective can be efficiently solved using alternating-minimization. In addition, we showed that the sequence of updates has a closed form solution and that it converges to the correct optima. We also derived a test for convergence of the iterative procedure that does not require the computation of the objective. A version of the squared-error graph-based SSL objective defined over measures was also presented. In this context we showed that squared-error has a geometric rate of convergence.

Our results show that MP is able to outperform other state-of-the-art graph-based SSL algorithms on a number of tasks from diverse set of domains ranging from speech to natural language to image processing. We have also shown how our algorithm can be scaled to very large data sets.

Acknowledgments

This work was supported by ONR MURI grant N000140510388, by NSF grant IIS-0093430, by the Companions project (IST programme under EC grant IST-FP6-034434), and by a Microsoft Research Fellowship.

Appendix A. Solving $\mathcal{P}_{KL}$ using Method of Multipliers

The first step in the application of MOM to solve $\mathcal{P}_{KL}$ is the construction of the augmented Lagrangian function for $\mathcal{C}_{KL}(\mathrm{p})$ which is given by

$$\mathcal{L}_{C_1}(\mathrm{p},\Lambda) = \mathcal{C}_{KL}(\mathrm{p}) + \sum_{i=1}^{n} \lambda_i \left(1 - \sum_y p_i(y)\right) + c \sum_{i=1}^{n} \left(1 - \sum_y p_i(y)\right)^2$$

where $\Lambda = \{\lambda_1, \ldots, \lambda_n\}$ are the Lagrange multipliers and $c \geq 0$ is the penalty parameter. Recall that we require $\sum_y p_i(y) = 1,\ \forall\, i$ and that $p_i(y) \geq 0,\ \forall\, i, y$. Notice that the objective $\mathcal{L}_{C_1}(\mathrm{p},\Lambda)$ only penalizes deviations from the equality constraints. In order to ensure that the inequality constraints in $\mathcal{P}_{KL}$ are met we make use of the *gradient projection method* (Bertsekas, 1999). Thus the update equation is given by

$$p_i^{(n)}(y) = \left[p_i^{(n-1)}(y) - \alpha^{(n-1)} \left(\frac{\partial \mathcal{L}_{C_1}(\mathrm{p},\Lambda)}{\partial p_i(y)} \right)_{\{\mathrm{p}=\mathrm{p}^{(n-1)}, \Lambda=\Lambda^{(n-1)}\}} \right]^+ .$$

Here $n = 1, \ldots,$ is the iteration index, $\alpha^{(n-1)}$ is the learning rate, and $[x]^+ = \max(x, 0)$. Determining an appropriate learning rate is often one of the biggest challenges associated with the application of gradient descent based optimization approaches. We use the Armijo rule (Bertsekas, 1999) to compute the learning rate, α. It can be shown that

$$\frac{\partial \mathcal{L}_{C_1}(\mathrm{p},\Lambda)}{\partial p_i(y)} = \mu \sum_{j=1}^{n} \left[w_{ej}\big(1 + \log p_i(y) - \log p_j(y)\big) - \frac{w_{je} p_j(y)}{p_i(y)} \right] - \frac{r_i(y)}{p_i(y)} \delta(e \leq l) + \nu(\log p_i(y) + 1) + \lambda_i + 2c\big(1 - \sum_y p_i(y)\big).$$

Under MOM, the update equation for the Lagrange multipliers is

$$\lambda_i^{(n)} = \lambda_i^{(n-1)} + c^{(n-1)} \left(\sum_y p_i^{(n-1)}(y) - 1 \right)$$

and the penalty parameter is updated using

$$c^{(n)} = \begin{cases} \beta c^{(n-1)} \text{ if } \sum_i \left(\tau_i^{(n)} - \gamma \tau_i^{(n-1)} \right) > 0 \\ c^{(n-1)} \text{ otherwise} \end{cases}$$

where $\tau_i^{(n)} = \big(1 - \sum_y p_i^{(n)}(y)\big)^2$. Intuitively, the above update rule for the penalty parameter increases its value only if the constraint violation is not decreased by a factor γ over the previous iteration. The iterative procedure terminates when

$$\frac{\mathcal{L}_{C_1}(\mathrm{p}^{(n-1)}, \Lambda^{(n-1)}) - \mathcal{L}_{C_1}(\mathrm{p}^{(n)}, \Lambda^{(n)})}{\mathcal{L}_{C_1}(\mathrm{p}^{(n-1)}, \Lambda^{(n-1)})} \leq \zeta.$$

Appendix B. Proof of Convergence

In this section we show that AM on $\mathcal{C}_{MP}$ converges to the correct optimum. We first show that the three-and-four points properties (to be defined shortly) hold for $\mathcal{C}_{MP}$ which then implies that the five-points property holds for $\mathcal{C}_{MP}$. We note that our proof is inspired by Csiszar and Tusnady (1984).

Definition 14 *If $\mathcal{P}$, $\mathcal{Q}$ are convex sets of finite measures, given a divergence $d(p,q)$, $p \in \mathcal{P}$, $q \in \mathcal{Q}$, then the "three points property" (3-pp) is said to hold for $p \in \mathcal{P}$ if $\forall\, q, q^{(0)} \in \mathcal{Q}$ we have*

$$\delta(p,p^{(1)}) + d(p^{(1)},q^{(0)}) \leq d(p,q^{(0)}) \text{ where } p^{(1)} \in \operatorname*{argmin}_{p\in\mathcal{P}} d(p,q^{(0)}) \text{ and}$$

$\delta(p,p^{(1)}) : \mathcal{P}\times\mathcal{P} \to \mathbb{R}^+$ is arbitrary and $\delta(p,p) = 0$.

Lemma 15 *$\mathcal{C}_{MP}(\mathrm{p},\mathrm{q})$ satisfies the 3-pp.*

Proof Let

$$\delta(\mathrm{p},\mathrm{p}^{(1)}) \triangleq \mu \sum_{i,j=1}^{n} w'_{ij} D_{KL}(p_i||p_i^{(1)}), \quad f(t) \triangleq \mathcal{C}_{MP}(\mathrm{p}^{(t)},\mathrm{q}^{(0)})$$

where $\mathrm{p}^{(t)} = (1-t)\mathrm{p} + t\mathrm{p}^{(1)}$, $0 < t \leq 1$ and thus $p_i^{(t)} = (1-t)p_i + t p_i^{(1)}$. As $f(t)$ attains its minimum at $t = 1$, $f(1) \leq f(t)$, $\forall\, 0 < t \leq 1$ and so

$$\frac{f(1) - f(t)}{1-t} \leq 0. \tag{3}$$

We have that

$$f(t) = \sum_{i=1}^{l} \sum_{y\in\mathrm{Y}} r_i \log \frac{r_i}{q_i^{(0)}} + \mu \sum_{i,j=1}^{n} w'_{ij} \sum_{y\in\mathrm{Y}} p_i^{(t)} \log \frac{p_i^{(t)}}{q_j^{(0)}} + \nu \sum_{i=1}^{n} \sum_{y\in\mathrm{Y}} p_i^{(t)} \log \frac{p_i^{(t)}}{u}$$

where we ignore the argument y in every measure for brevity (e.g., r_i is $r_i(y)$). Using the above in Equation 3 and taking the limit as $t \to 1$, we get

$$\begin{aligned}
&\lim_{t\to 1} \left(\mu \sum_{i,j=1}^{n} w'_{ij} \sum_{y\in\mathrm{Y}} \frac{1}{1-t} \left(p_i^{(1)} \log \frac{p_i^{(1)}}{q_j^{(0)}} - p_i^{(t)} \log \frac{p_i^{(t)}}{q_j^{(0)}} \right) + \nu \sum_{i=1}^{n} \sum_{y\in\mathrm{Y}} \frac{1}{1-t} \left(p_i^{(1)} \log \frac{p_i^{(1)}}{u} - p_i^{(t)} \log \frac{p_i^{(t)}}{u} \right) \right) \\
&\overset{(a)}{=} \mu \sum_{i,j=1}^{n} w'_{ij} \sum_{y\in\mathrm{Y}} \lim_{t\to 1} \left[\frac{1}{1-t} \left(p_i^{(1)} \log \frac{p_i^{(1)}}{q_j^{(0)}} - p_i^{(t)} \log \frac{p_i^{(t)}}{q_j^{(0)}} \right) \right] \\
&\qquad + \nu \sum_{i=1}^{n} \sum_{y\in\mathrm{Y}} \lim_{t\to 1} \left[\frac{1}{1-t} \left(p_i^{(1)} \log \frac{p_i^{(1)}}{u} - p_i^{(t)} \log \frac{p_i^{(t)}}{u} \right) \right] \\
&\overset{(b)}{=} \mu \sum_{i,j=1}^{n} w'_{ij} \sum_{y\in\mathrm{Y}} \left[\frac{\partial}{\partial t} \left(p_i^{(t)} \log \frac{p_i^{(t)}}{q_j^{(0)}} \right) \right]_{t=1} + \nu \sum_{i=1}^{n} \sum_{y\in\mathrm{Y}} \left[\frac{\partial}{\partial t} \left(p_i^{(t)} \log \frac{p_i^{(t)}}{u} \right) \right]_{t=1}
\end{aligned}$$

where (a) follows as both $p_i^{(t)} \log \frac{p_i^{(t)}}{q_j^{(0)}}$ and $p_i^{(t)} \log \frac{p_i^{(t)}}{u}$ are convex in t, and thus the terms within the summations are difference quotients of convex functions which are non-increasing. As a result we can use the monotone convergence theorem (MCT) (see page 87, Theorem 6 in H.L.Royden, 1988) to exchange the limit with the summations. Finally (b) follows from the definition of the derivative. Note that (a) can also be explained via the dominated convergence theorem (DCT) (see page 84, proposition 6 in H.L.Royden, 1988). If $q_j^{(0)}(y) > 0,\ \forall\ y, j$ then there exists $\gamma < \infty$ such that $p_i^{(1)} \log \frac{p_i^{(1)}}{q_j^{(0)}} - p_i^{(t)} \log \frac{p_i^{(t)}}{q_j^{(0)}} < \gamma$ because the difference of two finite real numbers is always bounded above which implies that the DCT can be used to distribute the limits within the summations. Thus we have that

$$\begin{aligned} 0 &\geq \mu \sum_{i,j=1}^{n} w'_{ij} \sum_{y \in \mathrm{Y}} \left[\frac{\partial}{\partial t} \left(p_i^{(t)} \log \frac{p_i^{(t)}}{q_j^{(0)}} \right) \right]_{t=1} + \nu \sum_{i=1}^{n} \sum_{y \in \mathrm{Y}} \left[\frac{\partial}{\partial t} \left(p_i^{(t)} \log \frac{p_i^{(t)}}{u} \right) \right]_{t=1} \\ &= \mu \sum_{i,j=1}^{n} w'_{ij} \sum_{y \in \mathrm{Y}} \left(p_i^{(1)} \log \frac{p_i^{(1)}}{q_j^{(0)}} - p_i \log \frac{p_i^{(1)}}{q_j^{(0)}} \right) + \nu \sum_{i=1}^{n} \sum_{y \in \mathrm{Y}} \left(p_i^{(1)} \log \frac{p_i^{(1)}}{u} - p_i \log \frac{p_i^{(1)}}{u} \right). \end{aligned}$$

The last equation follows as $\sum_{y \in \mathrm{Y}} (p_i^{(1)} - p_i) = 0$. As a result we have that

$$\begin{aligned} 0 &\geq \mu \sum_{i,j=1}^{n} w'_{ij} \sum_{y \in \mathrm{Y}} \left(p_i^{(1)} \log \frac{p_i^{(1)}}{q_j^{(0)}} - p_i \log \frac{p_i^{(1)}}{q_j^{(0)}} \right) + \nu \sum_{i=1}^{n} \sum_{y \in \mathrm{Y}} \left(p_i^{(1)} \log \frac{p_i^{(1)}}{u} - p_i \log \frac{p_i^{(1)}}{u} \right) \\ &= \mu \sum_{i,j=1}^{n} w'_{ij} D_{KL}(p_i^{(1)} || q_j^{(0)}) + \nu \sum_{i=1}^{n} D_{KL}(p_i^{(1)} || u) - \left(\mu \sum_{i,j=1}^{n} w'_{ij} \sum_{y \in \mathrm{Y}} p_i \log \frac{p_i^{(1)}}{q_j^{(0)}} + \nu \sum_{i=1}^{n} \sum_{y \in \mathrm{Y}} p_i \log \frac{p_i^{(1)}}{u} \right) \end{aligned}$$

From the definition of $\mathcal{C}_{MP}(\mathrm{p}, \mathrm{q})$ we have that

$$\mu \sum_{i,j=1}^{n} w'_{ij} D_{KL}(p_i^{(1)} || q_j^{(0)}) + \nu \sum_{i=1}^{n} D_{KL}(p_i^{(1)} || u) = \mathcal{C}_{MP}(\mathrm{p}^{(1)}, \mathrm{q}^{(0)}) - \sum_{i=1}^{l} D_{KL}\big(r_i || q_i^{(0)}\big).$$

Using the above we get

$$0 \geq \mathcal{C}_{MP}(\mathrm{p}^{(1)}, \mathrm{q}^{(0)}) - \sum_{i=1}^{l} D_{KL}\big(r_i || q_i^{(0)}\big) - \left(\mu \sum_{i,j=1}^{n} w'_{ij} \sum_{y \in \mathrm{Y}} p_i \log \frac{p_i^{(1)}}{q_j^{(0)}} + \nu \sum_{i=1}^{n} \sum_{y \in \mathrm{Y}} p_i \log \frac{p_i^{(1)}}{u} \right). \quad (4)$$

Consider

$$\sum_{y \in \mathrm{Y}} p_i \log \frac{p_i^{(1)}}{q_j^{(0)}} = \sum_{y \in \mathrm{Y}} p_i \log \frac{p_i^{(1)}}{q_j^{(0)}} \frac{p_i}{p_i} = \sum_{y \in \mathrm{Y}} p_i \left(\log \frac{p_i}{q_j^{(0)}} + \log \frac{p_i^{(1)}}{p_i} \right) = D_{KL}(p_i || q_j^{(0)}) - D_{KL}(p_i || p_i^{(1)}).$$

Similarly

$$\sum_{y \in \mathrm{Y}} p_i \log \frac{p_i^{(1)}}{u} = \sum_{y \in \mathrm{Y}} p_i \log \frac{p_i^{(1)}}{u} \frac{p_i}{p_i} = \sum_{y \in \mathrm{Y}} p_i \left(\log \frac{p_i}{u} + \log \frac{p_i^{(1)}}{p_i} \right) = D_{KL}(p_i || u) - D_{KL}(p_i || p_i^{(1)}).$$

Using the above two equations in Equation 4 we have that

$$\begin{aligned}
0 \geq\ & \mathcal{C}_{MP}(\mathrm{p}^{(1)},\mathrm{q}^{(0)}) - \sum_{i=1}^{l} D_{KL}\big(r_i||q_i^{(0)}\big) - \mu \sum_{i,j=1}^{n} w'_{ij}\Big(D_{KL}(p_i||q_j^{(0)}) - D_{KL}(p_i||p_i^{(1)})\Big) \\
& - \nu \sum_{i=1}^{n}\Big(D_{KL}(p_i||u) - D_{KL}(p_i||p_i^{(1)})\Big) \\
\overset{(a)}{\geq}\ & \mathcal{C}_{MP}(\mathrm{p}^{(1)},\mathrm{q}^{(0)}) - \mathcal{C}_{MP}(\mathrm{p},\mathrm{q}^{(0)}) + \mu \sum_{i,j=1}^{n} w'_{ij} D_{KL}(p_i||p_i^{(1)}) \\
=\ & \mathcal{C}_{MP}(\mathrm{p}^{(1)},\mathrm{q}^{(0)}) - \mathcal{C}_{MP}(\mathrm{p},\mathrm{q}^{(0)}) + \delta(\mathrm{p},\mathrm{p}^{(1)})
\end{aligned}$$

where (a) follows as $\nu \geq 0$ and $D_{KL}(p_i||p_i^{(1)}) \geq 0$. ■

Thus we have show that 3-pp holds for $\mathcal{C}_{MP}$.

Definition 16 *If $\mathcal{P}$, $\mathcal{Q}$ are convex sets of finite measures, given a divergence $d(p,q)$, $p \in \mathcal{P}$, $q \in \mathcal{Q}$, then the "four points property" (4-pp) is said to hold for $q \in \mathcal{Q}$ if $\forall\ p, p^{(1)} \in \mathcal{P}$ we have*

$$d(p,q^{(1)}) \leq \delta(p,p^{(1)}) + d(p,q)$$

where $q^{(1)} \in \operatorname*{argmin}_{q\in\mathcal{Q}} d(p^{(1)},q)$ and $\delta(p,p^{(1)})$ should match the definition of $\delta(.,.)$ used in 3-pp.

Lemma 17 *$\mathcal{C}_{MP}(\mathrm{p},\mathrm{q})$ satisfies the 4-pp.*

Proof Let

$$g(t) \triangleq \mathcal{C}_{MP}(\mathrm{p}^{(1)},\mathrm{q}^{(t)})$$

where $\mathrm{q}^{(t)} = (1-t)\mathrm{q} + t\mathrm{q}^{(1)}$, $0 < t \leq 1$ and thus $q_i^{(t)} = (1-t)q_i + tq_i^{(1)}$ and $q^{(1)}$ is as defined above. Also recall that $\delta(\mathrm{p},\mathrm{p}^{(1)}) \triangleq \mu \sum_{i,j=1}^{n} w'_{ij} D_{KL}(p_i||p_i^{(1)})$. The proof for this lemma proceeds in a manner similar to the proof of lemma 15. It should be clear that $g(t)$ achieves its minimum at $t = 1$ and as a result we have that

$$\frac{g(1) - g(t)}{1-t} \leq 0 \tag{5}$$

and

$$g(t) = \sum_{i=1}^{l}\sum_{y\in\mathrm{Y}} r_i \log\frac{r_i}{q_i^{(t)}} + \mu \sum_{i,j=1}^{n} w'_{ij} \sum_{y\in\mathrm{Y}} p_i^{(1)} \log\frac{p_i^{(1)}}{q_j^{(t)}} + \nu \sum_{i=1}^{n}\sum_{y\in\mathrm{Y}} p_i^{(1)} \log\frac{p_i^{(1)}}{u}.$$

Using the above in Equation 5 and passing it to the limit we get

$$\begin{aligned}
&\lim_{t\to 1}\Bigg(\sum_{i=1}^{l}\sum_{y\in Y}\frac{1}{1-t}\big(r_i\log\frac{r_i}{q_i^{(1)}}-r_i\log\frac{r_i}{q_i^{(t)}}\big)+\mu\sum_{i,j=1}^{n}w'_{ij}\sum_{y\in Y}\frac{1}{1-t}\big(p_i^{(1)}\log\frac{p_i^{(1)}}{q_j^{(1)}}-p_i^{(1)}\log\frac{p_i^{(1)}}{q_j^{(t)}}\big)\Bigg)\\
&\overset{(a)}{=}\sum_{i=1}^{l}\sum_{y\in Y}\lim_{t\to 1}\left[\frac{1}{1-t}\big(r_i\log\frac{r_i}{q_i^{(1)}}-r_i\log\frac{r_i}{q_i^{(t)}}\big)\right]+\mu\sum_{i,j=1}^{n}w'_{ij}\sum_{y\in Y}\lim_{t\to 1}\left[\frac{1}{1-t}\big(p_i^{(1)}\log\frac{p_i^{(1)}}{q_j^{(1)}}-p_i^{(1)}\log\frac{p_i^{(1)}}{q_j^{(t)}}\big)\right]\\
&\overset{(b)}{=}\sum_{i=1}^{l}\sum_{y\in Y}\left[\frac{\partial}{\partial t}\left(r_i\log\frac{r_i}{q_i^{(t)}}\right)\right]_{t=1}+\mu\sum_{i,j=1}^{n}w'_{ij}\sum_{y\in Y}\left[\frac{\partial}{\partial t}\left(p_i^{(1)}\log\frac{p_i^{(1)}}{q_j^{(t)}}\right)\right]_{t=1}\\
&=-\sum_{i=1}^{l}\sum_{y\in Y}r_i+\sum_{i=1}^{l}\sum_{y\in Y}\frac{r_i}{q_i^{(1)}}q_i-\mu\sum_{i,j=1}^{n}w'_{ij}\sum_{y\in Y}p_i^{(1)}+\mu\sum_{i,j=1}^{n}w'_{ij}\sum_{y\in Y}\frac{p_i^{(1)}}{q_j^{(1)}}q_j
\end{aligned}$$

where (a) once again follows from using DCT. This is because $\mathcal{C}_{MP}(\mathrm{p},\mathrm{q}^{(1)})$, $\mathcal{C}_{MP}(\mathrm{p},\mathrm{q})<\infty$ (else 4-pp trivially holds) and as $\mathcal{C}_{MP}(\mathrm{p},\mathrm{q})$ is the sum of all positive terms, it implies each term is finite and thus bounded above. Also (b) follows from the definition of the derivative. As a result we have that

$$\begin{aligned}
0&\geq-\sum_{i=1}^{l}\sum_{y\in Y}r_i+\sum_{i=1}^{l}\sum_{y\in Y}\frac{r_i}{q_i^{(1)}}q_i-\mu\sum_{i,j=1}^{n}w'_{ij}\sum_{y\in Y}p_i^{(1)}+\mu\sum_{i,j=1}^{n}w'_{ij}\sum_{y\in Y}\frac{p_i^{(1)}}{q_j^{(1)}}q_j\\
&=-l-\mu\sum_{i,j=1}^{n}w'_{ij}+\sum_{i=1}^{l}\sum_{y\in Y}\frac{r_i}{q_i^{(1)}}q_i+\mu\sum_{i,j=1}^{n}w'_{ij}\sum_{y\in Y}\frac{p_i^{(1)}}{q_j^{(1)}}q_j. \qquad (6)
\end{aligned}$$

Now consider

$$\begin{aligned}
&\mathcal{C}_{MP}(\mathrm{p},\mathrm{q})-\mathcal{C}_{MP}(\mathrm{p},\mathrm{q}^{(1)})\\
&\quad=\sum_{i=1}^{l}\sum_{y\in Y}\left(r_i\log\frac{r_i}{q_i}-r_i\log\frac{r_i}{q_i^{(1)}}\right)+\mu\sum_{i,j=1}^{n}w'_{ij}\sum_{y\in Y}\left(p_i\log\frac{p_i}{q_j}-p_i\log\frac{p_i}{q_j^{(1)}}\right)\\
&\quad=\sum_{i=1}^{l}\sum_{y\in Y}r_i\log\frac{q_i^{(1)}}{q_i}+\mu\sum_{i,j=1}^{n}w'_{ij}\sum_{y\in Y}p_i\log\frac{q_j^{(1)}p_i}{q_jp_i^{(1)}}-\mu\sum_{i,j=1}^{n}w'_{ij}D_{KL}(p_i||p_i^{(1)}).
\end{aligned}$$

Thus we have that

$$\begin{aligned}
&\mathcal{C}_{MP}(\mathrm{p},\mathrm{q})-\mathcal{C}_{MP}(\mathrm{p},\mathrm{q}^{(1)})+\delta(\mathrm{p},\mathrm{p}^{(1)})\\
&\quad=\sum_{i=1}^{l}\sum_{y\in Y}r_i\log\frac{q_i^{(1)}}{q_i}+\mu\sum_{i,j=1}^{n}w'_{ij}\sum_{y\in Y}p_i\log\frac{q_j^{(1)}p_i}{q_jp_i^{(1)}}.
\end{aligned}$$

Using the variational inequality $-\log(x) \geq (1-x)$ in the above we get

$$\begin{aligned}
&\mathcal{C}_{MP}(\mathrm{p},\mathrm{q}) - \mathcal{C}_{MP}(\mathrm{p},\mathrm{q}^{(1)}) + \delta(\mathrm{p},\mathrm{p}^{(1)}) \\
&\quad \geq \sum_{i=1}^{l} \sum_{y \in \mathrm{Y}} r_i \left(1 - \frac{q_i}{q_i^{(1)}}\right) + \mu \sum_{i,j=1}^{n} w'_{ij} \sum_{y \in \mathrm{Y}} p_i \left(1 - \frac{q_j p_i^{(1)}}{q_j^{(1)} p_i}\right) \\
&\quad = l + \mu \sum_{i,j=1}^{n} w'_{ij} - \sum_{i=1}^{l} \sum_{y \in \mathrm{Y}} \frac{r_i}{q_i^{(1)}} q_i - \mu \sum_{i,j=1}^{n} w'_{ij} \sum_{y \in \mathrm{Y}} \frac{p_i^{(1)}}{q_j^{(1)}} q_j \\
&\quad \overset{(a)}{\geq} 0
\end{aligned}$$

where (a) follows from Equation 6. ■

Which implies 4-pp holds for $\mathcal{C}_{MP}$.

Theorem 18 *$\mathcal{C}_{MP}(\mathrm{p},\mathrm{q})$ satisfies the 5-pp.*

Proof Follows as $\mathcal{C}_{MP}(\mathrm{p},\mathrm{q})$ satisfies both 3-pp and 4-pp. ■

Theorem 5 (Convergence of AM on $\mathcal{C}_{MP}$) *If*

$$\mathrm{p}^{(n)} = \operatorname*{argmin}_{\mathrm{p} \in \triangle^m} \mathcal{C}_{MP}(\mathrm{p},\mathrm{q}^{(n-1)}), \quad \mathrm{q}^{(n)} = \operatorname*{argmin}_{\mathrm{q} \in \triangle^m} \mathcal{C}_{MP}(\mathrm{p}^{(n)},\mathrm{q}) \text{ and } q_i^{(0)}(y) > 0 \; \forall \, y \in \mathrm{Y}, \; \forall i \text{ then}$$

(a) $\mathcal{C}_{MP}(\mathrm{p},\mathrm{q}) + \mathcal{C}_{MP}(\mathrm{p},\mathrm{p}^{(0)}) \geq \mathcal{C}_{MP}(\mathrm{p},\mathrm{q}^{(1)}) + \mathcal{C}_{MP}(\mathrm{p}^{(1)},\mathrm{q}^{(1)})$ *for all* $\mathrm{p},\mathrm{q} \in \triangle^m$, *and*

(b) $\lim_{n \to \infty} \mathcal{C}_{MP}(\mathrm{p}^{(n)},\mathrm{q}^{(n)}) = \inf_{\mathrm{p},\mathrm{q} \in \triangle^m} \mathcal{C}_{MP}(\mathrm{p},\mathrm{q})$.

Proof (a) follows as a result of Theorem 18. (b) is the direct result of (a) and theorem 3 in Csiszar and Tusnady (1984). ■

Appendix C. Equality of Solutions

Lemma 19 *If* $\mathrm{p} = \mathrm{q} = \tilde{\mathrm{p}}$ *then we have that* $\mathcal{C}_{MP}(\tilde{\mathrm{p}},\tilde{\mathrm{p}}) = \mathcal{C}_{KL}(\tilde{\mathrm{p}})$.

Proof Follows from the definitions of $\mathcal{C}_{KL}$ and $\mathcal{C}_{MP}$. ■

Lemma 6 *We have that*

$$\min_{\mathrm{p},\mathrm{q} \in \triangle^m} \mathcal{C}_{MP}(\mathrm{p},\mathrm{q}; w'_{ii} = 0) \leq \min_{\mathrm{p} \in \triangle^m} \mathcal{C}_{KL}(\mathrm{p}).$$

Proof Follows from the observation that

$$\min_{\mathrm{p}\in\triangle^m} \mathcal{C}_{KL}(\mathrm{p}) = \min_{\mathrm{p},\mathrm{q}\in\triangle^m,\mathrm{p}=\mathrm{q}} \mathcal{C}_{MP}(\mathrm{p},\mathrm{q};w'_{ii}=0) \geq \min_{\mathrm{p},\mathrm{q}\in\triangle^m} \mathcal{C}_{MP}(\mathrm{p},\mathrm{q};w'_{ii}=0\ \forall\ i)$$

The last step follows since the unconstrained minimum can never be larger than the constrained minimum. ■

Theorem 7 *Given any $A,B,S\in\triangle^m$ (i.e., $A=[a_1,\ldots,a_m]$, $B=[b_1,\ldots,b_m]$, $S=[s_1,\ldots,s_m]$) such that $a_i(y),b_i(y),s_i(y)>0,\ \forall\ i,y$ and $A\neq B$ (i.e., not all $a_i(y)=b_i(y)$) then there exists a finite α such that*

$$\mathcal{C}_{MP}(A,B) \geq \mathcal{C}_{MP}(S,S) = \mathcal{C}_{KL}(S).$$

Proof First

$$\begin{aligned}\mathcal{C}_{MP}(A,B) &= \sum_{i=1}^{l} D_{KL}\big(r_i||b_i\big) + \mu\sum_{i=1}^{n}\sum_{j\in\mathcal{N}'(i)} w'_{ij} D_{KL}\big(a_i||b_j\big) - \nu\sum_{i=1}^{n} H(a_i)\\ &= \sum_{i=1}^{l} D_{KL}\big(r_i||b_i\big) + \mu\sum_{i=1}^{n}\sum_{j\in\mathcal{N}(i)} w_{ij} D_{KL}\big(a_i||b_j\big) - \nu\sum_{i=1}^{n} H(a_i)\\ &\qquad + \mu\sum_{i=1}^{m}(w_{ii}+\alpha) D_{KL}\big(a_i||b_i\big)\end{aligned}$$

and so we want

$$\begin{aligned}&\sum_{i=1}^{l} D_{KL}\big(r_i||b_i\big) + \mu\sum_{i=1}^{n}\sum_{j\in\mathcal{N}(i)} w_{ij} D_{KL}\big(a_i||b_j\big) - \nu\sum_{i=1}^{n} H(a_i)\\ &\qquad + \mu\sum_{i=1}^{m}(w_{ii}+\alpha) D_{KL}\big(a_i||b_i\big) - \mathcal{C}_{MP}(S,S) \geq 0\end{aligned}$$

which holds if

$$\begin{aligned}\alpha &\geq \frac{\mathcal{C}_{MP}(S,S) - \sum_{i=1}^{l} D_{KL}\big(r_i||b_i\big) - \mu\sum_{i,j} w_{ij} D_{KL}\big(a_i||b_j\big) + \nu\sum_i H(a_i)}{\mu\sum_i D_{KL}\big(a_i||b_i\big)}\\ &= \frac{\mathcal{C}_{MP}(S,S) - \mathcal{C}_{MP}(A,B;\alpha=0)}{\mu\sum_i D_{KL}\big(a_i||b_i\big)} = \frac{\mathcal{C}_{KL}(S) - \mathcal{C}_{MP}(A,B;\alpha=0)}{\mu\sum_i D_{KL}\big(a_i||b_i\big)}.\end{aligned}$$

■

Theorem 8 (Equality of Solutions of $\mathcal{C}_{KL}$ and $\mathcal{C}_{MP}$) *Let*

$$\hat{\mathrm{p}} = \operatorname*{argmin}_{\mathrm{p}\in\triangle^m} \mathcal{C}_{KL}(\mathrm{p}) \text{ and } (\mathrm{p}^*_{\tilde{\alpha}},\mathrm{q}^*_{\tilde{\alpha}}) = \operatorname*{argmin}_{\mathrm{p},\mathrm{q}\in\triangle^m} \mathcal{C}_{MP}(\mathrm{p},\mathrm{q};\tilde{\alpha})$$

for an arbitrary $\alpha = \tilde{\alpha} > 0$ *where* $\mathrm{p}^*_{\tilde{\alpha}} = (p^*_{1;\tilde{\alpha}}, \cdots, p^*_{m;\tilde{\alpha}})$ *and* $\mathrm{q}^*_{\tilde{\alpha}} = (q^*_{1;\tilde{\alpha}}, \cdots, q^*_{m;\tilde{\alpha}})$. *Then there exists a finite* $\hat{\alpha}$ *such that at convergence of AM, we have that* $\hat{\mathrm{p}} = \mathrm{p}^*_{\hat{\alpha}} = \mathrm{q}^*_{\hat{\alpha}}$. *Further, it is the case that if* $\mathrm{p}^*_{\tilde{\alpha}} \neq \mathrm{q}^*_{\tilde{\alpha}}$, *then*

$$\hat{\alpha} \geq \frac{\mathcal{C}_{KL}(\hat{\mathrm{p}}) - \mathcal{C}_{MP}(\mathrm{p}^*_{\tilde{\alpha}}, \mathrm{q}^*_{\tilde{\alpha}}; \alpha = 0)}{\mu \sum_{i=1}^n D_{KL}(p^*_{i;\tilde{\alpha}} || q^*_{i;\tilde{\alpha}})}$$

and if $\mathrm{p}^*_{\tilde{\alpha}} = \mathrm{q}^*_{\tilde{\alpha}}$ *then* $\hat{\alpha} \geq \tilde{\alpha}$.

Proof First if $\mathrm{p}^*_{\tilde{\alpha}} = \mathrm{q}^*_{\tilde{\alpha}}$, this means the minimum of the unconstrained version at $\tilde{\alpha}$ resulted in equality, and since this also considers all solutions where $p = q$, and since both $\mathcal{C}_{KL}$ and $\mathcal{C}_{MP}$ are strictly convex, we must have $\mathcal{C}_{MP}(\mathrm{p}^*_{\tilde{\alpha}}, \mathrm{q}^*_{\tilde{\alpha}}; \tilde{\alpha}) = \mathcal{C}_{KL}(\hat{\mathrm{p}})$. Also, since for any $p \neq q$ we have $\mathcal{C}_{MP}(\mathrm{p}, \mathrm{q}; \hat{\alpha}) > \mathcal{C}_{MP}(\mathrm{p}, \mathrm{q}; \tilde{\alpha})$ whenever $\hat{\alpha} \geq \tilde{\alpha}$, then for all $\hat{\alpha} \geq \tilde{\alpha}$, $\mathcal{C}_{MP}(\mathrm{p}^*_{\hat{\alpha}}, \mathrm{q}^*_{\hat{\alpha}}; \hat{\alpha}) = \mathcal{C}_{KL}(\hat{\mathrm{p}})$. Next if $\mathrm{p}^*_{\tilde{\alpha}} \neq \mathrm{q}^*_{\tilde{\alpha}}$, then from Theorem 7 we have that if

$$\infty > \hat{\alpha} \geq \frac{\mathcal{C}_{KL}(\hat{\mathrm{p}}) - \mathcal{C}_{MP}(\mathrm{p}^*_{\tilde{\alpha}}, \mathrm{q}^*_{\tilde{\alpha}}; \alpha = 0)}{\mu \sum_{i=1}^n D_{KL}(p^*_{i;\tilde{\alpha}} || q^*_{i;\tilde{\alpha}})}$$

we are guaranteed that $\mathrm{p}^*_{\hat{\alpha}} = \mathrm{q}^*_{\hat{\alpha}}$, thereby making the first case applicable. ■

Appendix D. Test for Convergence

Theorem 9 (Test for Convergence) *If* $\{(\mathrm{p}^{(n)}, \mathrm{q}^{(n)})\}_{n=1}^{\infty}$ *is generated by AM of* $\mathcal{C}_{MP}(\mathrm{p}, \mathrm{q})$ *and* $\mathcal{C}_{MP}(\mathrm{p}^*, \mathrm{q}^*) \triangleq \inf_{\mathrm{p},\mathrm{q} \in \triangle^n} \mathcal{C}_{MP}(\mathrm{p}, \mathrm{q})$ *then*

$$\mathcal{C}_{MP}(\mathrm{p}^{(n)}, \mathrm{q}^{(n)}) - \mathcal{C}_{MP}(\mathrm{p}^*, \mathrm{q}^*) \leq \sum_{i=1}^n \big(\delta(i \leq l) + d_i\big)\beta_i,$$

$$\beta_i \triangleq \log \sup_y \frac{q_i^{(n)}(y)}{q_i^{(n-1)}(y)}, \quad d_j = \sum_i w_{ij}.$$

Proof As $\mathcal{C}_{MP}(\mathrm{p}, \mathrm{q})$ satisfies the 5-pp we have that

$$\mathcal{C}_{MP}(\mathrm{p}, \mathrm{q}) + \mathcal{C}_{MP}(\mathrm{p}, \mathrm{q}^{(n-1)}) \geq \mathcal{C}_{MP}(\mathrm{p}, \mathrm{q}^{(n)}) + \mathcal{C}_{MP}(\mathrm{p}^{(n)}, \mathrm{q}^{(n)}) \quad \forall \mathrm{p}, \mathrm{q} \in \mathcal{P}.$$

Rearranging the terms we have that

$$\mathcal{C}_{MP}(\mathrm{p}^{(n)}, \mathrm{q}^{(n)}) - \mathcal{C}_{MP}(\mathrm{p}, \mathrm{q}) \leq \mathcal{C}_{MP}(\mathrm{p}, \mathrm{q}^{(n-1)}) - \mathcal{C}_{MP}(\mathrm{p}, \mathrm{q}^{(n)}).$$

As the above holds for all $\mathrm{p}, \mathrm{q} \in \mathcal{P}$, it follows that

$$\mathcal{C}_{MP}(\mathrm{p}^{(n)}, \mathrm{q}^{(n)}) - \mathcal{C}_{MP}(\mathrm{p}^*, \mathrm{q}^*) \leq \mathcal{C}_{MP}(\mathrm{p}^*, \mathrm{q}^{(n-1)}) - \mathcal{C}_{MP}(\mathrm{p}^*, \mathrm{q}^{(n)}).$$

Now

$$\begin{aligned}
\mathcal{C}_{MP}(\mathrm{p}^*,\mathrm{q}^{(n-1)}) - \mathcal{C}_{MP}(\mathrm{p}^*,\mathrm{q}^{(n)}) &= \sum_{i=1}^{l}\sum_{y} r_i(y)\log\frac{q_i^{(n)}(y)}{q_i^{(n-1)}(y)} + \mu\sum_{i,j=1}^{m} w_{ij}\sum_{y} p_i^*(y)\log\frac{q_j^{(n)}(y)}{q_j^{(n-1)}(y)} \\
&= \sum_{i=1}^{l}\mathbf{E}_{r_i}\left[\log\frac{q_i^{(n)}(y)}{q_i^{(n-1)}(y)}\right] + \mu\sum_{i,j=1}^{m} w_{ij}\,\mathbf{E}_{p_i^*}\left[\log\frac{q_j^{(n)}(y)}{q_j^{(n-1)}(y)}\right] \\
&\overset{(a)}{\leq} \sum_{i=1}^{l}\sup_{y}\left[\log\frac{q_i^{(n)}(y)}{q_i^{(n-1)}(y)}\right] + \mu\sum_{i,j=1}^{m} w_{ij}\sup_{y}\left[\log\frac{q_j^{(n)}(y)}{q_j^{(n-1)}(y)}\right] \\
&= \sum_{i=1}^{l}\log\sup_{y}\left[\frac{q_i^{(n)}(y)}{q_i^{(n-1)}(y)}\right] + \mu\sum_{i,j=1}^{m} w_{ij}\,\log\sup_{y}\left[\frac{q_j^{(n)}(y)}{q_j^{(n-1)}(y)}\right] \\
&= \sum_{i=1}^{m}\big(\delta(i\leq l)+d_i\big)\log\sup_{y}\frac{q_i^{(n)}(y)}{q_i^{(n-1)}(y)}
\end{aligned}$$

where (a) follows as $E(f(x)) \leq \sup f(x)$ and recall $d_j = \sum_i w_{ij}$.

■

Appendix E. Update Equations for $\mathrm{p}^{(n)}$ and $\mathrm{q}^{(n)}$

The Lagrangian (ignoring the non-negativity constraints) for solving $\min_{\mathrm{p}\in\triangle^n} \mathcal{C}_{MP}(\mathrm{p},\mathrm{q}^{(n-1)})$ is given by

$$\mathcal{L}(\mathrm{p},\Lambda) = \sum_{i=1}^{l} D_{KL}\big(r_i||q_i\big) + \mu\sum_{i,j=1}^{n} w'_{ij}D_{KL}\big(p_i||q_j^{(n-1)}\big) - \nu\sum_{i=1}^{n}H(p_i) + \sum_{i}\lambda_i\left(\sum_{y}p_i(y)-1\right)$$

where $\Lambda = \{\lambda_1,\ldots,\lambda_n\}$. As KKT conditions apply (since we have a convex optimization problem), we have that $\nabla_{p_i(y)}\mathcal{L}(\mathrm{p},\Lambda) = 0$ and $\mathrm{p}\in\triangle^n$ at the optimal solution. Solving the above we have

$$\log p_i(y) = \frac{-\lambda_i - \beta_i^{(n-1)}(y)}{\alpha_i}.$$

Recall $\alpha_i = \nu + \mu\sum_j w'_{ij}$, $\beta_i^{(n-1)}(y) = -\nu + \mu\sum_j w'_{ij}(\log q_j^{(n-1)}(y) - 1)$. Using the above in Equation 7 leads to the dual problem in Λ which admits a closed form solution given by

$$\lambda_i = \alpha_i\log\left(\sum_{y}\exp^{\frac{\beta_i^{(n-1)}(y)}{\alpha_i}}\right) \implies \mathbf{p_i^{(n)}}(\mathbf{y}) = \frac{\mathbf{1}}{Z_{\mathbf{i}}}\exp^{\frac{\beta_{\mathbf{i}}^{(\mathbf{n-1})}(\mathbf{y})}{\alpha_{\mathbf{i}}}}.$$

Clearly $p_i^{(n)}(y) \geq 0,\ \forall\, i,y$.

The update for $\mathrm{q}^{(n)}$ may be obtained by constructing the Lagrangian for the optimization problem $\min_{\mathrm{q}\in\triangle^n} \mathcal{C}_{MP}(\mathrm{p}^{(n)},\mathrm{q})$ which is given by

$$\mathcal{L}(\mathrm{q},\Lambda) = \sum_{i=1}^{l} D_{KL}\big(r_i||q_i\big) + \mu \sum_{i,j=1}^{n} w'_{ij} D_{KL}\big(p_i^{(n)}||q_j\big) - \nu \sum_{i=1}^{n} H(p_i^{(n)}) + \sum_i \lambda_i \Bigg(\sum_y q_i(y) - 1 \Bigg) + \sum_{i,y} \sigma_{iy} q_i(y)$$

where $\Lambda = \{\lambda_1,\ldots,\lambda_n,\sigma_{11},\ldots,\sigma_{n|\mathrm{Y}|}\}$. In this case KKT conditions require that $\nabla_{q_i(y)}\mathcal{L}(\mathrm{q},\Lambda) = 0$, $\sum_y q_i(y) - 1 \;\forall\, y$, $\sigma_{iy} q_i(y) = 0 \;\forall\, i,y$ solving which yields

$$\mathbf{q_i^{(n)}}(\mathbf{y}) = \frac{\mathbf{r_i}(\mathbf{y})\delta(\mathbf{i}\le\mathbf{l}) + \mu \sum_{\mathbf{j}} \mathbf{w'_{ji}} \mathbf{p_j^{(n)}}(\mathbf{y})}{\delta(\mathbf{i}\le\mathbf{l}) + \mu \sum_{\mathbf{j}} \mathbf{w'_{ji}}}.$$

Appendix F. Convergence Rate of SQ-Loss

Lemma 21 (Linear Rate of Convergence, see page 64 in Bertsekas, 1999) *If $\{x_n\}$ is a convergent sequence such that $x_n \to 0$ and $x_n > 0 \;\forall\, n$, then x_n is said to converge linearly if*

$$\limsup_{n\to\infty} \frac{x_n}{x_{n-1}} \le \eta$$

where $\eta \in (0,1)$.

Theorem 11 (Geometric Rate of Convergence for SQ-Loss) *If*

(a) $\nu > 0$, and

(b) $\mathbf{W}$ has at least one non-zero off-diagonal element in every row (i.e., $\mathbf{W}$ is irreducible)

then the sequence of updates

$$p_i^{(n)}(y) = \frac{r_i(y)\delta(i\le l) + \nu u(y) + \mu \sum_j w_{ij} p_j^{(n-1)}(y)}{\delta(i\le l) + \nu + \mu \sum_j w_{ij}}$$

has a linear (geometric) rate of convergence for all i and y.

Proof

The updates can re-written in matrix form as

$$\mathrm{p}^{(n)} = [S + \nu \mathbf{I}_m + \mu \mathbf{D}]^{-1} \left(\mathrm{r}' + \frac{\nu}{|\mathrm{Y}|} \mathbf{1}_{m\times|\mathrm{Y}|} + \mu \mathbf{W} \mathrm{p}^{(n-1)} \right)$$

where

$$S \triangleq \begin{pmatrix} \mathbf{I}_l & 0 \\ 0 & 0 \end{pmatrix}, \mathrm{r}' \triangleq \begin{pmatrix} \mathrm{r} \\ \mathbf{0}_{(m-l)\times|\mathrm{Y}|} \end{pmatrix},$$

$[\mathbf{D}]_{ii} = \sum_j w_{ij}$, $\mathbf{1}_{m\times|\mathrm{Y}|}$ is a matrix of all 1's of size $m \times |\mathrm{Y}|$ and $\mathbf{0}_{(m-l)\times|\mathrm{Y}|}$ is similarly defined to be matrix of all 0's . It can be shown that $\mathrm{p}^{(n)} \rightarrow \mathrm{p}^*$ and so we have that

$$\mathrm{p}^* = [S + \nu\mathbf{I}_m + \mu\mathbf{D}]^{-1}\left(\mathrm{r}' + \frac{\nu}{|\mathrm{Y}|}\mathbf{1}_{m\times|\mathrm{Y}|} + \mu\mathbf{W}\mathrm{p}^*\right).$$

As a result

$$\mathrm{p}^{(n)} - \mathrm{p}^* = [S + \nu\mathbf{I}_m + \mu\mathbf{D}]^{-1}\left(\mu\mathbf{W}(\mathrm{p}^{(n-1)} - \mathrm{p}^*)\right)$$

which implies that

$$\| \mathrm{p}^{(n)} - \mathrm{p}^* \| = \| [S + \nu\mathbf{I}_m + \mu\mathbf{D}]^{-1}\left(\mu\mathbf{W}(\mathrm{p}^{(n-1)} - \mathrm{p}^*)\right) \|$$

where $\| A \|$ is the 2-norm (Euclidean norm) of the matrix A. Thus

$$\begin{aligned}\| \mathrm{p}^{(n)} - \mathrm{p}^* \| &= \| [S + \nu\mathbf{I}_m + \mu\mathbf{D}]^{-1}\left(\mu\mathbf{W}(\mathrm{p}^{(n-1)} - \mathrm{p}^*)\right) \| \\ &\leq \mu \| [S + \nu\mathbf{I}_m + \mu\mathbf{D}]^{-1}\mathbf{W} \| \| \mathrm{p}^{(n-1)} - \mathrm{p}^* \|\end{aligned}$$

and so

$$\frac{\| \mathrm{p}^{(n)} - \mathrm{p}^* \|}{\| \mathrm{p}^{(n-1)} - \mathrm{p}^* \|} \leq \mu \| [S + \nu\mathbf{I}_m + \mu\mathbf{D}]^{-1}\mathbf{W} \| .$$

Let $Z \triangleq \frac{1}{\mu}S + \frac{\nu}{\mu}\mathbf{I}_m + \mathbf{D}$ and so

$$\frac{\| \mathrm{p}^{(n)} - \mathrm{p}^* \|}{\| \mathrm{p}^{(n-1)} - \mathrm{p}^* \|} \leq \| Z^{-1}\mathbf{W} \| .$$

It should be clear that Z is a diagonal matrix.

The Perron-Frobenius theorem states that given any irreducible matrix A such that $a_{ij} \geq 0$ and a_{ij} are real then

$$\min_i \sum_j a_{ij} \leq \lambda_{max}(A) \leq \max_i \sum_j a_{ij}$$

where $\lambda_{max}(A)$ represents the maximum eigenvalue of A. If we apply the above theorem to the matrix $\mathbf{D}^{-1}\mathbf{W}$, then we have that $\lambda_{max}(\mathbf{D}^{-1}\mathbf{W}) = 1$. If we apply the same to $Z^{-1}\mathbf{W}$, then we have that

$$\min_i \sum_j \frac{w_{ij}}{\frac{1}{\mu}\delta(i \leq l) + \frac{\nu}{\mu} + \sum_k w_{ik}} \leq \lambda_{max}(Z^{-1}\mathbf{W}) \leq \max_i \sum_j \frac{w_{ij}}{\frac{1}{\mu}\delta(i \leq l) + \frac{\nu}{\mu} + \sum_k w_{ik}}.$$

But we have that

$$\sum_j \frac{w_{ij}}{\sum_k w_{ik}} = 1 \tag{7}$$

and so if $\nu > 0$ then we have that

$$\sum_j \frac{w_{ij}}{\frac{1}{\mu}\delta(i \leq l) + \frac{\nu}{\mu} + \sum_k w_{ik}} < 1.$$

As a result

$$\min_i \sum_j \frac{w_{ij}}{\frac{1}{\mu}\delta(i \leq l) + \frac{\nu}{\mu} + \sum_k w_{ik}} \leq \lambda_{max}(Z^{-1}\mathbf{W}) < 1.$$

In addition we also have that $\sum_j w_{ij} > 0$ for all i and so

$$0 < \lambda_{max}(Z^{-1}\mathbf{W}) < 1.$$

As a result

$$\| Z^{-1}\mathbf{W} \| = \sqrt{\lambda_{max}\left((Z^{-1}\mathbf{W})^T Z^{-1}\mathbf{W}\right)} = \sqrt{\lambda_{max}\left(Z^{-1}\mathbf{W}\right)^2} = \lambda_{max}(Z^{-1}\mathbf{W}).$$

The above implies that

$$\limsup_{n\to\infty} \frac{\| \mathrm{p}^{(n)} - \mathrm{p}^* \|}{\| \mathrm{p}^{(n-1)} - \mathrm{p}^* \|} \leq \| Z^{-1}\mathbf{W} \| = \lambda_{max}(Z^{-1}\mathbf{W}).$$

As $0 < \lambda_{max}(Z^{-1}\mathbf{W}) < 1$, we have that $\mathrm{p}^{(n)}$ has a linear rate of convergence.

■

References

A. Alexandrescu and K. Kirchhoff. Data-driven graph construction for semi-supervised graph-based learning in nlp. In *Proc. of the Human Language Technologies Conference (HLT-NAACL)*, 2007a.

A. Alexandrescu and K. Kirchhoff. Graph-based learning for statistical machine translation. In *Proc. of the Human Language Technologies Conference (HLT-NAACL)*, 2007b.

S. Arya and D. M. Mount. Approximate nearest neighbor queries in fixed dimensions. In *ACM-SIAM Symp. on Discrete Algorithms (SODA)*, 1993.

S. Arya, D. M. Mount, N. S. Netanyahu, R. Silverman, and A. Wu. An optimal algorithm for approximate nearest neighbor searching. *Journal of the ACM*, 1998.

A. Banerjee, S. Merugu, I. S. Dhillon, and J. Ghosh. Clustering with bregman divergences. *Journal of Machine Learning Research*, 2005.

R. Bekkerman, R. El-Yaniv, N. Tishby, and Y. Winter. Distributional word clusters vs. words for text categorization. *Journal of Machine Learning Research*, 3:1183–1208, 2003. ISSN 1533-7928.

M. Belkin, P. Niyogi, and V. Sindhwani. On manifold regularization. In *Proc. of the Conference on Artificial Intelligence and Statistics (AISTATS)*, 2005.

Y. Bengio, O. Delalleau, and N. L. Roux. Label propogation and quadratic criterion. In O. Chapelle, B. Scholkopf, and A. Zien, editors, *Semi-Supervised Learning*. MIT Press, 2007.

D. Bertsekas. *Nonlinear Programming*. Athena Scientific Publishing, 1999.

J. Bilmes and A. Subramanya. Parallel graph-based semi-supervised learning. In R. Bekkerman, M. Bilenko, and J. Langford, editors, *Scaling Up Machine Learning*. Cambridge University Press, 2011. Forthcoming.

C. Bishop, editor. *Neural Networks for Pattern Recognition*. Oxford University Press, 1995.

J. Blitzer and J. Zhu. ACL 2008 tutorial on Semi-Supervised learning. `http://ssl-acl08.wikidot.com/`, 2008.

A. Blum and S. Chawla. Learning from labeled and unlabeled data using graph mincuts. In *Proc. 18th International Conf. on Machine Learning*, pages 19–26. Morgan Kaufmann, San Francisco, CA, 2001.

A. Blum and T. Mitchell. Combining labeled and unlabeled data with co-training. In *COLT: Proceedings of the Workshop on Computational Learning Theory*, 1998.

S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, 2006.

O. Chapelle, B. Scholkopf, and A. Zien. *Semi-Supervised Learning*. MIT Press, 2007.

W. Cheney and A. Goldstien. Proximity maps for convex sets. *American Mathematical Society*, 1959.

R. Collobert, F. Sinz, J. Weston, L. Bottou, and T. Joachims. Large scale transductive svms. *Journal of Machine Learning Research*, 2006.

A. Corduneanu and T. Jaakkola. On information regularization. In *Uncertainty in Artificial Intelligence*, 2003.

T. M. Cover and J. A. Thomas. *Elements of Information Theory*. Wiley Series in Telecommunications. Wiley, New York, 1991.

I. Csiszar and G Tusnady. Information geometry and alternating minimization procedures. *Statistics and Decisions*, 1984.

O. Delalleau, Y. Bengio, and N. L. Roux. Efficient non-parametric function induction in semi-supervised learning. In *Proc. of the Conference on Artificial Intelligence and Statistics (AISTATS)*, 2005.

A.P. Dempster, N.M. Laird, and D.B. Rubin. Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society, Series B*, 39(1):1–38, 1977.

N. Deshmukh, A. Ganapathiraju, A. Gleeson, J. Hamaker, and J. Picone. Resegmentation of switchboard. In *Proceedings of the International Conference on Spoken Language Processing*, pages 1543–1546, Sydney, Australia, November 1998.

S. Dumais, J. Platt, D. Heckerman, and M. Sahami. Inductive learning algorithms and representations for text categorization. In *CIKM '98: Proceedings of the Seventh International Conference on Information and Knowledge Management*, New York, NY, USA, 1998.

G. Evermann, H. Y. Chan, M. J. F. Gales, B. Jia, D. Mrva, P. C. Woodland, and K. Yu. Training lvcsr systems on thousands of hours of data. In *Proc. of ICASSP*, 2005.

W. Fei and Z. Changshui. Label propagation through linear neighborhoods. In *Proceedings of the 23rd International Conference on Machine Learning (ICML)*, pages 985–992, New York, NY, USA, 2006. ACM.

J.H. Friedman, J.L. Bentley, and R.A. Finkel. An algorithm for finding best matches in logarithmic expected time. *ACM Transaction on Mathematical Software*, 3, 1977.

J. Godfrey, E. Holliman, and J. McDaniel. Switchboard: telephone speech corpus for research and development. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing*, volume 1, pages 517–520, San Francisco, California, March 1992.

A. B. Goldberg and X. Zhu. Keepin' it real: Semi-supervised learning with realistic tuning. In *SemiSupLearn '09: Proceedings of the NAACL HLT 2009 Workshop on Semi-Supervised Learning for Natural Language Processing*, Morristown, NJ, USA, 2009. Association for Computational Linguistics.

Y. Grandvalet and Y. Bengio. Semi-supervised learning by entropy minimization. In *CAP*, 2005.

S Greenberg. The switchboard transcription project. Technical report, The Johns Hopkins University (CLSP) Summer Research Workshop, 1995.

A. Gunawardena. *The Information Geometri of EM Variants for Speech and Image Processing*. PhD thesis, Johns Hopkins University, 2001.

A. K. Halberstadt and J. R. Glass. Heterogeneous acoustic measurements for phonetic classification. In *Proc. Eurospeech '97*, pages 401–404, Rhodes, Greece, 1997. URL `citeseer.ist.psu.edu/article/halberstadt97heterogeneous.html`.

H.L.Royden. *Real Analysis*. Prentice Hall, 1988.

G. Ji and J. Bilmes. Dialog act tagging using graphical models. In *Proc. of ICASSP*, 2005.

T. Joachims. Transductive inference for text classification using support vector machines. In *Proc. of the International Conference on Machine Learning (ICML)*, 1999.

T. Joachims. SVM Light. `http://svmlight.joachims.org`, 2002.

T. Joachims. Transductive learning via spectral graph partitioning. In *Proc. of the International Conference on Machine Learning (ICML)*, 2003.

T. Joachims. SGT light. `http://sgt.joachims.org`, 2004.

D. Jurafsky and C. V. Ess-Dykema. Switchboard discourse language modeling project. Johns Hopkins Summer Workshop, 1997.

M. Karlen, J. Weston, A. Erkan, and R. Collobert. Large scale manifold transduction. In *International Conference on Machine Learning, ICML*, 2008.

J. Lafferty, S. D. Pietra, and V. D. Pietra. Statistical learning algorithms based on bregman distances. In *Proceedings of 1997 Canadian Workshop on Information Theory*, 1997.

K. F. Lee and H. Hon. Speaker independant phone recognition using hidden Markov models. *IEEE Transactions on Acoustics, Speech and Signal Processing*, 37(11), 1989.

D. Lewis et al. Reuters-21578. `http://www.daviddlewis.com/resources/testcollections/reuters21578`, 1987.

X. Li and J. Bilmes. Regularized adaptation of discriminative classifiers. In *Proc. IEEE Intl. Conf. on Acoustic, Speech and Signal Processing*, September 2006.

J. Malkin, A. Subramanya, and J. A. Bilmes. On the semi-supervised learning of multi-layered perceptrons. In *Proc. Annual Conference of the International Speech Communication Association (INTERSPEECH)*, Brighton, UK, September 2009.

G. S. Mann and A. McCallum. Simple, robust, scalable, semi-supervised learning via expectation regularization. In *Proc. of the International Conference on Machine Learning (ICML)*, 2007.

C. Martínez-Hinarejos, J. Benedí, and R. Granell. Statistical framework for a spanish spoken dialogue corpus. *Speech Communication*, 50(11-12), 2008.

N. Morgan. Personal communication, 2009.

B. Nadler, N. Srebro, and X. Zhou. Statistical analysis of semi-supervised learning: The limit of infinite unlabelled data. In *Advances in Neural Information Processing Systems (NIPS)*, 2010.

A. Ng and M. Jordan. On discriminative vs. generative classifiers: A comparison of logistic regression and naive bayes. In *Advances in Neural Information Processing Systems (NIPS)*, 2002.

K. Nigam, A. McCallum, S. Thrun, and T. Mitchell. Learning to classify text from labeled and unlabeled documents. In *AAAI '98/IAAI '98: Proceedings of the fifteenth national/tenth conference on Artificial intelligence/Innovative applications of artificial intelligence*, pages 792–799, 1998.

J. Pearl. *Jeffrey's Rule, Passage of Experience and Neo-Bayesianism in Knowledge Representation and Defeasible Reasoning*. Kluwer Academic Publishers, 1990.

M.F. Porter. An algorithm for suffix stripping. *Program*, 14(3):130–137, 1980.

V. Raghavan, P. Bollmann, and G. S. Jung. A critical investigation of recall and precision as measures of retrieval system performance. *ACM Trans. Inf. Syst.*, 7(3):205–229, 1989. ISSN 1046-8188.

R. Rifkin and A. Klautau. In defense of one-vs-all classification. *Journal of Machine Learning Research*, 2004.

G. Salton and C. Buckley. Term weighting approaches in automatic text retrieval. Technical report, Cornell University, Ithaca, NY, USA, 1987.

H. J. Scudder. Probability of error of some adaptive pattern-recognition machines. *IEEE Transactions on Information Theory*, 11, 1965.

M. Seeger. Learning with labeled and unlabeled data. Technical report, University of Edinburgh, U.K., 2000.

V. Sindhwani and S. Keerthi. Large scale semi-supervised linear svms. In *SIGIR '06: Proceedings of the 29th Annual International ACM SIGIR*, 2006.

V. Sindhwani, P. Niyogi, and M. Belkin. Beyond the point cloud: From transductive to semi-supervised learning. In *Proc. of the International Conference on Machine Learning (ICML)*, 2005.

P. Somervuo. Experiments with linear and nonlinear feature transformations in HMM based phone recognition. In *Proc. of ICASSP 2003*, 2003. URL `citeseer.ist.psu.edu/somervuo03experiments.html`.

A. Subramanya and J. Bilmes. Soft-supervised text classification. In *EMNLP*, 2008.

A. Subramanya and J. Bilmes. The semi-supervised switchboard transcription project. In *Interspeech*, 2009.

A. Subramanya, C. Bartels, J. Bilmes, and P. Nguyen. Uncertainty in training large vocabulary speech recognizers. In *Proc. of the IEEE Workshop on Speech Recognition and Understanding*, 2007.

M. Szummer and T. Jaakkola. Partially labeled classification with Markov random walks. In *Advances in Neural Information Processing Systems*, volume 14, 2001.

A. Tomkins. Keynote speech. CIKM Workshop on Search and Social Media, 2008.

I. W. Tsang and J. T. Kwok. Large-scale sparsified manifold regularization. In *Advances in Neural Information Processing Systems (NIPS) 19*, 2006.

K. Tsuda. Propagating distributions on a hypergraph by dual information regularization. In *Proceedings of the 22nd International Conference on Machine Learning*, 2005.

K. Tsuda, G. Rätsch, and M. K. Warmuth. Matrix exponentiated gradient updates for on-line learning and bregman projection. *J. Mach. Learn. Res.*, 6:995–1018, 2005. ISSN 1533-7928.

V. Vladimir. *Statistical Learning Theory*. Wiley Series, 1998.

J. Wang, T. Jebara, and S. Chang. Graph transduction via alternating minimization. In *Proc. of the International Conference on Machine Learning (ICML)*, 2008.

C. F. Jeff Wu. On the convergence properties of the EM algorithm. *The Annals of Statistics*, 11(1): 95–103, 1983.

W. Zangwill. *Nonlinear Programming: a Unified Approach*. Prentice-Hall International Series in Management, Englewood Cliffs: N.J., 1969.

X.H. Zhang and W.S. Lee. Hyperparameter learning for semi-supervised learning algorithms. In *NIPS*, 2006.

D. Zhou, J. Huang, and B. Schölkopf. Learning from labeled and unlabeled data on a directed graph. In *Proceedings of the 22nd International Conference on Machine Learning*. ACM, 2005.

X. Zhu. *Semi-Supervised Learning with Graphs*. PhD thesis, Carnegie Mellon University, 2005a.

X. Zhu. Semi-supervised learning literature survey. Technical Report 1530, Computer Sciences, University of Wisconsin-Madison, 2005b.

X. Zhu and Z. Ghahramani. Learning from labeled and unlabeled data with label propagation. Technical report, Carnegie Mellon University, 2002a.

X. Zhu and Z. Ghahramani. Towards semi-supervised classification with Markov random fields. Technical Report CMU-CALD-02-106, Carnegie Mellon University, 2002b.

X. Zhu, Z. Ghahramani, and J. Lafferty. Semi-supervised learning using gaussian fields and harmonic functions. In *Proc. of the International Conference on Machine Learning (ICML)*, 2003.

X. Zhu, J. Kandola, Z. Ghahramani, and J. Lafferty. Nonparametric transforms of graph kernels for semi-supervised learning. In L. Saul, Y. Weiss, and L. Bottou, editors, *Advances in Neural Information Processing Systems (NIPS)*, 2005.

V.W. Zue, S. Seneff, and J. Glass. Speech database development at MIT:TIMIT and beyond. In *Speech Comminication*, 1990.

Journal of Machine Learning Research 12 (2011) 3371-3412 Submitted 6/10; Revised 12/10; Published 11/11

Learning with Structured Sparsity

Junzhou Huang JZHUANG@UTA.EDU
Department of Computer Science and Engineering
University of Texas at Arlington
Arlington, TX, 76019 USA *

Tong Zhang TZHANG@STAT.RUTGERS.EDU
Department of Statistics
Rutgers University
Piscataway, NJ, 08854 USA †

Dimitris Metaxas DNM@CS.RUTGERS.EDU
Department of Computer Science
Rutgers University
Piscataway, NJ, 08854 USA

Editor: Francis Bach

Abstract

This paper investigates a learning formulation called *structured sparsity*, which is a natural extension of the standard sparsity concept in statistical learning and compressive sensing. By allowing arbitrary structures on the feature set, this concept generalizes the group sparsity idea that has become popular in recent years. A general theory is developed for learning with structured sparsity, based on the notion of coding complexity associated with the structure. It is shown that if the coding complexity of the target signal is small, then one can achieve improved performance by using coding complexity regularization methods, which generalize the standard sparse regularization. Moreover, a structured greedy algorithm is proposed to efficiently solve the structured sparsity problem. It is shown that the greedy algorithm approximately solves the coding complexity optimization problem under appropriate conditions. Experiments are included to demonstrate the advantage of structured sparsity over standard sparsity on some real applications.

Keywords: structured sparsity, standard sparsity, group sparsity, tree sparsity, graph sparsity, sparse learning, feature selection, compressive sensing

1. Introduction

We are interested in the sparse learning problem under the fixed design condition. Consider a fixed set of p basis vectors $\{\mathbf{x}_1, \ldots, \mathbf{x}_p\}$ where $\mathbf{x}_j \in \mathbb{R}^n$ for each j. Here, n is the sample size. Denote by X the $n \times p$ data matrix, with column j of X being $\mathbf{x}_j$. Given a random observation $\mathbf{y} = [\mathbf{y}_1, \ldots, \mathbf{y}_n] \in \mathbb{R}^n$ that depends on an underlying coefficient vector $\beta \in \mathbb{R}^p$, we are interested in the problem of estimating β under the assumption that the target coefficient β is sparse. Throughout the paper, we consider fixed design only. That is, we assume X is fixed, and randomization is with respect to the noise in the observation $\mathbf{y}$.

∗. An extended abstract of the paper was presented at the international conference of machine learning, 2009.

†. This author is partially supported by NSF DMS-1007527, NSF IIS-1016061, and AFOSR-10097389.

We consider the situation that the true mean of the observation $\mathbb{E}\mathbf{y}$ can be approximated by a sparse linear combination of the basis vectors. That is, there exists a target vector $\beta \in \mathbb{R}^p$ such that either $\mathbb{E}\mathbf{y} = X\beta$ or $\mathbb{E}\mathbf{y} - X\beta$ is small. Moreover, we assume that β is sparse. Define the support of a vector $\beta \in \mathbb{R}^p$ as

$$\mathrm{supp}(\beta) = \{j : \beta_j \neq 0\},$$

and $\|\beta\|_0 = |\mathrm{supp}(\beta)|$. A natural method for sparse learning is L_0 regularization:

$$\hat{\beta}_{L0} = \arg\min_{\beta \in \mathbb{R}^p} \hat{Q}(\beta) \quad \text{subject to } \|\beta\|_0 \leq s, \tag{1}$$

where s is the desired sparsity. For simplicity, unless otherwise stated, the objective function considered throughout this paper is the least squares loss

$$\hat{Q}(\beta) = \|X\beta - \mathbf{y}\|_2^2,$$

where $\|\cdot\|_2$ denotes the Euclidean norm.

Since this optimization problem is generally NP-hard, in practice, one often considers approximate solutions. A standard approach is convex relaxation of L_0 regularization to L_1 regularization, often referred to as Lasso (Tibshirani, 1996). Another commonly used approach is greedy algorithms, such as the orthogonal matching pursuit (OMP) (Tropp and Gilbert, 2007).

In practical applications, one often knows a structure on the coefficient vector β in addition to sparsity. For example, in group sparsity, one assumes that variables in the same group tend to be zero or nonzero simultaneously. The purpose of this paper is to study the more general estimation problem under structured sparsity. If meaningful structures exist, we show that one can take advantage of such structures to improve the standard sparse learning. Specifically, we study the following natural extension of L_0 regularization to structured sparsity problems. It replaces the L_0 constraint in (1) by a more general term $c(\beta)$, which we call *coding complexity*. The precise definition will be given later in Section 2, and some concrete examples will be given later in Section 4.

$$\hat{\beta}_{constr} = \arg\min_{\beta \in \mathbb{R}^p} \hat{Q}(\beta) \quad \text{subject to } c(\beta) \leq s. \tag{2}$$

In this formulation, s is a tuning parameter. Alternatively, we may also consider the penalized formulation

$$\hat{\beta}_{pen} = \arg\min_{\beta \in \mathbb{R}^p} \left[\hat{Q}(\beta) + \lambda c(\beta)\right], \tag{3}$$

where $\lambda > 0$ is a regularization parameter that can be tuned. Since (2) and (3) penalize the coding complexity $c(\beta)$, we shall call this approach *coding complexity regularization*.

The optimization of either (2) or (3) is generally hard. For related problems, there are two common approaches to alleviate this difficulty. One is convex relaxation (L_1 regularization to replace L_0 regularization for standard sparsity); the other is forward greedy selection (also called orthogonal matching pursuit or OMP). We do not know any extensions of L_1 regularization like convex relaxation methods that can handle general structured sparsity formulations with provable performance guarantees. In particular, the theoretical analysis in our companion paper (Huang and Zhang, 2010) for group Lasso fails to yield meaningful bounds for more complex convex relaxation methods that are proposed for general structured sparsity formulations considered in this paper. For this reason, we present an extension of the standard greedy OMP algorithm that can be applied to general structured sparsity problems, and more importantly, meaningful sparse recovery bounds can be obtained for this algorithm. We call the resulting procedure *structured greedy algorithm* or StructOMP, which approximately solves (2). The details will be described later in Section 3.

1.1 Related Work

The idea of using structure in addition to sparsity has been explored before. An example is group structure, which has received much attention recently. For example, group sparsity has been considered for simultaneous sparse approximation (Wipf and Rao, 2007) and multi-task compressive sensing and learning (Argyriou et al., 2008; Ji et al., 2008) from the Bayesian hierarchical modeling point of view. Under the Bayesian hierarchical model framework, data from all sources contribute to the estimation of hyper-parameters in the sparse prior model. The shared prior can then be inferred from multiple sources. He et al. recently extend the idea to the tree sparsity in the Bayesian framework (He and Carin, 2009a,b). Although the idea can be justified using standard Bayesian intuition, there are no theoretical results showing how much better (and under what kind of conditions) the resulting algorithms perform. In the statistical literature, Lasso has been extended to the group Lasso when there exist group/block structured dependencies among the sparse coefficients (Yuan and Lin, 2006).

However, none of the above mentioned work was able to show advantage of using group structure. Although some theoretical results were developed in Bach (2008) and Nardi and Rinaldo (2008), neither showed that group Lasso is superior to the standard Lasso. Koltchinskii and Yuan (2008) showed that group Lasso can be superior to standard Lasso when each group is an infinite dimensional kernel, by relying on the fact that meaningful analysis can be obtained for kernel methods in infinite dimension. Obozinski et al. (2008) considered a special case of group Lasso in the multi-task learning scenario, and showed that the number of samples required for recovering the exact support set is smaller for group Lasso under appropriate conditions. Huang and Zhang (2010) developed a theory for group Lasso using a concept called strong group sparsity, which is a special case of the general structured sparsity idea considered here. It was shown in Huang and Zhang (2010) that group Lasso is superior to standard Lasso for strongly group-sparse signals, which provides a convincing theoretical justification for using group structured sparsity. Related results can also be found in Chesneau and Hebiri (2008) and Lounici et al. (2009).

While group Lasso works under the strong group sparsity assumption, it doesn't handle the more general structures considered in this paper. Several limitations of group Lasso were mentioned by Huang and Zhang (2010). For example, group Lasso does not correctly handle overlapping groups (in that overlapping components are over-counted); that is, a given coefficient should not belong to different groups. This requirement is too rigid for many practical applications. To address this issue, a method called composite absolute penalty (CAP) is proposed in Zhao et al. (2009) which can handle overlapping groups. A satisfactory theory remains to be developed to rigorously demonstrate the effectiveness of the approach. In a related development, Kowalski and Torresani (2009) generalized the mixed norm penalty to structured shrinkage, which can identify structured significance maps and thus can handle the case of the overlapping groups. However, there were no additional theory to justify their methods.

It is also worth pointing out that independent of this paper, two recent work (Jacob et al., 2009; Jenatton et al., 2009) considered structured sparsity in the convex relaxation setting, and extended group Lasso to more complicated sparse regularization conditions. These work complement the idea considered in this paper, which focuses on a natural non-convex formulation of general structured sparsity, as well as its greedy approximation. Again, since convex relaxation methods are more difficult to analyze in the structured sparsity setting with overlapping groups, a satisfactory theoretical justification remains an open challenge. For example the analysis in our companion work (Huang

and Zhang, 2010) on group Lasso does not correctly generalize to the above mentioned convex relaxation formulations because a straight-forward application leads to a bound proportional to the number of overlapping groups covering a true variable. Unfortunately, at least for some of the structures considered in this paper (such as hierarchical tree structure), in order to show the effectiveness of using the extra structural information, we need $\Omega(\log_2(p))$ groups to cover each variable, which leads to a bound showing no benefits over standard Lasso if we directly apply the analysis of Huang and Zhang (2010). It is worth noting that the lack of analysis doesn't mean that formulations in Jacob et al. (2009) and Jenatton et al. (2009) are ineffective. For example, some algorithmic techniques are employed by Jenatton et al. (2009) to address the over-counting issue we mentioned above, but the resulting procedures are non-trivial to analyze. In comparison the greedy algorithm is easier to analyze and (being non-convex) doesn't suffer from the above mentioned problem. Therefore this paper focuses on developing a direct generalization of the popular OMP algorithm to handle structured sparsity.

In addition to the above mentioned work, other structures have also been explored in the literature. For example, so-called tonal and transient structures were considered for sparse decomposition of audio signals in Daudet (2004). Grimm et al. (2007) investigated positive polynomials with structured sparsity from an optimization perspective. The theoretical result there did not address the effectiveness of such methods in comparison to standard sparsity. The closest work to ours is a recent paper by Baraniuk et al. (2010). In that paper, model based sparsity was considered and the structures comes from the predefined models. It is important to note that some theoretical results were obtained there to show the effectiveness of their method in compressive sensing. Moreover a generic algorithmic template was presented for structured sparsity. A drawback of the template is that it relies on finding the pruning of residue or signal estimates to a subset of variables with small structured complexity. These steps have to be specifically designed for different data models under specialized assumptions. In this regard, while the algorithmic template is generic, the actual implementation for the pruning steps will be quite different for different types of structures (for example, see Cevher et al., 2009a,b). In other words, it does not provide a common scheme to represent their "models" for different structured sparsity data. Different structure representation schemes have to be built for different "models". It thus remains as an open issue how to develop a general theory for structured sparsity, together with a general algorithm based on a generic structure representation scheme that can be applied to a wide class of such problems. The Structured OMP algorithm, which is proposed in this paper, is an attempt to address this issue. Although each type of structures requires an appropriately chosen block set (see Section 3 and Section 4), the algorithmic implementation based on a generic structure representation scheme is the same for different structures. We note that in general it is much easier to pick an appropriate block set than to design a new pruning algorithm.

We see from the above discussion that there exists extensive literature on combining sparsity with structured priors, with empirical evidence showing that one can achieve better performance by imposing additional structures. However, it is still useful to establish a general theoretical framework for structured sparsity that can quantify its effectiveness, as well as an efficient algorithmic implementation. The goal of this paper is to develop such a general theory that addresses the following issues, where we pay special attention to the benefit of structured sparsity over the standard non-structured sparsity:

- quantifying structured sparsity;

- the minimal number of measurements required in compressive sensing;
- estimation accuracy under stochastic noise;
- an efficient algorithm that can solve a wide class of structured sparsity problems with meaningful sparse recovery performance bounds.

2. Coding Complexity Regularization

In structured sparsity, not all sparse patterns are equally likely. For example, in group sparsity, coefficients within the same group are more likely to be zeros or nonzeros simultaneously. This means that if a sparse coefficient vector's support set is consistent with the underlying group structure, then it is more likely to occur, and hence incurs a smaller penalty in learning. One contribution of this work is to formulate how to define structure on top of sparsity, and how to penalize each sparsity pattern. We then develop a theory for the corresponding penalized estimators (2) and (3).

2.1 Structured Sparsity and Coding Complexity

In order to formalize the idea of structured sparsity, we denote by $I = \{1,\ldots,p\}$ the index set of the coefficients. Consider any sparse subset $F \subset \{1,\ldots,p\}$, we assign a cost $\mathrm{cl}(F)$. In structured sparsity, the cost of F is an upper bound of the coding length of F (number of bits needed to represent F by a computer program) in a pre-chosen prefix coding scheme. It is a well-known fact in information theory (e.g., Cover and Thomas, 1991) that mathematically, the existence of such a coding scheme is equivalent to

$$\sum_{F\subset I} 2^{-\mathrm{cl}(F)} \leq 1.$$

From the Bayesian statistics point of view, $2^{-\mathrm{cl}(F)}$ can be regarded as a lower bound of the probability of F. The probability model of structured sparse learning is thus: first generate the sparsity pattern F according to probability $2^{-\mathrm{cl}(F)}$; then generate the coefficients in F.

Definition 1 *A cost function* $\mathrm{cl}(F)$ *defined on subsets of* I *is called a coding length (in base-2) if*

$$\sum_{F\subset I, F\neq\emptyset} 2^{-\mathrm{cl}(F)} \leq 1.$$

We give $\emptyset$ *a coding length 0. The corresponding structured sparse coding complexity of* F *is defined as*

$$c(F) = |F| + \mathrm{cl}(F).$$

A coding length $\mathrm{cl}(F)$ *is sub-additive if*

$$\mathrm{cl}(F\cup F') \leq \mathrm{cl}(F) + \mathrm{cl}(F'),$$

and a coding complexity $c(F)$ *is sub-additive if*

$$c(F\cup F') \leq c(F) + c(F').$$

Clearly if $\mathrm{cl}(F)$ is sub-additive, then the corresponding coding complexity $c(F)$ is also sub-additive. Note that for simplicity, we do not introduce a trade-off between $|F|$ and $\mathrm{cl}(F)$ in the definition of $c(F)$. However, in real applications, such a trade-off may be beneficial: for example we may define $c(F) = \gamma|F| + \mathrm{cl}(F)$, where γ is considered a tuning parameter in the algorithm.

Based on the structured coding complexity of subsets of I, we can now define the structured coding complexity of a sparse coefficient vector $\beta \in \mathbb{R}^p$.

Definition 2 *Giving a coding complexity $c(F)$, the structured sparse coding complexity of a coefficient vector $\beta \in \mathbb{R}^p$ is*

$$c(\beta) = \min\{c(F) : \mathrm{supp}(\beta) \subset F\}.$$

We will later show that if a coefficient vector β has a small coding complexity $c(\beta)$, then β can be effectively learned, with good in-sample prediction performance (in statistical learning) and reconstruction performance (in compressive sensing). In order to see why the definition requires adding $|F|$ to $\mathrm{cl}(F)$, we consider the generative model for structured sparsity mentioned earlier. In this model, the number of bits to encode a sparse coefficient vector is the sum of the number of bits to encode F (which is $\mathrm{cl}(F)$) and the number of bits to encode nonzero coefficients in F (this requires $O(|F|)$ bits up to a fixed precision). Therefore the total number of bits required is $\mathrm{cl}(F) + O(|F|)$. This information theoretical result translates into a statistical estimation result: without additional regularization, the learning complexity for least squares regression within any fixed support set F is $O(|F|)$. By adding the model selection complexity $\mathrm{cl}(F)$ for each support set F, we obtain an overall statistical estimation complexity of $O(\mathrm{cl}(F) + |F|)$. We would like to mention that the coding complexity approach in this paper is related to but extends the Union-of-Subspaces model of Lu and Do (2008), which corresponds to a hard assignment of $\mathrm{cl}(F)$ to be either a constant c or $+\infty$.

While the idea of using coding based penalization is clearly motivated by the minimum description length (MDL) principle, the actual penalty we obtain for structured sparsity problems is different from the standard MDL penalty for model selection. Moreover, our analysis differs from some other MDL based analysis (such as Haupt and Nowak, 2006) that only deals with minimization over a countably many candidate coefficients β (the candidates are chosen a priori). This difference is important in sparse learning, and analysis as in Haupt and Nowak (2006) cannot be applied to the estimators of (2) or (3). Therefore in order to prevent confusion, we avoid using MDL in our terminology. Nevertheless, one may consider our framework as a natural combination of the MDL idea and the modern sparsity analysis. We will consider detailed examples of $\mathrm{cl}(F)$ in Section 4.

2.2 Theory of Coding Complexity Regularization

We assume sub-Gaussian noise as follows.

Assumption 1 *Assume that $\{\mathbf{y}_i\}_{i=1,\ldots,n}$ are independent (but not necessarily identically distributed) sub-Gaussians: there exists a constant $\sigma \geq 0$ such that $\forall i$ and $\forall t \in \mathbb{R}$,*

$$\mathbb{E}_{\mathbf{y}_i} e^{t(\mathbf{y}_i - \mathbb{E}\mathbf{y}_i)} \leq e^{\sigma^2 t^2/2}.$$

Both Gaussian and bounded random variables are sub-Gaussian using the above definition. For example, if a random variable $\xi \in [a,b]$, then $\mathbb{E}_\xi e^{t(\xi - \mathbb{E}\xi)} \leq e^{(b-a)^2 t^2/8}$. If a random variable is Gaussian: $\xi \sim N(0,\sigma^2)$, then $\mathbb{E}_\xi e^{t\xi} \leq e^{\sigma^2 t^2/2}$.

The following property of sub-Gaussian noise is important in our analysis. Our simple proof yields a sub-optimal choice of the constants.

Proposition 3 *Let $P \in \mathbb{R}^{n\times n}$ be a projection matrix of rank k, and $\mathbf{y}$ satisfies Assumption 1. Then for all $\eta \in (0,1)$, with probability larger than $1-\eta$:*

$$\|P(\mathbf{y}-\mathbb{E}\mathbf{y})\|_2^2 \le \sigma^2[7.4k + 2.7\ln(2/\eta)].$$

We also need to generalize sparse eigenvalue condition, used in the modern sparsity analysis. It is related to (and weaker than) the RIP (restricted isometry property) assumption (Candes and Tao, 2005) in the compressive sensing literature. This definition takes advantage of coding complexity, and can be also considered as (a weaker version of) structured RIP. We introduce a definition.

Definition 4 *For all $F \subset \{1,\ldots,p\}$, define*

$$\rho_-(F) = \inf\left\{\frac{1}{n}\|X\beta\|_2^2/\|\beta\|_2^2 : \mathrm{supp}(\beta)\subset F\right\},$$
$$\rho_+(F) = \sup\left\{\frac{1}{n}\|X\beta\|_2^2/\|\beta\|_2^2 : \mathrm{supp}(\beta)\subset F\right\}.$$

Moreover, for all $s > 0$, define

$$\rho_-(s) = \inf\{\rho_-(F) : F \subset I, c(F) \le s\},$$
$$\rho_+(s) = \sup\{\rho_+(F) : F \subset I, c(F) \le s\}.$$

In the theoretical analysis, we need to assume that $\rho_-(s)$ is not too small for some s that is larger than the signal complexity. Since we only consider eigenvalues for submatrices with small cost $c(\beta)$, the sparse eigenvalue $\rho_-(s)$ can be significantly larger than the corresponding ratio for standard sparsity (which will consider all subsets of $\{1,\ldots,p\}$ up to size s). For example, for random projections used in compressive sensing applications, the coding length $c(\mathrm{supp}(\beta))$ is $O(k\ln p)$ in standard sparsity, but can be as low as $c(\mathrm{supp}(\beta)) = O(k)$ in structured sparsity (if we can guess $\mathrm{supp}(\beta)$ approximately correctly. Therefore instead of requiring $n = O(k\ln p)$ samples, we require only $O(k + \mathrm{cl}(\mathrm{supp}(\beta)))$. The difference can be significant when p is large and the coding length $\mathrm{cl}(\mathrm{supp}(\beta)) \ll k\ln p$. An example for this is group sparsity, where we have p/k_0 even sized groups, and variables in each group are simultaneously zero or nonzero. The coding length of the groups are $(k/k_0)\ln(p/k_0)$, which is significantly smaller than $k\ln p$ when p is large (see Section 4 for details).

More precisely, we have the following random projection sample complexity bound for the structured sparse eigenvalue condition. The theorem implies that the structured RIP condition is satisfied with sample size $n = O(k + (k/k_0)\ln(p/k_0))$ in group sparsity (where $s = O(k + (k/k_0)\ln(p/k_0))$) rather than $n = O(k\ln(p))$ in standard sparsity (where $s = O(k\ln p)$). For hierarchical tree sparsity (see Section 4 for details), it requires $n = O(k)$ examples (with $s = O(k)$), which matches the result of Baraniuk et al. (2010). Therefore Theorem 6 shows that in the compressive sensing applications, it is possible to reconstruct signals with fewer number of random projections by using structured sparsity.

Proposition 5 (Structured-RIP) *Suppose that elements in X are iid standard Gaussian random variables $N(0,1)$. For any $t > 0$ and $\delta \in (0,1)$, let*

$$n \ge \frac{8}{\delta^2}[\ln 3 + t + s\ln(1+8/\delta)].$$

Then with probability at least $1-e^{-t}$*, the random matrix* $X \in \mathbb{R}^{n\times p}$ *satisfies the following structured-RIP inequality for all vector* $\beta \in \mathbb{R}^p$ *with coding complexity no more than* s*:*

$$(1-\delta)\|\beta\|_2 \le \frac{1}{\sqrt{n}}\|X\beta\|_2 \le (1+\delta)\|\beta\|_2. \tag{4}$$

Although in the theorem, we assume Gaussian random matrix in order to state explicit constants, it is clear that similar results hold for other sub-Gaussian random matrices. Note that the proposed generalization of RIP extends related results in compressive sensing and statistics (Baraniuk et al., 2010; Huang and Zhang, 2010).

The following result gives a performance bound for constrained coding complexity regularization in (2). The 2-norm parameter estimation bound $\|\hat{\beta}-\beta\|_2$ requires that $\rho_-(\cdot) > 0$ (otherwise, the bound becomes trivial). For random design matrix X, the lower-bound in (4) is thus needed.

Theorem 6 *Suppose that Assumption 1 is valid. Consider any fixed target* $\beta \in \mathbb{R}^p$*. Then with probability exceeding* $1-\eta$*, for all* $\varepsilon \ge 0$ *and* $\hat{\beta} \in \mathbb{R}^p$ *such that:* $\hat{Q}(\hat{\beta}) \le \hat{Q}(\beta)+\varepsilon$*, we have*

$$\|X\hat{\beta}-\mathbb{E}\mathbf{y}\|_2 \le \|X\beta-\mathbb{E}\mathbf{y}\|_2 + \sigma\sqrt{2\ln(6/\eta)} + 2(7.4\sigma^2 c(\hat{\beta}) + 4.7\sigma^2\ln(6/\eta)+\varepsilon)^{1/2}.$$

Moreover, if the coding scheme $c(\cdot)$ *is sub-additive, then*

$$n\rho_-(c(\hat{\beta})+c(\beta))\|\hat{\beta}-\beta\|_2^2 \le 10\|X\beta-\mathbb{E}\mathbf{y}\|_2^2 + 37\sigma^2 c(\hat{\beta}) + 29\sigma^2\ln(6/\eta) + 2.5\varepsilon.$$

This theorem immediately implies the following result for (2): $\forall\beta$ such that $c(\beta) \le s$,

$$\frac{1}{\sqrt{n}}\|X\hat{\beta}_{constr}-\mathbb{E}\mathbf{y}\|_2 \le \frac{1}{\sqrt{n}}\|X\beta-\mathbb{E}\mathbf{y}\|_2 + \frac{\sigma}{\sqrt{n}}\sqrt{2\ln(6/\eta)} + \frac{2\sigma}{\sqrt{n}}(7.4s+4.7\ln(6/\eta))^{1/2},$$

$$\|\hat{\beta}_{constr}-\beta\|_2^2 \le \frac{1}{\rho_-(s+c(\beta))n}\left[10\|X\beta-\mathbb{E}\mathbf{y}\|_2^2 + 37\sigma^2 s + 29\sigma^2\ln(6/\eta)\right].$$

Although for simplicity this paper does not consider the problem of estimating $\rho_-(s+c(\beta))$, it is possible to estimate it approximately (for example, using ideas of d'Aspremont et al., 2008). We can generally expect $\rho_-(s+c(\beta)) = O(1)$ by assuming that the sample size is sufficiently large according to Proposition 5. The result immediately implies that as sample size $n\to\infty$ and $s/n \to 0$, the root mean squared error prediction performance $\|X\hat{\beta}-\mathbb{E}\mathbf{y}\|_2/\sqrt{n}$ converges to the optimal prediction performance $\inf_{c(\beta)\le s}\|X\beta-\mathbb{E}\mathbf{y}\|_2/\sqrt{n}$. This result is agnostic in that even if $\|X\beta-\mathbb{E}\mathbf{y}\|_2/\sqrt{n}$ is large, the result is still meaningful because it says the performance of the estimator $\hat{\beta}$ is competitive to the best possible estimator in the class $c(\beta)\le s$.

In compressive sensing applications, we take $\sigma = 0$, and we are interested in recovering β from random projections. For simplicity, we let $X\beta = \mathbb{E}\mathbf{y} = \mathbf{y}$, and our result shows that the constrained coding complexity penalization method achieves exact reconstruction $\hat{\beta}_{constr} = \beta$ as long as $\rho_-(2c(\beta)) > 0$ (by setting $s = c(\beta)$). According to Proposition 5, this is possible when the number of random projections (sample size) reaches $n = O(c(\beta))$. This is a generalization of corresponding results in compressive sensing (Candes and Tao, 2005). As we have pointed out earlier, this number can be significantly smaller than the standard sparsity requirement of $n = O(\|\beta\|_0\ln p)$, if the structure imposed in the formulation is meaningful.

As an example, for group sparsity (see Section 4), we consider m pre-defined groups, each of size k_0. If the support of β is covered by g of the m groups, we know from Section 4 that the

complexity can be defined as $s = g\log_2(2m) + gk_0$. In comparison, the standard sparsity complexity is given by $s = \|\beta\|_0 \log_2(2p)$, which may be significantly larger if $g \ll \|\beta\|_0$ (that is, the group structure is meaningful). It can be shown that the group-Lasso estimator may also achieve the group sparsity complexity of $s = g\log_2(2m) + gk_0$ (Huang and Zhang, 2010; Lounici et al., 2009), but the result for group-Lasso requires a stronger condition involving structured-RIP. Note that the first bound in Theorem 6 does not require any RIP assumption, while the second bound only requires a very weak dependency of the form $\rho_-(\cdot) > 0$. In contrast, the required dependency for group Lasso is significantly stronger, and details can be seen in Huang and Zhang (2010), Lounici et al. (2009) and Nardi and Rinaldo (2008). Although the result for the coding complexity estimator (2) is better due to weaker RIP dependency, we shall point out that it doesn't mean that for group sparsity, we should use (2) instead of group-Lasso in practice. This is because solving (2) requires non-convex optimization, while group-Lasso is a convex formulation. This is why we will consider an efficient algorithm to approximately solve (2) in Section 3.

Similar to Theorem 6, we can obtain the following result for (3). A related result for standard sparsity under Gaussian noise can be found in Bunea et al. (2007).

Theorem 7 *Suppose that Assumption 1 is valid. Consider any fixed target* $\beta \in \mathbb{R}^p$. *Then with probability exceeding* $1-\eta$, *for all* $\lambda > 7.4\sigma^2$ *and* $a \geq 7.4\sigma^2/(\lambda - 7.4\sigma^2)$, *we have*

$$\|X\hat{\beta}_{pen} - \mathbb{E}\mathbf{y}\|_2^2 \leq (1+a)^2\|X\beta - \mathbb{E}\mathbf{y}\|_2^2 + (1+a)\lambda c(\beta) + \sigma^2(10 + 5a + 7a^{-1})\ln(6/\eta).$$

Unlike the result for (2), the prediction performance $\|X\hat{\beta}_{pen} - \mathbb{E}\mathbf{y}\|_2$ of the estimator in (3) is competitive to $(1+a)\|X\beta - \mathbb{E}\mathbf{y}\|_2$, which is a constant factor larger than the optimal prediction performance $\|X\beta - \mathbb{E}\mathbf{y}\|_2$. By optimizing λ and a, it is possible to obtain a similar result as that of Theorem 6. However, this requires tuning λ, which is not as convenient as tuning s in (2). Note that both results presented here, and those in Bunea et al. (2007) are superior to the more traditional least squares regression results with λ explicitly fixed (for example, theoretical results for AIC). This is because one can only obtain the form presented in Theorem 6 by tuning λ. Such tuning is important in real applications.

3. Structured Greedy Algorithm

In this section, we describe a generalization of the OMP algorithm for standard sparsity. Our generalization, which we refer to as structured greedy algorithm or simply StructOMP, takes advantage of block structures to approximately solve the structured sparsity formulation (2). It would be worthwhile to mention that the notion of block structures here is different from block sparsity in model-based compressive sensing (Baraniuk et al., 2010).

Note that in this algorithm, we assume that $c(F)$ is relatively easy to compute (up to a constant) for any given F. For this purpose, we may use a relatively easy to compute upper bound of $c(F)$. For example, for graph structured sparsity described later in Section 4, we may simply use the right hand side of Proposition 11 as the definition of $c(F)$. If the maximum degree of a graph is small, we can simply use $c(F) = g\ln(p) + |F|$, where g is the number of connected components in F. For practical purposes, a multiplicative constant in the definition of $c(F)$ is not important because it can be absorbed into the tuning parameter s.

3.1 Algorithm Description

The main idea of StructOMP is to limit the search space of the greedy algorithm to small blocks. We will show that if a coding scheme can be approximated with blocks, then StructOMP is effective. Additional discussion of block approximation can be found in Section 4.

Formally, we consider a subset $\mathcal{B} \subset 2^I$. That is, each element (which we call a block or a base block) of $\mathcal{B}$ is a subset of I. We call $\mathcal{B}$ a block set if $I = \cup_{B\in\mathcal{B}} B$ and all single element sets $\{j\}$ belong to $\mathcal{B}$ ($j \in I$). Note that $\mathcal{B}$ may contain additional non single-element blocks. The requirement of $\mathcal{B}$ containing all single element sets is for notational convenience, as it implies that every subset $F \subset I$ can be expressed as the union of blocks in $\mathcal{B}$. Mathematically this requirement is non-important because we may simply assign ∞ coding length to single-element blocks, which is equivalent to excluding these single element sets.

Input: $(X, \mathbf{y})$, $\mathcal{B} \subset 2^I$, $s > 0$
Output: $F^{(k)}$ and $\beta^{(k)}$
let $F^{(0)} = \emptyset$ and $\beta^{(0)} = 0$
for $k = 1, 2, \ldots$
  select $B^{(k)} \in \mathcal{B}$ to maximize progress $\qquad (*)$
  let $F^{(k)} = B^{(k)} \cup F^{(k-1)}$
  let $\beta^{(k)} = \arg\min_{\beta\in\mathbb{R}^p} \hat{Q}(\beta)$ subject to $\mathrm{supp}(\beta) \subset F^{(k)}$
  if $(c(\beta^{(k)}) > s)$ **break**
end

Figure 1: Structured Greedy Algorithm

In Figure 1, we are given a set of blocks $\mathcal{B}$ that contains subsets of I. Instead of searching all subsets $F \subset I$ up to a certain complexity $|F| + c(F)$, which is computationally infeasible, we search only the blocks restricted to $\mathcal{B}$. It is assumed that searching over $\mathcal{B}$ is computationally manageable. In practice, the computational cost is linear in the number of base blocks $|\mathcal{B}|$.

At each step $(*)$, we try to find a block from $\mathcal{B}$ to maximize progress. It is thus necessary to define a quantity that measures progress. Our idea is to approximately maximize the gain ratio:

$$\frac{\hat{Q}(\beta^{(k-1)}) - \hat{Q}(\beta^{(k)})}{c(\beta^{(k)}) - c(\beta^{k-1})},$$

which measures the reduction of objective function per unit increase of coding complexity. This greedy criterion is a natural generalization of the standard greedy algorithm, and essential in our analysis. For least squares regression, we can define the gain ratio as follows:

$$\phi(B) = \frac{\|P_{B-F^{(k-1)}}(X\beta^{(k-1)} - \mathbf{y})\|_2^2}{c(B \cup F^{(k-1)}) - c(F^{(k-1)})}, \tag{5}$$

where

$$P_F = X_F (X_F^\top X_F)^+ X_F^\top$$

is the projection matrix to the subspaces generated by columns of X_F. Here $(X_F^\top X_F)^+$ denotes the Moore-Penrose pseudo-inverse.

More precisely, for least squares regression, at each step $(*)$ of Figure 1, we select a block $B^{(k)}$ that satisfies the condition

$$\phi(B^{(k)}) \geq \nu \max_{B \in \mathcal{B}} \phi(B) \tag{6}$$

for some $\nu \in (0,1]$. We may regard ν as a fixed approximation ratio (to ensure the quality of approximate optimization) that will appear in our analysis, although the algorithm does not have to pick ν a priori.

The reason to allow approximate maximization in (6) is that our practical implementation of StructOMP maximizes a simpler quantity

$$\tilde{\phi}(B) = \frac{\|X^\top_{B-F^{(k-1)}}(X\beta^{(k-1)} - \mathbf{y})\|_2^2}{c(B \cup F^{(k-1)}) - c(F^{(k-1)})}, \tag{7}$$

which is more efficient to compute (especially when blocks are overlapping). Since the ratio

$$\|X^\top_{B-F^{(k-1)}}\mathbf{r}\|_2^2 / \|P_{B-F^{(k-1)}}\mathbf{r}\|_2^2$$

is bounded between $\rho_+(B)$ and $\rho_-(B)$ (these quantities are defined in Definition 4), we know that maximization of $\tilde{\phi}(B)$ leads to an approximate maximization of $\phi(B)$ with $\nu \geq \rho_-(B)/\rho_+(B)$. That is, maximization of (7) in our practical StructOMP implementation corresponds to an approximate maximization in (6). Moreover, ν only appears in our analysis, and it does not appear explicitly in our implementation.

Note that we shall ignore $B \in \mathcal{B}$ such that $B \subset F^{(k-1)}$, and just let the corresponding gain to be 0. Moreover, if there exists a base block $B \not\subset F^{(k-1)}$ but $c(B \cup F^{(k-1)}) \leq c(F^{(k-1)})$, we can always select B and let $F^{(k)} = B \cup F^{(k-1)}$ (this is because it is always beneficial to add more features into $F^{(k)}$ without additional coding complexity). We assume this step is always performed if such a $B \in \mathcal{B}$ exists. The non-trivial case is $c(B \cup F^{(k-1)}) > c(F^{(k-1)})$ for all $B \in \mathcal{B}$; in this case both $\phi(B)$ and $\tilde{\phi}(B)$ are well defined.

3.2 Convergence Analysis

It is important to understand that the block structure is only used to limit the search space in the structured greedy algorithm. However, our theoretical analysis shows that if in addition, the underlying coding scheme can be approximated by block coding using base blocks employed in the greedy algorithm, then the algorithm is effective in minimizing (2). Although one does not need to know the specific approximation in order to use the greedy algorithm, knowing its existence (which can be shown for the examples discussed in Section 4) guarantees the effectiveness of the algorithm. It is also useful to understand that our result does not imply that the algorithm won't be effective if the actual coding scheme cannot be approximated by block coding.

We shall introduce a definition before stating our main results.

Definition 8 *Given $\mathcal{B} \subset 2^I$, define*

$$\rho_0(\mathcal{B}) = \max_{B \in \mathcal{B}} \rho_+(B), \qquad c_0(\mathcal{B}) = \max_{B \in \mathcal{B}} c(B)$$

and

$$c(\beta, \mathcal{B}) = \min \left\{ \sum_{j=1}^{b} c(B_j) : \mathrm{supp}(\beta) \subset \bigcup_{j=1}^{b} B_j \quad (B_j \in \mathcal{B}); b \geq 1 \right\}.$$

The following theorem shows that if $c(\beta, \mathcal{B})$ is small, then one can use the structured greedy algorithm to find a coefficient vector $\beta^{(k)}$ that is competitive to β, and the coding complexity $c(\beta^{(k)})$ is not much worse than that of $c(\beta, \mathcal{B})$. This implies that if the original coding complexity $c(\beta)$ can be approximated by block complexity $c(\beta, \mathcal{B})$, then we can approximately solve (2).

Theorem 9 *Suppose the coding scheme is sub-additive. Consider* β *and* ε *such that*

$$\varepsilon \in (0, \|\mathbf{y}\|_2^2 - \|X\beta - \mathbf{y}\|_2^2]$$

and

$$s \geq \frac{\rho_0(\mathcal{B}) c(\beta, \mathcal{B})}{\nu \rho_-(s + c(\beta))} \ln \frac{\|\mathbf{y}\|_2^2 - \|X\beta - \mathbf{y}\|_2^2}{\varepsilon}.$$

Then at the stopping time k, we have

$$\hat{Q}(\beta^{(k)}) \leq \hat{Q}(\beta) + \varepsilon.$$

By Theorem 6, the result in Theorem 9 implies that

$$\|X\beta^{(k)} - \mathbb{E}\mathbf{y}\|_2 \leq \|X\beta - \mathbb{E}\mathbf{y}\|_2 + \sigma\sqrt{2\ln(6/\eta)} + 2\sigma\sqrt{7.4(s + c_0(\mathcal{B})) + 4.7\ln(6/\eta) + \varepsilon/\sigma^2},$$
$$\|\beta^{(k)} - \beta\|_2^2 \leq \frac{10\|X\beta - \mathbb{E}\mathbf{y}\|_2^2 + 37\sigma^2(s + c_0(\mathcal{B})) + 29\sigma^2\ln(6/\eta) + 2.5\varepsilon}{\rho_-(s + c_0(\mathcal{B}) + c(\beta))n}.$$

The result shows that in order to approximate a signal β up to accuracy ε, one needs to use coding complexity $O(\ln(1/\varepsilon))c(\beta, \mathcal{B})$. Now, consider the case that $\mathcal{B}$ contains small blocks and their sub-blocks with equal coding length, and the actual coding scheme can be approximated (up to a constant) by block coding generated by $\mathcal{B}$; that is, $c(\beta, \mathcal{B}) = O(c(\beta))$. In this case we need $O(s\ln(1/\varepsilon))$ to approximate a signal with coding complexity s. For this reason, we will extensively discuss block approximation in Section 4.

In order to improve forward greedy procedures, backward greedy strategies can be employed, as shown in various recent works such as Zhang (2011). For simplicity, we will not analyze such strategies in this paper. It is worth mentioning that in practice, greedy algorithm is often adequate. In particular the $O(\ln(1/\varepsilon))$ factor vanishes for a weakly sparse target signal β, where the magnitude of its coefficients gradually decrease to zero. This concept has been considered in previous work such as Donoho (2006) and Baraniuk et al. (2010). In such case, we may choose an appropriate optimal stopping point to avoid the $O(\ln(1/\varepsilon))$ factor. In fact, practitioners often observe that OMP can be more effective than Lasso for weakly sparse target signals (in spite of stronger theoretical results for Lasso with strongly sparse target signals). This will be confirmed in our experiments as well. Without cluttering the main text, we leave the detailed analysis of StructOMP for weakly sparse signals to Appendix F. Our analysis is the first theoretical justification of this empirical phenomenon.

4. Structured Sparsity Examples

Before giving detailed examples, we describe a general coding scheme called *block coding*, which is an expansion of Definition 8. The basic idea of block coding is to define a coding scheme on

a small number of base blocks (a block is a subset of I), and then define a coding scheme on all subsets of I using these base blocks.

Consider block set $\mathcal{B} \subset 2^I$. We assume that every subset $F \subset I$ can be expressed as the union of blocks in $\mathcal{B}$. Let cl_0 be a code length on $\mathcal{B}$:

$$\sum_{B \in \mathcal{B}} 2^{-\mathrm{cl}_0(B)} \leq 1,$$

we define $\mathrm{cl}(B) = \mathrm{cl}_0(B) + 1$ for $B \in \mathcal{B}$. It not difficult to show that the following cost function on $F \subset I$ is a coding length

$$\mathrm{cl}_{\mathcal{B}}(F) = \min \left\{ \sum_{j=1}^{b} \mathrm{cl}(B_j) : F = \bigcup_{j=1}^{b} B_j \quad (B_j \in \mathcal{B}) \right\}.$$

This is because

$$\sum_{F \subset I, F \neq \emptyset} 2^{-\mathrm{cl}(F)} \leq \sum_{b \geq 1} \sum_{B_\ell \in \mathcal{B}: 1 \leq \ell \leq b} 2^{-\sum_{\ell=1}^{b} \mathrm{cl}(B_\ell)} \leq \sum_{b \geq 1} \prod_{\ell=1}^{b} \sum_{B_\ell \in \mathcal{B}} 2^{-\mathrm{cl}(B_\ell)} \leq \sum_{b \geq 1} 2^{-b} = 1.$$

We call the coding scheme $\mathrm{cl}_{\mathcal{B}}$ block coding. It is clear from the definition that block coding is sub-additive.

From Theorem 9 and the discussions thereafter, we know that under appropriate conditions, a target coefficient vector with a small block coding complexity can be approximately learned using the structured greedy algorithm. This means that the block coding scheme has important algorithmic implications. That is, if a coding scheme can be approximated by block coding with a small number of base blocks, then the corresponding estimation problem can be approximately solved using the structured greedy algorithm.

For this reason, we shall pay special attention to block coding approximation schemes for examples discussed below. In particular, a coding scheme $\mathrm{cl}(\cdot)$ can be polynomially approximated by block coding if there exists a block coding scheme $\mathrm{cl}_{\mathcal{B}}$ with polynomial (in p) number of base blocks in $\mathcal{B}$, such that there exists a positive constant $C_{\mathcal{B}}$ independent of p:

$$\mathrm{cl}_{\mathcal{B}}(F) \leq C_{\mathcal{B}} \mathrm{cl}(F).$$

That is, up to a constant, the block coding scheme $\mathrm{cl}_{\mathcal{B}}()$ is dominated by the coding scheme $\mathrm{cl}()$.

While it is possible to work with blocks with non-uniform coding schemes, for simplicity examples provided in this paper only consider blocks with uniform coding, which is similar to the representation used in the Union-of-Subspaces model of Lu and Do (2008).

4.1 Standard Sparsity

A simple coding scheme is to code each subset $F \subset I$ of cardinality k using $k \log_2(2p)$ bits, which corresponds to block coding with $\mathcal{B}$ consisted only of single element sets, and each base block has a coding length $\mathrm{cl}_0 = \log_2 p$. This corresponds to the complexity for the standard sparse learning.

A more general version is to consider single element blocks $\mathcal{B} = \{\{j\} : j \in I\}$, with a non-uniform coding scheme $\mathrm{cl}_0(\{j\}) = c_j$, such that $\sum_j 2^{-c_j} \leq 1$. It leads to a non-uniform coding length on I as

$$\mathrm{cl}(B) = |B| + \sum_{j \in B} c_j.$$

In particular, if a feature j is likely to be nonzero, we should give it a smaller coding length c_j, and if a feature j is likely to be zero, we should give it a larger coding length. In this case, a subset $F \subset I$ has coding length $\mathrm{cl}(F) = \sum_{j \in F}(1 + c_j)$.

4.2 Group Sparsity

The concept of group sparsity has appeared in various recent work, such as the group Lasso in Yuan and Lin (2006) or multi-task learning in Argyriou et al. (2008). Consider a partition of $I = \cup_{j=1}^{m} G_j$ into m disjoint groups. Let $\mathcal{B}_G$ contain the m groups $\{G_j\}$, and $\mathcal{B}_1$ contain p single element blocks. The strong group sparsity coding scheme is to give each element in $\mathcal{B}_1$ a code-length cl_0 of ∞, and each element in $\mathcal{B}_G$ a code-length cl_0 of $\log_2 m$. Then the block coding scheme with blocks $\mathcal{B} = \mathcal{B}_G \cup \mathcal{B}_1$ leads to group sparsity, which only looks for signals consisted of the groups. The resulting coding length is: $\mathrm{cl}(B) = g \log_2(2m)$ if B can be represented as the union of g disjoint groups G_j; and $\mathrm{cl}(B) = \infty$ otherwise.

Note that if the support of the target signal F can be expressed as the union of g groups, and each group size is k_0, then the group coding length $g \log_2(2m)$ can be significantly smaller than the standard sparsity coding length of $|F| \log_2(2p) = g k_0 \log_2(2p)$. As we shall see later, the smaller coding complexity implies better learning behavior, which is essentially the advantage of using group sparse structure. It was shown by Huang and Zhang (2010) that strong group sparsity defined above also characterizes the performance of group Lasso. Therefore if a signal has a pre-determined group structure, then group Lasso is superior to the standard Lasso.

An extension of this idea is to allow more general block coding length for $\mathrm{cl}_0(G_j)$ and $\mathrm{cl}_0(\{j\})$ so that

$$\sum_{j=1}^{m} 2^{-\mathrm{cl}_0(G_j)} + \sum_{j=1}^{p} 2^{-\mathrm{cl}_0(\{j\})} \leq 1.$$

This leads to non-uniform coding of the groups, so that a group that is more likely to be nonzero is given a smaller coding length. If feature set F can be represented as the union of g groups $G_{j_1}, \ldots, G_{j_g}$, then its coding length is $\mathrm{cl}(F) = g + \sum_{j=1}^{g} \mathrm{cl}_0(G_j)$.

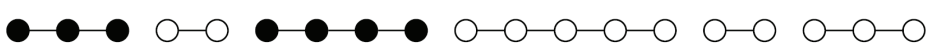


Figure 2: Group sparsity: nodes are variables, and black nodes are selected variables

Group sparsity is a special case of graph sparsity discussed below. Figure 2 shows an example of group sparsity, where the variables are represented by nodes, and the selected variables are represented by black nodes. Each pre-defined group is represented as a connected components in the graph, and the example contains six groups. Two groups, the first and the third from the left, are selected in the example. The number of selected variables (black nodes) is seven. Therefore we have $g = 2$ and $|F| = 7$. If we encode each group uniformly, then the coding length is $\mathrm{cl}(F) = 2\log_2(12)$.

4.3 Hierarchical Sparsity

One may also create a hierarchical group structure. A simple example is wavelet coefficients of a signal (Mallat, 1999). Another simple example is a binary tree with the variables as leaves, which we describe below. Each internal node in the tree is associated with three options: only left child, only right child, or both children; each option can be encoded in $\log_2 3$ bits.

Given a subset $F \subset I$, we can go down from the root of the tree, and at each node, decide whether only left child contains elements of F, or only right child contains elements of F, or both children contain elements of F. Therefore the coding length of F is $\log_2 3$ times the total number of internal nodes leading to elements of F. Since each leaf corresponds to no more than $\log_2 p$ internal nodes, the total coding length is no worse than $\log_2 3 \log_2 p|F|$. However, the coding length can be significantly smaller if nodes are close to each other or are clustered. In the extreme case, when the nodes are consecutive, we have $O(|F| + \log_2 p)$ coding length. More generally, if we can order elements in F as $F = \{j_1, \ldots, j_q\}$, then the coding length can be bounded as $\mathrm{cl}(F) = O(|F| + \log_2 p + \sum_{s=2}^{q} \log_2 \min_{\ell<s} |j_s - j_\ell|)$.

If all internal nodes of the tree are also variables in I (for example, in the case of wavelet decomposition), then one may consider feature set F with the following property: if a node is selected, then its parent is also selected. This requirement is very effective in wavelet compression, and often referred to as the zero-tree structure (Shapiro, 1993). Similar requirements have also been applied in statistics (Zhao et al., 2009) for variable selection and in compressive sensing (Baraniuk et al., 2010). The argument presented in this section shows that if we require F to satisfy the zero-tree structure, then its coding length is at most $O(|F|)$, without any explicit dependency on the dimensionality p. This is because one does not have to reach a leave node. Figure 3 shows an example of hierarchical sparsity, where the nodes of the tree are variables, and black nodes indicate those variables that are selected. The total number of selected variables (number of black nodes) is $|F| = 8$. This example obeys the requirement that if a node is selected, then its parent is also selected. Therefore the complexity is measured by $O(|F|)$.

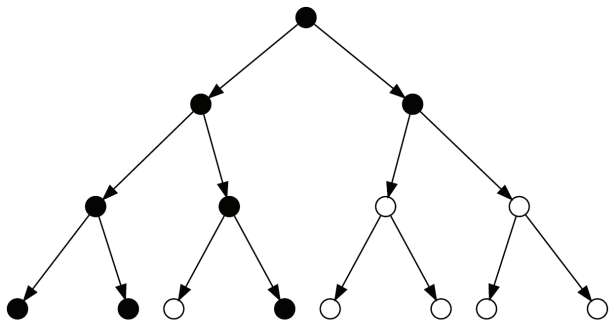


Figure 3: Hierarchical sparsity: nodes are variables, and black nodes are selected variables

The tree-based coding scheme discussed in this section can be polynomially approximated by block coding using no more than $p^{1+\delta}$ base blocks ($\delta > 0$). The idea is similar to that of the image coding example in the more general graph sparsity scheme which we discuss next.

4.4 Graph Sparsity

We consider a generalization of the hierarchical and group sparsity ideas by employing a (directed or undirected) graph structure G on I. To the best of our knowledge, this general structure has not been considered in any previous work.

In graph sparsity, each variable (an element of I) is a node of G but G may also contain additional nodes that are not variables. In order to take advantage of the graph structure, we favor connected regions (that is, nodes that are grouped together with respect to the graph structure). The following result defines a coding length on graphs based on the underlying graph structure. We leave its analysis to Appendix A.

Proposition 10 *Let G be a graph with maximum degree d_G. There exists a constant $C_G \leq 2\log_2(1+d_G)$ such that for any probability distribution q on G ($\sum_{v\in G} q(v) = 1$ and $q(v) \geq 0$ for $v \in G$), the following quantity (which we call graph coding) is a coding length on 2^G:*

$$\mathrm{cl}(F) = C_G|F| + g - \sum_{j=1}^{g} \max_{v\in F_j} \log_2(q(v)),$$

where $F \subset 2^G$ can be decomposed into the union of g connected components $F = \cup_{j=1}^{g} F_j$.

Note that graph coding is sub-additive. As a concrete example, we consider image processing, where each image is a rectangle of pixels (nodes); each pixel is connected to four adjacent pixels, which forms the underlying graph structure. We may take $q(v) = 1/p$ for all $v \in G$, where $p = |G|$ is the number of variables. Proposition 10 implies that if F is composed of g connected regions, then the coding length is $g\log_2(2p) + 2\log_2(5)|F|$, which can be significantly better than standard sparse coding length of $|F|\log_2(2p)$. For example, Figure 4 shows an image grid, where nodes are variables and selected variables are denoted by black nodes. In this example, the selected variables have two connected components (that is, $g = 2$): one in the top-left part, and the other in the bottom-right part of the grid. The total number of selected variables (the number of black nodes) is $|F| = 11$.

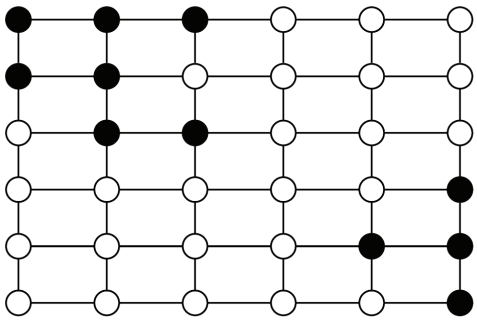


Figure 4: Graph sparsity: nodes are variables, and black nodes are selected variables

Note that group sparsity is a special case of graph sparsity, where each group is one connected region, as shown in Figure 2. We may also link adjacent groups to form the more general line-structured sparsity as in Figure 2. The advantage of line structure over group structure is that we do

not need to know the specific group divisions a priori as in Figure 2. From Proposition 10, similar coding complexity can be obtained as long as F can be covered by a small number of connected regions. Tree-structured hierarchical sparsity is also a special case of graph sparsity with a single connected region containing the root (we may take $q(\text{root}) = 1$). In fact, one may generalize this concept as follows. We consider a special case of sparse sparsity where we limit F to be a connected region that contains a fixed starting node v_0. We can simply let $q(v_0) = 1$, and the coding length of F is $O(|F|)$, which is independent of the dimensionality p. This generalizes the similar claim for the zero-tree structure described earlier.

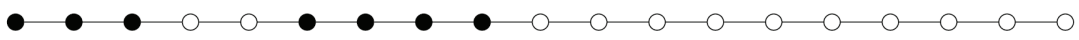


Figure 5: Line-structured sparsity: nodes are variables, and black nodes are selected variables

The following result shows that under uniform encoding of the nodes $q(v) = 1/p$ for $v \in G$, general graph coding schemes can be polynomially approximated with block coding. The idea is to consider relatively small sized base blocks consisted of nodes that are close together with respect to the graph structure, and then use the induced block coding scheme to approximate the graph coding.

Proposition 11 *Let G be a graph with maximum degree d_G, and $p = |G|$. Consider any number $\delta > 0$ such that $L = \delta \log_2 p$ is an even integer. Let $\mathcal{B}$ be the set of connected nodes of size up to L; that is, $B \in \mathcal{B}$ is a connected region in G such that $|B| \leq L$. Then there exists a constant $C_G \leq 2\log_2(1+d_G)$, such that $|\mathcal{B}| \leq p^{1+C_G\delta}$. If we consider the uniform code-length $\text{cl}_0(B) = (1+C_G\delta)\log_2 p$ for all $B \in \mathcal{B}$, then the induced block-coding scheme $\text{cl}_{\mathcal{B}}$ satisfies*

$$\text{cl}_{\mathcal{B}}(F) \leq g(1+C_G\delta)\log_2 p + 2(C_G+\delta^{-1})|F|.$$

where g is the number of connected regions in F.

The result means that graph sparsity can be polynomially approximated with a block coding scheme if we let $q(v) = 1/p$ for all $v \in G$. As we have pointed out, block approximation is useful because the latter is required in the structured greedy algorithm which we propose in this paper.

Note that a refined result holds for hierarchical sparsity (where we have $q(\text{root}) = 1$) using block approximation that does not explicitly depend on $\log_2 p$. In this case, for each tree depth $\ell = 1, 2, 3, \ldots$, we can restrict the underlying tree upto depth ℓ, and apply Proposition 11 on the restricted tree. Using this idea, the coding length for F depends explicitly on the maximum depth of F in the tree instead of $\log_2 p$.

4.5 Random Field Sparsity

Let $z_j \in \{0,1\}$ be a random variable for $j \in I$ that indicates whether j is selected or not. The most general coding scheme is to consider a joint probability distribution of $z = [z_1, \ldots, z_p]$. The coding length for F can be defined as $-\log_2 p(z_1, \ldots, z_p)$ with $z_j = I(j \in F)$ indicating whether $j \in F$ or not.

Such a probability distribution can often be conveniently represented as a binary random field on an underlying graph. In order to encourage sparsity, on average, the marginal probability $p(z_j)$

should take 1 with probability close to $O(1/p)$, so that the expected number of j's with $z_j = 1$ is $O(1)$. For disconnected graphs (z_j are independent), the variables z_j are iid Bernoulli random variables with probability $1/p$ being one. In this case, the coding length of a set F is $|F|\log_2(p) - (p-|F|)\log_2(1-1/p) \approx |F|\log_2(p)+1$. This is essentially the probability model for the standard sparsity scheme. In a more sophisticated situation, one may also let $E(z_j)$ to grow with sample size n. This is useful in non-parametric statistics.

We note that random field model has been considered in Cevher et al. (2009a). For many such models, it is possible to approximate a general random field coding scheme with block coding by using approximation methods in the graphical model literature. However, such approximations are problem specific, and the details are beyond the scope of this paper.

5. Experiments

The purpose of these experiments is to demonstrate the advantage of structured sparsity over standard sparsity. We compare the proposed StructOMP to OMP and Lasso, which are standard algorithms to achieve sparsity but without considering structure (Tibshirani, 1996; Tropp and Gilbert, 2007). For graph sparsity, the choice of $c(F)$ is simply $c(F) = g\log_2 p + |F|$, where g is the number of connected regions of F. This is adequate based on the discussion in Section 3. However, as pointed out after Definition 1, a better method is to use $c(F) = g\log_2 p + \gamma|F|$, where we tune γ appropriately. We observe that in practice, such tuning often improves performance. Nevertheless, in our experiments, we only report results with fixed $\gamma = 1$ for simplicity. This also means our experiments only demonstrate the advantage of StructOMP very conservatively without fine-tuning. The base blocks used in StructOMP are described in each experiment. Parameters (such as s in StructOMP or λ in Lasso) are tuned by cross-validation on the training data. We test various aspects of our theory to check whether the experimental results are consistent with the theory. Although in order to fully test the theory, one should also verify the RIP (or structured RIP) assumptions, in practice this is difficult to check precisely (however, it is possible to verify it approximately using ideas of d'Aspremont et al., 2008). Therefore in the following, we shall only study whether the experimental results are consistent with what can be expected from our theory, without verifying the detailed assumptions. The experimental protocols follow the setup of compressive sensing, where the original signals are projected using random projections, with noise added. Our goal is to recover the original signals from the noise corrupted projections.

In the experiments, we use Lasso-modified least angle regression (LARS/Lasso) as the solver of Lasso (B. Efron and Tibshirani, 2004). In order to quantitatively compare performance of different algorithms, we use recovery error, defined as the relative difference in 2-norm between the estimated sparse coefficient vector $\hat{\beta}_{est}$ and the ground-truth sparse coefficient β: $\|\hat{\beta}_{est} - \beta\|_2/\|\beta\|_2$. Our experiments focus on graph sparsity, with several different underlying graph structures. Note that graph sparsity is more general than group sparsity; in fact connected regions may be regarded as dynamic groups that are not pre-defined. However, for illustration, we include a comparison with group Lasso using some 1D simulated examples, where the underlying structure can be more easily approximated by pre-defined groups. Since additional experiments involving more complicated structures are more difficult to approximate by pre-defined groups, we exclude group-Lasso in those experiments.

All experiments were conducted on a 2.4GHz PC in Matlab. The code for our implementation of StructOMP can be obtained from `http://ranger.uta.edu/~huang/Downloads.htm`. In the

simulation experiments, we use k to denote the sparsity (number of nonzeros) of the true signal, and this should not be confused with the number of iterations k which we used earlier in the description of the StructOMP algorithm.

5.1 Simulated 1D Signals with Line-Structured Sparsity

In the first experiment, we randomly generate a $1D$ structured sparse signal with values ± 1, where data dimension $p = 512$, sparsity number $k = 64$ and group number $g = 4$. The support set of these signals is composed of g connected regions. Here, each component of the sparse coefficient is connected to two of its adjacent components, which forms the underlying graph structure. The graph sparsity concept introduced earlier is used to compute the coding length of sparsity patterns in StructOMP. The projection matrix X is generated by creating an $n \times p$ matrix with i.i.d. draws from a standard Gaussian distribution $N(0,1)$. For simplicity, the rows of X are normalized to unit magnitude. Zero-mean Gaussian noise with standard deviation $\sigma = 0.01$ is added to the measurements. Our task is to compare the recovery performance of StructOMP to those of OMP, Lasso and group Lasso for these structured sparsity signals under the framework of compressive sensing.

Figure 6 shows one instance of generated signal and the corresponding recovered results by different algorithms when $n = 160$. Since the sample size n is not big enough, OMP and Lasso do not achieve good recovery results, whereas the StructOMP algorithm achieves near perfect recovery of the original signal. We also include group Lasso in this experiment for illustration. We use predefined consecutive groups that do not completely overlap with the support of the signal. Since we do not know the correct group size, we just try group Lasso with several different group sizes (gs=2, 4, 8, 16). Although the results obtained with group Lasso are better than those of OMP and Lasso, they are still inferior to the results with StructOMP. As mentioned, this is because the pre-defined groups do not completely overlap with the support of the signal, which reduces the efficiency. In StructOMP, the base blocks are simply small connected line segments of size gs=3: that is, one node plus its two neighbors. This choice is only for simplicity, and it already produces good results in our experiments. If we include larger line segments into the base blocks (e.g., segments of size gs=4,5, etc), one can expect even better performance from StructOMP.

To study how the sample size n effects the recovery performance, we vary the sample size and record the recovery results by different algorithms. To reduce the randomness, we perform the experiment 100 times for each sample size. Figure 7(a) shows the recovery performance in terms of Recovery Error and Sample Size, averaged over 100 random runs for each sample size. As expected, StructOMP is better than the group Lasso and far better than the OMP and Lasso. The results show that the proposed StructOMP can achieve better recovery performance for structured sparsity signals with less samples. Figure 7(b) shows the recovery performance in terms of CPU Time and Sample Size, averaged over 100 random runs for each sample size. The computation complexities of StructOMP and OMP are far lower than those of Lasso and Group Lasso.

It is worth noting that the performance of StructOMP is less stable than the other algorithms when the sample size n is small. This is because for randomly generated design matrix, the structured RIP condition is only satisfied probabilistically. For small n, the necessary structured RIP condition can be violated with relatively large probability, and in such case StructOMP does not have much advantage (at least theoretically). This implies the relatively large variance. The effect is much less noticeable with weakly sparse signal in Figure 11(a) because the necessary structured

RIP condition is easier to satisfied for weakly sparse signals (based on our theory). Therefore the experimental results are consistent with our theory.

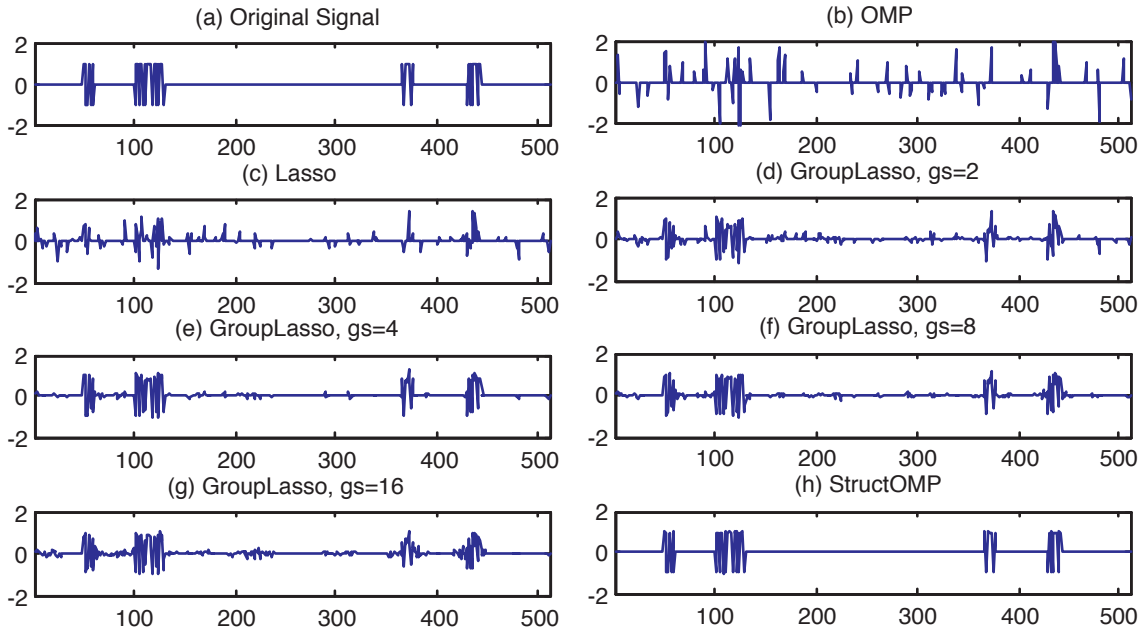
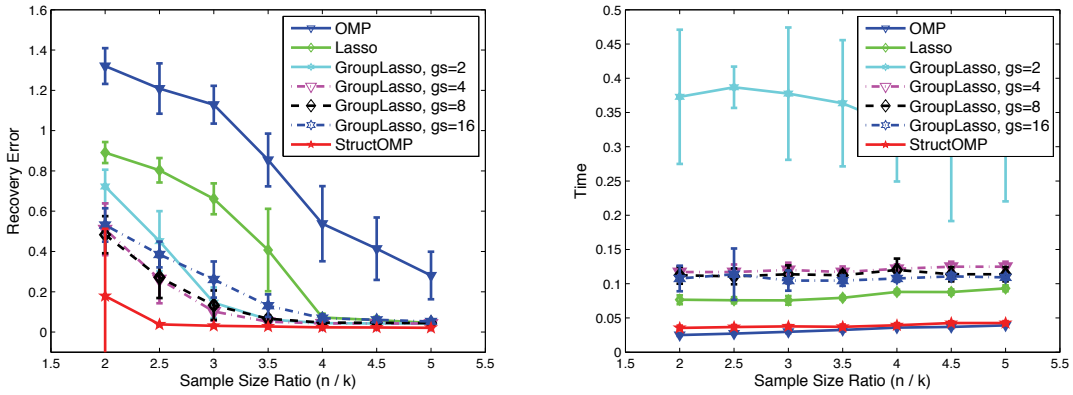


Figure 6: Recovery results of 1D signal with strongly line-structured sparsity. (a) original data; (b) recovered results with OMP (error is 0.9921); (c) recovered results with Lasso (error is 0.8660);; (d) recovered results with Group Lasso (error is 0.4832 with group size gs=2); (e) recovered results with Group Lasso (error is 0.4832 with group size gs=4);(f) recovered results with Group Lasso (error is 0.2646 with group size gs=8);(g) recovered results with Group Lasso (error is 0.3980 with group size gs=16); (h) recovered results with StructOMP (error is 0.0246).

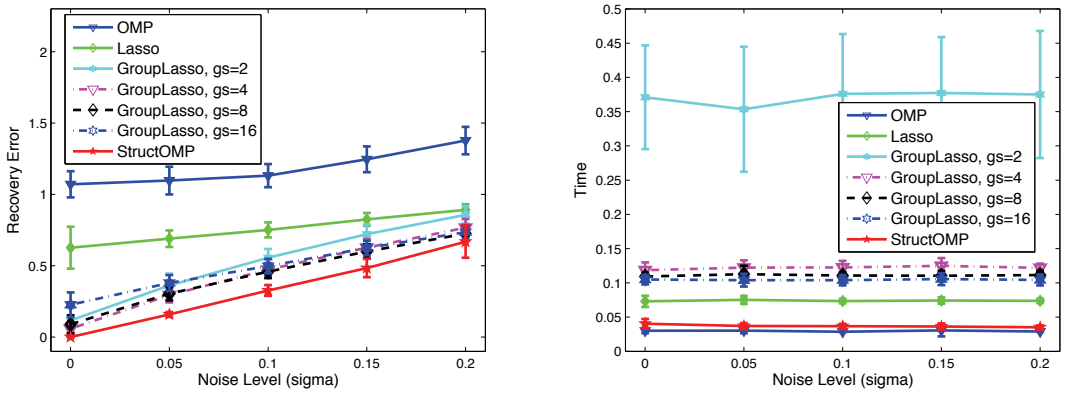
To study how the additive noise affects the recovery performance, we adjust the noise power σ and then record the recovery results by different algorithms. In this case, we fix the sample size at $n = 3k = 192$, and perform the experiment 100 times for each noise level tested. Figure 8(a) shows the recovery performance in terms of Recovery Error and Noise Level, averaged over 100 random runs for each noise level. As expected, StructOMP is also better than the group Lasso and far better than the OMP and Lasso. Figure 8(b) shows the recovery performance in terms of CPU Time and Noise Level, averaged over 100 random runs for each sample size. The computational complexities of StructOMP and OMP are lower than those of Lasso and Group Lasso.

To further study the performance of the StructOMP, we also compare it to two other methods for structured sparsity including OverlapLasso (Jacob et al., 2009) and ModelCS (Baraniuk et al., 2010) using the implementations available from the web. For fair comparisons, the same structures are used in OverlapLasso, ModelCS and StructOMP. As mentioned before, in these experiments, we use small connected line segments of size gs=3 (including one node plus its two neighbors) as base blocks in StructOMP. Therefore in OverlapLasso, the groups are also connected line segments of size gs=3; in ModelCS, this structure leads to the model assumption that if one node is nonzero, then its two neighbors has a high probability of being nonzeros. Figure 9(a) shows the recovery



(a) (b)

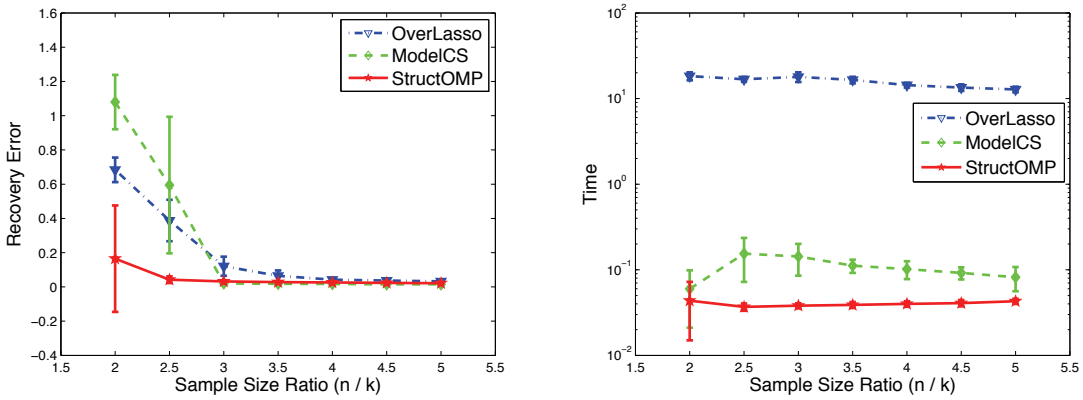
Figure 7: Recovery performance: (a) Recovery Error vs. Sample Size Ratio (n/k); (b) CPU Time vs. Sample Size Ratio (n/k)



(a) (b)

Figure 8: Recovery performance in terms of Noise Levels: (a) Recovery Error vs. Noise Level; (b) CPU Time vs. Noise Level

performance in terms of Recovery Error and Sample Size, averaged over 100 random runs for each sample size. At least for this problem, StructOMP achieves better performance than OverlapLasso and ModelCS, which shows that the proposed StructOMP algorithm can achieve better recovery performance than other structured sparsity algorithms for some problems. Figure 9(b) shows the recovery performance in terms of CPU Time and Sample Size, averaged over 100 random runs for each sample size. Although it is difficult to see from the figure, the computational complexity of StructOMP is lower than that of ModelCS (about half CPU time) and are far lower than that of OverlapLasso, at least based on the implementation of Jacob et al. (2009).



(a) (b)

Figure 9: Performance Comparisons between methods related with structured sparsity(OverlapLasso (Jacob et al., 2009), ModelCS (Baraniuk et al., 2010), StructOMP): (a) Recovery Error vs. Sample Size Ratio (n/k); (b) CPU Time vs. Sample Size Ratio (n/k)

Note that Lasso performs better than OMP in the first example. This is because the signal is strongly sparse (that is, all nonzero coefficients are significantly different from zero). In the second experiment, we randomly generate a $1D$ structured sparse signal with weak sparsity, where the nonzero coefficients decay gradually to zero, but there is no clear cutoff. One instance of generated signal is shown in Figure 10 (a). Here, $p = 512$ and all coefficient of the signal are not zeros. We define the sparsity k as the number of coefficients that contain 95% of the image energy. The support set of these signals is composed of $g = 2$ connected regions. Again, each element of the sparse coefficient is connected to two of its adjacent elements, which forms the underlying 1D line graph structure. The graph sparsity concept introduced earlier is used to compute the coding length of sparsity patterns in StructOMP. The projection matrix X is generated by creating an $n \times p$ matrix with i.i.d. draws from a standard Gaussian distribution $N(0,1)$. For simplicity, the rows of X are normalized to unit magnitude. Zero-mean Gaussian noise with standard deviation $\sigma = 0.01$ is added to the measurements.

Figure 10 shows one generated signal and its recovered results by different algorithms when $k = 32$ and $n = 48$. Again, we observe that OMP and Lasso do not achieve good recovery results, whereas the StructOMP algorithm achieves near perfect recovery of the original signal. As we do not know the predefined groups for group Lasso, we just try group Lasso with several different

group sizes (gs=2, 4, 8, 16). Although the results obtained with group Lasso are better than those of OMP and Lasso, they are still inferior to the results with StructOMP. In order to study how the sample size n effects the recovery performance, we vary the sample size and record the recovery results by different algorithms. To reduce the randomness, we perform the experiment 100 times for each of the sample sizes.

Figure 11(a) shows the recovery performance in terms of Recovery Error and Sample Size, averaged over 100 random runs for each sample size. As expected, StructOMP algorithm is superior in all cases. What's different from the first experiment is that the recovery error of OMP becomes smaller than that of Lasso. This result is consistent with our theory, which predicts that if the underlying signal is weakly sparse, then the relatively performance of OMP becomes comparable to Lasso. Figure 11(b) shows the recovery performance in terms of CPU Time and Sample Size, averaged over 100 random runs for each sample size. The computational complexities of StructOMP and OMP are far lower than those of Lasso and Group Lasso.

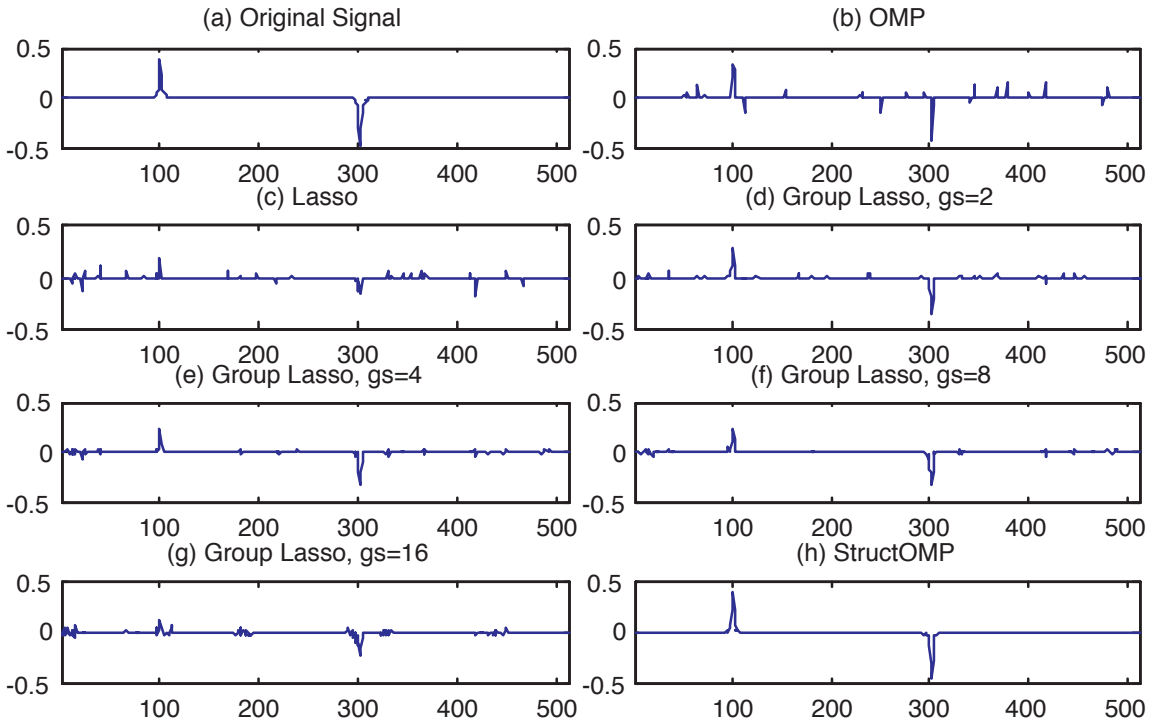
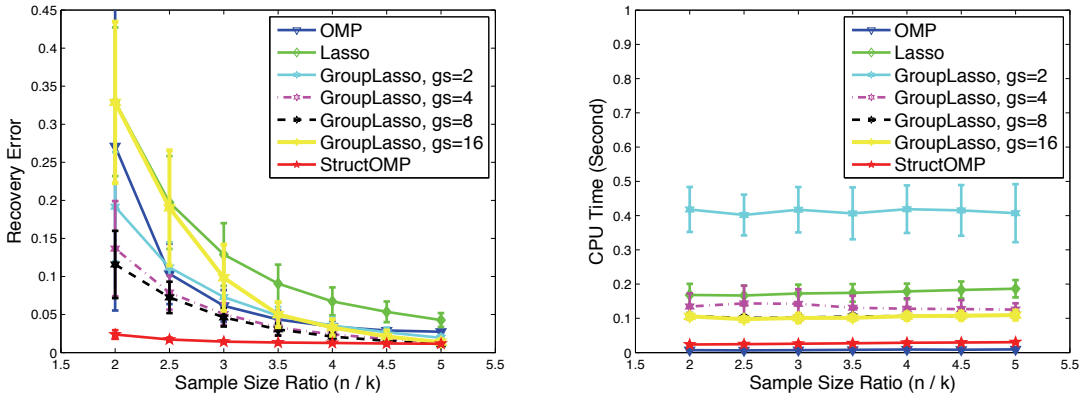


Figure 10: Recovery results of 1D weakly sparse signal with line-structured sparsity. (a) original data; (b) recovered results with OMP (error is 0.5599); (c) recovered results with Lasso (error is 0.6686); (d) recovered results with Group Lasso (error is 0.4732 with group size gs=2); (e) recovered results with Group Lasso (error is 0.2893 with group size gs=4);(f) recovered results with Group Lasso (error is 0.2646 with group size gs=8);(g) recovered results with Group Lasso (error is 0.5459 with group size gs=16); (h) recovered results with StructOMP (error is 0.0846).



(a) (b)

Figure 11: Recovery performance for 1D Weak Line-Sparsity: (a) Recovery Error vs. Sample Size Ratio (n/k); (b) CPU Time vs. Sample Size Ratio (n/k)

5.2 2D Image Compressive Sensing with Tree-structured Sparsity

It is well known that 2D natural images are sparse in a wavelet basis. Their wavelet coefficients have a hierarchical tree structure, which is widely used for wavelet-based compression algorithms (Shapiro, 1993). Figure 12(a) shows a widely used example image with size 64×64: *cameraman*. Note that we use a reduced image instead of the original for computational efficiency since the experiments is run many times with different random matrices. This reduction should not affect the relative performance among various algorithms.

In this experiment, each 2D wavelet coefficient of this image is connected to its parent coefficient and child coefficients, which forms the underlying hierarchical tree structure (which is a special case of graph sparsity). In our experiment, we choose Haar-wavelet to obtain its tree-structured sparsity wavelet coefficients. The projection matrix X and noises are generated with the same method as that for 1D structured sparsity signals. OMP, Lasso and StructOMP are used to recover the wavelet coefficients from the random projection samples respectively. Then, the inverse wavelet transform is used to reconstruct the images with these recovered wavelet coefficients. Our task is to compare the recovery performance of the StructOMP to those of OMP and Lasso under the framework of compressive sensing.

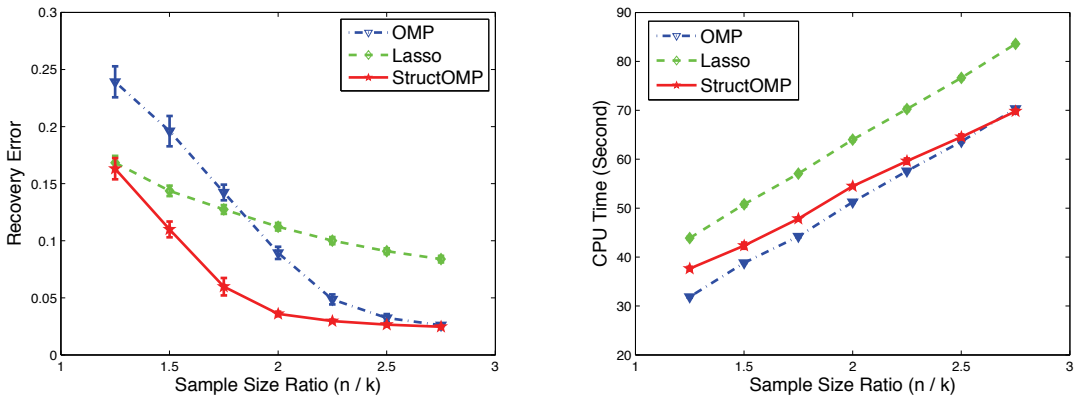
For Lasso, we use identical regularization parameter for all coefficients (without varying regularization parameters based on bands or tree depth). For StructOMP, a simple block-structure is used, where each block corresponds to a node in the tree, plus its ancestors leading to the root. This corresponds to setting $\delta = 0$ in Proposition 11. We use this block set for efficiency only because the number of blocks is only linear in p.

Figure 12 shows one example of the recovered results by different algorithms with sparsity number $k = 1133$ and sample size $n = 2048$. It shows that StructOMP obtains the best recovered result. Figure 13(a) shows the recovery performance in terms of Sample Size and Recovery Error, averaged over 100 random runs for each sample size. The StructOMP algorithm is better than both Lasso and OMP in this case. Since real image data are weakly sparse, the performance of standard

OMP (without structured sparsity) is similar to that of Lasso. Figure 13(b) shows the recovery performance in terms of Sample Size and CPU Time, averaged over 100 random runs for each sample size. The computational complexity of StructOMP is comparable to that of OMP and lower than that of Lasso.

(a) (b) (c) (d)

Figure 12: Recovery results with sample size $n = 2048$: (a) cameraman image, (b) recovered image with OMP (error is 0.1886; CPU time is 46.16s), (c) recovered image with Lasso (error is 0.1670; CPU time is 60.26s) and (d) recovered image with StructOMP (error is 0.0375; CPU time is 48.99s)



(a) (b)

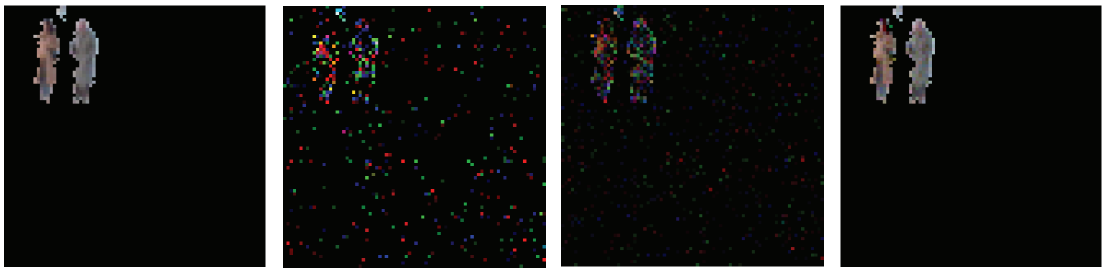
Figure 13: Recovery performance for 2D wavelet tree sparsity: (a) Recovery Error vs. Sample Size; (b) CPU Time vs. Sample size

5.3 Background Subtracted Images for Robust Surveillance

Background subtracted images are typical structure sparsity data in static video surveillance applications. They generally correspond to the foreground objects of interest. Unlike the whole scene, these images are not only spatially sparse but also inclined to cluster into groups, which correspond to different foreground objects. Thus, the StructOMP algorithm can obtain superior recovery

from compressive sensing measurements that are received by a centralized server from multiple and randomly placed optical sensors. In this experiment, the testing video is downloaded from `http://homepages.inf.ed.ac.uk/rbf/CAVIARDATA1/`. The background subtracted images are obtained with the software (Zivkovic and Heijden, 2006). One sample image frame is shown in Figure 14(a). The support set of 2D images is thus composed of several connected regions. Here, each pixel of the 2D background subtracted image is connected to four of its adjacent pixels, forming the underlying graph structure in graph sparsity. We randomly choose 100 background subtracted images as test images.

Note that color images have three channels. We can consider three channels separately and perform sparse recovery independently for each channel. On the other hand, since in this application, three channels of the color background subtracted image share the same support set, we can enforce group sparsity across the color channels for each pixel. That is, a pixel in the color image can be considered as a triplet with three color intensities. We will thus consider both cases in our comparisons. In the latter case, we simply replace OMP and Lasso by Group OMP (which has also been studied by Lozano et al., 2009) and Group Lasso respectively.

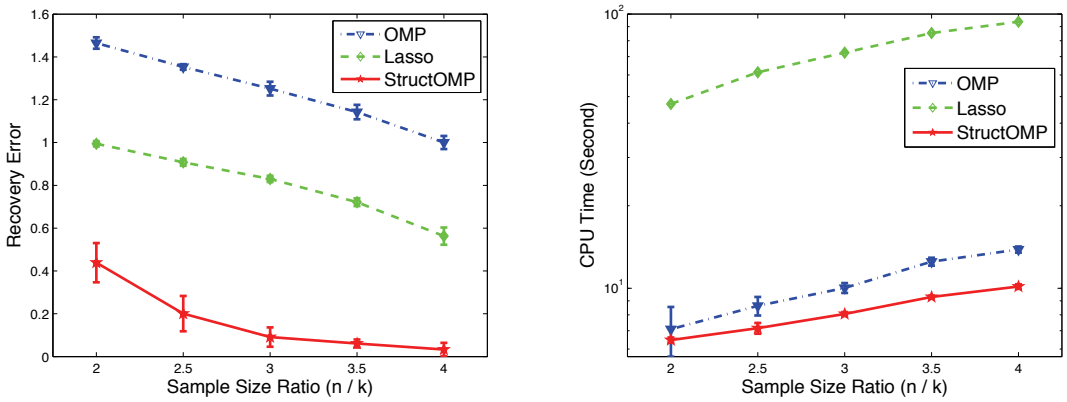


(a) (b) (c) (d)

Figure 14: Recovery results with sample size $n = 900$: (a) the background subtracted image, (b) recovered image with OMP (error is 1.1833), (c) recovered image with Lasso (error is 0.7075) and (d) recovered image with StructOMP (error is 0.1203)

In this experiment, we firstly consider the 3 color channel independently, and use OMP, Lasso and StructOMP to separately recover each channel. The results shown in Figure 14 demonstrates that the StructOMP outperforms both OMP and Lasso in recovery. Figure 15(a) shows the recovery performance as a function of increasing sample size ratios. It demonstrates again that StructOMP significantly outperforms OMP and Lasso in recovery performance on video data. Comparing to the image compression example in the previous section, the background subtracted images have a more clearly defined sparsity pattern where nonzero coefficients are generally distinct from zero (that is, stronger sparsity); this explains why Lasso performs better than the OMP on this particular data. The results is again consistent with our theory. Figure 17(b) shows the recovery performance in terms of Sample Size and CPU Time, averaged over 100 random runs for each sample size. The computational complexity of StructOMP is again comparable to that of OMP and lower than that of Lasso.

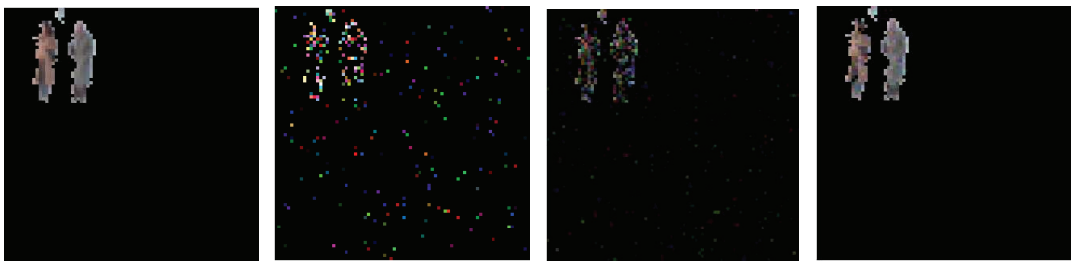
If we consider a pixel as a triplet in the background subtracted image, we replace OMP and Lasso by Group OMP and Group Lasso (across the color channels), and compare their performance to StructOMP. The results in Figure 16 indicate that StructOMP is still superior, although



(a) (b)

Figure 15: Recovery performance: (a) Recovery Error vs. Sample Size; (b) CPU Time vs. Sample size

as expected, the recovery performance of Group OMP (or Group Lasso) improves that of OMP (or Lasso). Figure 17(a) shows the recovery performance as a function of increasing sample size ratios. It demonstrates again that StructOMP outperforms Group OMP and Group Lasso in this application. Figure 17(b) shows the recovery performance in terms of Sample Size and CPU Time, averaged over 100 random runs for each sample size. The computational complexity of StructOMP is again comparable to that of Group OMP and lower than that of Group Lasso.

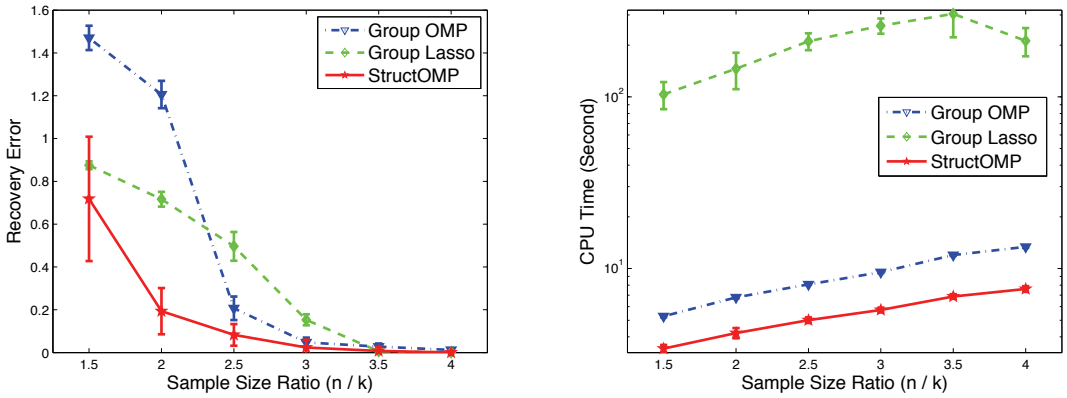


(a) (b) (c) (d)

Figure 16: Recovery results with sample size $n = 600$: (a) the background subtracted image, (b) recovered image with Group OMP (error is 1.1340), (c) recovered image with Group Lasso (error is 0.6972) and (d) recovered image with StructOMP (error is 0.0808)

6. Discussion

This paper develops a theory for structured sparsity where prior knowledge allows us to prefer certain sparsity patterns to others. Some examples are presented to illustrate the concept. The



(a) (b)

Figure 17: Recovery performance: (a) Recovery Error vs. Sample Size; (b) CPU Time vs. Sample size

general framework established in this paper includes the recently popularized group sparsity idea as a special case.

In structured sparsity, the complexity of learning is measured by the coding complexity $c(\beta) \leq \|\beta\|_0 + \mathrm{cl}(\mathrm{supp}(\beta))$ instead of $\|\beta\|_0 \ln p$ which determines the complexity in standard sparsity. Using this notation, a theory parallel to that of the standard sparsity is developed. The theory shows that if the coding length $\mathrm{cl}(\mathrm{supp}(\beta))$ is small for a target coefficient vector β, then the complexity of learning β can be significantly smaller than the corresponding complexity in standard sparsity. Experimental results demonstrate that significant improvements can be obtained on some real problems that have natural structures.

The structured greedy algorithm presented in this paper is the first efficient algorithm proposed to handle the general structured sparsity learning. It is shown that the algorithm is effective under appropriate conditions. Future work include additional computationally efficient methods such as convex relaxation methods (e.g. L_1 regularization for standard sparsity, and group Lasso for strong group sparsity) and backward greedy strategies to improve the forward greedy method considered in this paper.

Appendix A. Proof of Proposition 10 and Proposition 11

Proof of Proposition 10.

First we show that we can encode all connected regions F (that is, with $g = 1$) using no more than

$$C_G|F| - \max_{v \in F} \log_2 q(v) \tag{8}$$

bits. We consider the following procedure to encode F: first, we pick a node v_* from F achieving $-\max_{v \in F} \log_2 q(v)$, which requires $-\max_{v \in F} \log_2 q(v)$ bits. We then push v_* into a stack S. We encode the remaining nodes in F using the following algorithm: until the stack S is empty, we take the top element v out of the stack S, and do the following

(a) Encode the number of neighbors of v in F that has not been visited so far, with no more than $\log_2(1+d_G)$ bits.

(b) For each neighbor v' of v in F that has not been visited, we encode it (i.e., the associated edge between v and v') with no more than $\log_2 d_G$ bits. We then push v' into the stack S.

Since F is connected, after this procedure finishes (the stack becomes empty), we have visited all nodes in F. Since step (a) can be invoked only $|F|$ times, the total number of bits in step (a) is no more than $|F|\log_2(1+d_G)$. The number of bits in step (b) is no more than the number of nodes in F (except for node v_*) times the bits to encode each node, which is no more than $(|F|-1)\log_2(1+d_G)$. Therefore the total number of bits in step (a) and (b) is less than $C_G|F|$. This proves (8).

For $g > 1$, we may encode each connected component F_j of F sequentially, using number of bits according to (8). Then after encoding each connected region F_j, we use 1 bit to encode whether $j = g$ or not (that is, whether we should stop or encode an additional connected component). This gives the formula in Proposition 10.

Proof of Proposition 11.

We first prove the following two lemmas.

Lemma 12 *Given a positive even integer L. Let F be a connected region of G such that $|F| \geq L+1$. Then it is possible to partition F as the union of two connected regions F_1 and F_2 such that: $F = F_1 \cup F_2$, $|F_1 \cap F_2| = 1$, and*

- *either* $\min(|F_1|, |F_2|) \geq 0.5L+1$;
- *or* $0.5L+1 \leq |F_1| \leq L$.

Proof We consider the following algorithm. Start with a node v of F and set $F_1 = \{v\}$ and let $u_1 = v$. Repeat the following procedure

(a) If $|F_1| \geq 0.5L+1$, then exit the procedure with the current F_1 and $F_2 = (F - F_1) \cup \{u_1\}$.

(b) If $F - F_1$ is connected: let v be a node in $F - F_1$ that is connected to F_1. We add v to F_1, and set $u_1 = v$. We then repeat the procedure (a)(b)(c).

(c) If $F - F_1$ is not connected: $(F - F_1) \cup \{u_1\}$ is connected by construction. Merge the smallest connected component of $F - F_1$ into F_1. Repeat the procedure (a)(b)(c).

Clearly the procedure eventually will end at step (a) because each iteration $|F_1|$ is increased by at least 1. When it ends, $F_1 \cap F_2 = \{u_1\}$. Moreover, there were two possible scenarios in the previous iteration:

(1) Step (b) was invoked. That is, $|F_1|$ was increased by 1 in the previous iteration, and hence $|F_1| = 0.5L+1 \leq L$. Moreover, F_2 is connected.

(2) Step (c) was invoked. Still, F_2 is connected by the construction of u_1. If $|F_1|$ was increased by no more than $L/2$ in the previous step (c), then $|F_1| \leq L$ and the lemma holds. Otherwise, $F - F_1$ has more than $L/2$ nodes because this scenario implies that even the smallest connected component has more than $L/2$ nodes in the previous step (c). Therefore in this case we have $|F_2| > 0.5L+1$.

■

Lemma 13 *Given a positive even integer L. Any connected region F such that $|F| \geq 0.5L+1$ can be covered by at most $2(|F|-1)/L$ connected regions, each of size no more than L.*

Proof We fix L and prove the claim by induction on $|F|$. If $0.5L+1 \leq |F| \leq L$, then F is covered by itself, and the claim is trivial. If $L < |F| \leq 1.5L$, then by Lemma 12, we can partition F into two connected regions, each $\leq L$. Therefore the claim also holds trivially.

Now assume that the claim holds for $|F| \leq k$ with $k \geq 1.5L$. For F such that $|F| = k+1$, we apply Lemma 12 and partition it into two regions $F = F_1 \cup F_2$ such that $\min(|F_1|, |F_2|) \geq 0.5L+1$ and $|F_1| + |F_2| = |F| + 1$. Therefore by the induction hypothesis, we can cover each F_j ($j = 1,2$) by $2(|F_j|-1)/L$ connected regions, each of size no more than L. It follows that the total number of connected regions to cover both F_1 and F_2 is no more than $2(|F_1| + |F_2| - 2)/L = 2(|F|-1)/L$, which completes the induction. ■

We are now ready to prove Proposition 11. First, from (8), we know that $C_G|B| + \log_2 p$ is a coding-length for connected regions $B \in \mathcal{B}$. Therefore

$$2^{-(C_G L + \log_2 p)}|\mathcal{B}| \leq \sum_{B \in \mathcal{B}} 2^{-(C_G|B| + \log_2 p)} \leq 1.$$

This implies that $|\mathcal{B}| \leq p^{1+C_G\delta}$.

Since Lemma 13 implies that each connected component F_j of F can be covered by $1+2(|F_j|-1)/L$ connected regions from $\mathcal{B}$, we have $\mathrm{cl}_{\mathcal{B}}(F_j) \leq (1+2(|F_j|-1)/L)(1+C_G\delta)\log_2 p$ under the uniform coding on $\mathcal{B}$. By summing over the connected components, we obtain the desired bound.

Appendix B. Proof of Proposition 3

Lemma 14 *Consider a fixed vector $\mathbf{x} \in \mathbb{R}^n$, and a random vector $\mathbf{y} \in \mathbb{R}^n$ with independent sub-Gaussian components: $\mathbb{E}e^{t(\mathbf{y}_i - \mathbb{E}\mathbf{y}_i)} \leq e^{\sigma^2 t^2/2}$ for all t and i, then $\forall \varepsilon > 0$:*

$$\Pr\left(\left|\mathbf{x}^\top \mathbf{y} - \mathbb{E}\mathbf{x}^\top \mathbf{y}\right| \geq \varepsilon\right) \leq 2e^{-\varepsilon^2/(2\sigma^2\|\mathbf{x}\|_2^2)}.$$

Proof Let $s_n = \sum_{i=1}^n (\mathbf{x}_i\mathbf{y}_i - \mathbb{E}\mathbf{x}_i\mathbf{y}_i)$; then by assumption, $\mathbb{E}(e^{ts_n} + e^{-ts_n}) \leq 2e^{\sum_i \mathbf{x}_i^2\sigma^2 t^2/2}$, which implies that $\Pr(|s_n| \geq \varepsilon)e^{t\varepsilon} \leq 2e^{\sum_i \mathbf{x}_i^2\sigma^2 t^2/2}$. Now let $t = \varepsilon/(\sum_i \mathbf{x}_i^2\sigma^2)$, we obtain the desired bound. ■

The following lemma is taken from Pisier (1989).

Lemma 15 *Consider the unit sphere $S^{k-1} = \{x : \|x\|_2 = 1\}$ in $\mathbb{R}^k$ ($k \geq 1$). Given any $\varepsilon > 0$, there exists an ε-cover $Q \subset S^{k-1}$ such that $\min_{q \in Q} \|x - q\|_2 \leq \varepsilon$ for all $\|x\|_2 = 1$, with $|Q| \leq (1+2/\varepsilon)^k$.*

B.1 Proof of Proposition 3

According to Lemma 15, given $\varepsilon_1 > 0$, there exists a finite set $Q = \{q_i\}$ with $|Q| \leq (1+2/\varepsilon_1)^k$ such that $\|Pq_i\|_2 = 1$ for all i, and $\min_i \|P\mathbf{z} - Pq_i\|_2 \leq \varepsilon_1$ for all $\|P\mathbf{z}\|_2 = 1$. To see the existence of Q,

we consider a rotation of the coordinate system (which does not change 2-norm) so that $P\mathbf{z}$ is the projection of $\mathbf{z} \in \mathbb{R}^n$ to its first k coordinates in the new coordinate system. Lemma 15 can now be directly applied to the first k coordinates in the new system, implying that we can pick q_i such that $Pq_i = q_i$.

For each i, Lemma 14 implies that $\forall \varepsilon_2 > 0$:

$$\Pr\left(\left|q_i^\top P(\mathbf{y} - \mathbb{E}\mathbf{y})\right| \geq \varepsilon_2\right) \leq 2e^{-\varepsilon_2^2/(2\sigma^2)}.$$

Taking union bound for all $q_i \in Q$, we obtain with probability exceeding $1 - 2(1+2/\varepsilon_1)^k e^{-\varepsilon_2^2/2\sigma^2}$:

$$\left|q_i^\top P(\mathbf{y} - \mathbb{E}\mathbf{y})\right| \leq \varepsilon_2$$

for all i.

Let $\mathbf{z} = P(\mathbf{y} - \mathbb{E}\mathbf{y})/\|P(\mathbf{y} - \mathbb{E}\mathbf{y})\|_2$, then there exists i such that $\|P\mathbf{z} - Pq_i\|_2 \leq \varepsilon_1$. We have

$$\begin{aligned}\|P(\mathbf{y} - \mathbb{E}\mathbf{y})\|_2 =& \mathbf{z}^\top P(\mathbf{y} - \mathbb{E}\mathbf{y}) \\ \leq& \|P\mathbf{z} - Pq_i\|_2 \|P(\mathbf{y} - \mathbb{E}\mathbf{y})\|_2 + |q_i^\top P(\mathbf{y} - \mathbb{E}\mathbf{y})| \\ \leq& \varepsilon_1 \|P(\mathbf{y} - \mathbb{E}\mathbf{y})\|_2 + \varepsilon_2.\end{aligned}$$

Therefore

$$\|P(\mathbf{y} - \mathbb{E}\mathbf{y})\|_2 \leq \varepsilon_2/(1 - \varepsilon_1).$$

Let $\varepsilon_1 = 2/15$, and $\eta = 2(1+2/\varepsilon_1)^k e^{-\varepsilon_2^2/2\sigma^2}$, we have

$$\varepsilon_2^2 = 2\sigma^2[(4k+1)\ln 2 - \ln \eta],$$

and thus

$$\|P(\mathbf{y} - \mathbb{E}\mathbf{y})\|_2 \leq \frac{15}{13}\sigma\sqrt{2(4k+1)\ln 2 - 2\ln\eta}.$$

This simplifies to the desired bound.

Appendix C. Proof of Proposition 5

We use the following lemma from Huang and Zhang (2010).

Lemma 16 *Suppose X is generated according to Proposition 5. For any fixed set $F \subset I$ with $|F| = k$ and $0 < \delta < 1$, we have with probability exceeding $1 - 3(1 + 8/\delta)^k e^{-n\delta^2/8}$:*

$$(1 - \delta)\|\beta\|_2 \leq \frac{1}{\sqrt{n}}\|X_F\beta\|_2 \leq (1 + \delta)\|\beta\|_2$$

for all $\beta \in \mathbb{R}^k$.

C.1 Proof of Proposition 5

Since $\mathrm{cl}(F)$ is a coding length, we have (for any fixed $\gamma < 1$)

$$\sum_{F:|F|+\mathrm{cl}(F)\leq s}(1+8/\delta)^{|F|} \leq \sum_{F:|F|+\gamma\mathrm{cl}(F)\leq s}(1+8/\delta)^{|F|}$$
$$\leq \sum_F (1+8/\delta)^{s-\gamma\mathrm{cl}(F)} = (1+8/\delta)^s \sum_F 2^{-\mathrm{cl}(F)} \leq (1+8/\delta)^s,$$

where in the above derivation, we take $\gamma = 1/\log_2(1+8/\delta)$.

For each F, we know from Lemma 16 that for all β such that $\mathrm{supp}(\beta) \subset F$:

$$(1-\delta)\|\beta\|_2 \leq \frac{1}{\sqrt{n}}\|X\beta\|_2 \leq (1+\delta)\|\beta\|_2$$

with probability exceeding $1-3(1+8/\delta)^{|F|}e^{-n\delta^2/8}$.

We can thus take the union bound over $F : |F| + \mathrm{cl}(F) \leq s$, which shows that with probability exceeding

$$1 - \sum_{F:|F|+\mathrm{cl}(F)\leq s} 3(1+8/\delta)^{|F|}e^{-n\delta^2/8},$$

the structured RIP in Equation (4) holds. Since

$$\sum_{F:|F|+\mathrm{cl}(F)\leq s} 3(1+8/\delta)^{|F|}e^{-n\delta^2/8} \leq 3(1+8/\delta)^s e^{-n\delta^2/8} \leq e^{-t},$$

we obtain the desired bound.

Appendix D. Proof of Theorem 6 and Theorem 7

Lemma 17 *Suppose that Assumption 1 is valid. For any fixed subset $F \subset I$, with probability $1-\eta$, $\forall\beta$ such that $\mathrm{supp}(\beta) \subset F$, and $a > 0$, we have*

$$\|X\beta - \mathbb{E}\mathbf{y}\|_2^2 \leq (1+a)[\|X\beta-\mathbf{y}\|_2^2 - \|\mathbf{y}-\mathbb{E}\mathbf{y}\|_2^2] + (2+a+a^{-1})\sigma^2[7.4|F| + 4.7\ln(4/\eta)].$$

Proof Let

$$P_F = X_F(X_F^\top X_F)^+ X_F^\top$$

be the projection matrix to the subspace generated by columns of X_F. Here X_F may not be full-rank, and $(X_F^\top X_F)^+$ denotes the Moore-Penrose pseudo-inverse. Since $X\beta$ belongs to this subspace, we have $P_F X\beta = X\beta$.

Let $\mathbf{z} = (I - P_F)\mathbb{E}\mathbf{y}/\|(I-P_F)\mathbb{E}\mathbf{y}\|_2$, $\delta_1 = \|P_F(\mathbf{y}-\mathbb{E}\mathbf{y})\|_2$ and $\delta_2 = |\mathbf{z}^\top(\mathbf{y}-\mathbb{E}\mathbf{y})|$, we have

$$\begin{aligned}
&\|X\beta - \mathbb{E}\mathbf{y}\|_2^2 \\
=&\|X\beta-\mathbf{y}\|_2^2 - \|\mathbf{y}-\mathbb{E}\mathbf{y}\|_2^2 + 2(\mathbf{y}-\mathbb{E}\mathbf{y})^\top(X\beta-\mathbb{E}\mathbf{y}) \\
=&\|X\beta-\mathbf{y}\|_2^2 - \|\mathbf{y}-\mathbb{E}\mathbf{y}\|_2^2 + 2(\mathbf{y}-\mathbb{E}\mathbf{y})^\top(X\beta-P_F\mathbb{E}\mathbf{y}) - 2\mathbf{z}^\top(\mathbf{y}-\mathbb{E}\mathbf{y})\|(I-P_F)\mathbb{E}\mathbf{y}\|_2 \\
=&\|X\beta-\mathbf{y}\|_2^2 - \|\mathbf{y}-\mathbb{E}\mathbf{y}\|_2^2 + 2(\mathbf{y}-\mathbb{E}\mathbf{y})^\top P_F(X\beta-P_F\mathbb{E}\mathbf{y}) - 2\mathbf{z}^\top(\mathbf{y}-\mathbb{E}\mathbf{y})\|(I-P_F)\mathbb{E}\mathbf{y}\|_2 \\
\le&\|X\beta-\mathbf{y}\|_2^2 - \|\mathbf{y}-\mathbb{E}\mathbf{y}\|_2^2 + 2\delta_1\|X\beta-P_F\mathbb{E}\mathbf{y}\|_2 + 2\delta_2\|(I-P_F)\mathbb{E}\mathbf{y}\|_2 \\
\le&\|X\beta-\mathbf{y}\|_2^2 - \|\mathbf{y}-\mathbb{E}\mathbf{y}\|_2^2 + 2\sqrt{\delta_1^2+\delta_2^2}\sqrt{\|X\beta-P_F\mathbb{E}\mathbf{y}\|_2^2 + \|(I-P_F)\mathbb{E}\mathbf{y}\|_2^2} \\
=&\|X\beta-\mathbf{y}\|_2^2 - \|\mathbf{y}-\mathbb{E}\mathbf{y}\|_2^2 + 2\sqrt{\delta_1^2+\delta_2^2}\|X\beta-\mathbb{E}\mathbf{y}\|_2.
\end{aligned}$$

Note that in the above derivation, the first two equalities are simple algebra. The third equality uses the fact that $P_F X\beta = X\beta$. The first inequality uses the Cauchy-Schwartz inequality and the definitions of δ_1 and δ_2. The second inequality uses the Cauchy-Schwartz inequality of the form $\delta_1 a_1 + \delta_2 a_2 \le \sqrt{\delta_1^2+\delta_2^2}\sqrt{a_1^2+a_2^2}$. The last equality uses the fact that $\|X\beta - P_F\mathbb{E}\mathbf{y}\|_2^2 + \|(I-P_F)\mathbb{E}\mathbf{y}\|_2^2 = \|X\beta-\mathbb{E}\mathbf{y}\|_2^2$, which is a consequence of the fact that P_F is a projection matrix and $P_F X\beta = X\beta$.

Now, by solving the above displayed inequality with respect to $\|X\beta-\mathbb{E}\mathbf{y}\|_2$, we obtain

$$\begin{aligned}
\|X\beta-\mathbb{E}\mathbf{y}\|_2^2 \le& \left[\sqrt{\|X\beta-\mathbf{y}\|_2^2 - \|\mathbf{y}-\mathbb{E}\mathbf{y}\|_2^2 + \delta_1^2+\delta_2^2} + \sqrt{\delta_1^2+\delta_2^2}\right]^2 \\
\le&(1+a)[\|X\beta-\mathbf{y}\|_2^2 - \|\mathbf{y}-\mathbb{E}\mathbf{y}\|_2^2] + (2+a+1/a)(\delta_1^2+\delta_2^2).
\end{aligned}$$

The desired bound now follows easily from Proposition 3 and Lemma 14, where we know that with probability $1-\eta/2$,

$$\delta_1^2 = (\mathbf{y}-\mathbb{E}\mathbf{y})^\top P_F(\mathbf{y}-\mathbb{E}\mathbf{y}) \le \sigma^2(7.4|F| + 2.7\ln(4/\eta)),$$

and with probability $1-\eta/2$,

$$\delta_2^2 = |\mathbf{z}^\top(\mathbf{y}-\mathbb{E}\mathbf{y})|^2 \le 2\sigma^2\ln(4/\eta).$$

We obtain the desired result by substituting the above two estimates and simplify. ■

Lemma 18 *Suppose that Assumption 1 is valid. Then we have with probability* $1-\eta$, $\forall\beta\in\mathbb{R}^p$ *and* $a>0$*:*

$$\|X\beta-\mathbb{E}\mathbf{y}\|_2^2 \le (1+a)\left[\|X\beta-\mathbf{y}\|_2^2 - \|\mathbf{y}-\mathbb{E}\mathbf{y}\|_2^2\right] + (2+a+1/a)\sigma^2[7.4c(\beta) + 4.7\ln(4/\eta)].$$

Proof Note that for each F, with probability $2^{-\mathrm{cl}(F)}\eta$, we obtain from Lemma 17 that $\forall\mathrm{supp}(\beta)\in F$,

$$\|X\beta-\mathbb{E}\mathbf{y}\|_2^2 \le (1+a)\left[\|X\beta-\mathbf{y}\|_2^2 - \|\mathbf{y}-\mathbb{E}\mathbf{y}\|_2^2\right] + (2+a+1/a)\sigma^2[7.4(|F|+\mathrm{cl}(F)) + 4.7\ln(4/\eta)].$$

Since $\sum_{F\subset I, F\neq\emptyset} 2^{-\mathrm{cl}(F)}\eta \le \eta$, the result follows from the union bound. ■

Lemma 19 *Consider a fixed subset $F \subset I$. Given any $\eta \in (0,1)$, we have with probability $1-\eta$:*

$$|\|X\beta - \mathbf{y}\|_2^2 - \|\mathbf{y} - \mathbb{E}\mathbf{y}\|_2^2| \leq \|X\beta - \mathbb{E}\mathbf{y}\|_2^2 + 2\sigma\sqrt{2\ln(2/\eta)}\|X\beta - \mathbb{E}\mathbf{y}\|_2.$$

Proof Let $\tilde{a} = (X\beta - \mathbb{E}\mathbf{y})/\|X\beta - \mathbb{E}\mathbf{y}\|_2$, we have

$$\begin{aligned}
&|\|X\beta - \mathbf{y}\|_2^2 - \|\mathbf{y} - \mathbb{E}\mathbf{y}\|_2^2| \\
=&|-2(X\beta - \mathbb{E}\mathbf{y})^\top(\mathbf{y} - \mathbb{E}\mathbf{y}) + \|X\beta - \mathbb{E}\mathbf{y}\|_2^2| \\
\leq& 2\|X\beta - \mathbb{E}\mathbf{y}\|_2|\tilde{a}^\top(\mathbf{y} - \mathbb{E}\mathbf{y})| + \|\mathbb{E}\mathbf{y} - X\beta\|_2^2.
\end{aligned}$$

The desired result now follows from Lemma 14. ■

Lemma 20 *Suppose that Assumption 1 is valid. Consider any fixed target $\beta \in \mathbb{R}^p$. Then with probability exceeding $1-\eta$, for all $\lambda \geq 0, \varepsilon \geq 0, \hat{\beta} \in \mathbb{R}^p$ such that: $\hat{Q}(\hat{\beta}) + \lambda c(\hat{\beta}) \leq \hat{Q}(\beta) + \lambda c(\beta) + \varepsilon$, and for all $a > 0$, we have*

$$\begin{aligned}
\|X\hat{\beta} - \mathbb{E}\mathbf{y}\|_2^2 \leq& (1+a)[\|X\beta - \mathbb{E}\mathbf{y}\|_2^2 + 2\sigma\sqrt{2\ln(6/\eta)}\|X\beta - \mathbb{E}\mathbf{y}\|_2] \\
&+ (1+a)\lambda c(\beta) + a' c(\hat{\beta}) + b'\ln(6/\eta) + (1+a)\varepsilon,
\end{aligned}$$

where $a' = 7.4(2 + a + a^{-1})\sigma^2 - (1+a)\lambda$ and $b' = 4.7\sigma^2(2 + a + a^{-1})$. Moreover, if the coding scheme $c(\cdot)$ is sub-additive, then

$$n\rho_-(c(\hat{\beta}) + c(\beta))\|\hat{\beta} - \beta\|_2^2 \leq 10\|X\beta - \mathbb{E}\mathbf{y}\|_2^2 + 2.5\lambda c(\beta) + (37\sigma^2 - 2.5\lambda)c(\hat{\beta}) + 29\sigma^2\ln(6/\eta) + 2.5\varepsilon.$$

Proof We obtain from the union bound of Lemma 18 (with probability $1 - \eta/3$) and Lemma 19 (with probability $1 - 2\eta/3$) that with probability $1 - \eta$:

$$\begin{aligned}
&\|X\hat{\beta} - \mathbb{E}\mathbf{y}\|_2^2 \\
\leq& (1+a)\left[\|X\hat{\beta} - \mathbf{y}\|_2^2 - \|\mathbf{y} - \mathbb{E}\mathbf{y}\|_2^2\right] + (2 + a + a^{-1})[7.4\sigma^2 c(\hat{\beta}) + 4.7\sigma^2\ln(6/\eta)] \\
\leq& (1+a)\left[\|X\beta - \mathbf{y}\|_2^2 - \|\mathbf{y} - \mathbb{E}\mathbf{y}\|_2^2 + \lambda c(\beta) + \varepsilon\right] + a' c(\hat{\beta}) + b'\ln(6/\eta) \\
\leq& (1+a)[\|X\beta - \mathbb{E}\mathbf{y}\|_2^2 + 2\sigma\sqrt{2\ln(6/\eta)}\|X\beta - \mathbb{E}\mathbf{y}\|_2] + (1+a)\lambda c(\beta) + a' c(\hat{\beta}) \\
&+ b'\ln(6/\eta) + (1+a)\varepsilon.
\end{aligned}$$

This proves the first claim of the lemma.

The first claim with $a = 1$ implies that

$$\begin{aligned}
&\|X\hat{\beta} - X\beta\|_2^2 \leq [\|X\hat{\beta} - \mathbb{E}\mathbf{y}\|_2 + \|X\beta - \mathbb{E}\mathbf{y}\|_2]^2 \\
\leq& 1.25\|X\hat{\beta} - \mathbb{E}\mathbf{y}\|_2^2 + 5\|X\beta - \mathbb{E}\mathbf{y}\|_2^2 \\
\leq& 7.5\|X\beta - \mathbb{E}\mathbf{y}\|_2^2 + 5\sigma\sqrt{2\ln(6/\eta)}\|X\beta - \mathbb{E}\mathbf{y}\|_2 + 2.5\lambda c(\beta) + 1.25(29.6\sigma^2 - 2\lambda)c(\hat{\beta}) \\
&+ 1.25 \times 18.8\sigma^2\ln(6/\eta) + 2.5\varepsilon \\
\leq& 10\|X\beta - \mathbb{E}\mathbf{y}\|_2^2 + 2.5\lambda c(\beta) + (37\sigma^2 - 2.5\lambda)c(\hat{\beta}) + 29\sigma^2\ln(6/\eta) + 2.5\varepsilon.
\end{aligned}$$

Since $c(\hat{\beta} - \beta) \leq c(\hat{\beta}) + c(\beta)$, we have $\|X\hat{\beta} - X\beta\|_2^2 \geq n\rho_-(c(\hat{\beta}) + c(\beta))\|\hat{\beta} - \beta\|_2^2$. This implies the second claim. ■

D.1 Proof of Theorem 6

We take $\lambda = 0$ in Lemma 20, and obtain:

$$\begin{aligned}\|X\hat{\beta} - \mathbb{E}\mathbf{y}\|_2^2 \leq & (1+a)[\|X\beta - \mathbb{E}\mathbf{y}\|_2^2 + 2\sigma\sqrt{2\ln(6/\eta)}\|X\beta - \mathbb{E}\mathbf{y}\|_2] \\ & + 7.4(2+a+a^{-1})\sigma^2 c(\hat{\beta}) + 4.7\sigma^2(2+a+a^{-1})\ln(6/\eta) + (1+a)\varepsilon \\ = & (\|X\beta - \mathbb{E}\mathbf{y}\|_2 + \sigma\sqrt{2\ln(6/\eta)})^2 + 14.8\sigma^2 c(\hat{\beta}) + 7.4\sigma^2\ln(6/\eta) + \varepsilon \\ & + a[(\|X\beta - \mathbb{E}\mathbf{y}\|_2 + \sigma\sqrt{2\ln(6/\eta)})^2 + 7.4\sigma^2 c(\hat{\beta}) + 2.7\sigma^2\ln(6/\eta) + \varepsilon] \\ & + a^{-1}[7.4\sigma^2 c(\hat{\beta}) + 4.7\sigma^2\ln(6/\eta)].\end{aligned}$$

Now let $z = \|X\beta - \mathbb{E}\mathbf{y}\|_2 + \sigma\sqrt{2\ln(6/\eta)}$, and we choose a to minimize the right hand side as:

$$\begin{aligned}\|X\hat{\beta} - \mathbb{E}\mathbf{y}\|_2^2 \leq & z^2 + 14.8\sigma^2 c(\hat{\beta}) + 7.4\sigma^2\ln(6/\eta) + \varepsilon \\ & + 2[z^2 + 7.4\sigma^2 c(\hat{\beta}) + 2.7\sigma^2\ln(6/\eta) + \varepsilon]^{1/2}[7.4\sigma^2 c(\hat{\beta}) + 4.7\sigma^2\ln(6/\eta)]^{1/2} \\ \leq & [(z^2 + 7.4\sigma^2 c(\hat{\beta}) + 2.7\sigma^2\ln(6/\eta) + \varepsilon)^{1/2} + (7.4\sigma^2 c(\hat{\beta}) + 4.7\sigma^2\ln(6/\eta))^{1/2}]^2 \\ \leq & [z + 2(7.4\sigma^2 c(\hat{\beta}) + 4.7\sigma^2\ln(6/\eta) + \varepsilon)^{1/2}]^2.\end{aligned}$$

This proves the first inequality. The second inequality follows directly from Lemma 20 with $\lambda = 0$.

D.2 Proof of Theorem 7

The desired bound is a direct consequence of Lemma 20, by noticing that

$$2\sigma\sqrt{2\ln(6/\eta)}\|X\beta - \mathbb{E}\mathbf{y}\|_2 \leq a\|X\beta - \mathbb{E}\mathbf{y}\|_2^2 + a^{-1}2\sigma^2\ln(6/\eta),$$

$a' \leq 0$, and

$$b' + a^{-1}2\sigma^2 \leq (10 + 5a + 7a^{-1})\sigma^2.$$

Appendix E. Proof of Theorem 9

The following lemma is an adaptation of a similar result in Zhang (2011) on greedy algorithms for standard sparsity.

Lemma 21 *Suppose the coding scheme is sub-additive. Consider any* β*, and a cover of* β *by* $\mathcal{B}$*:*

$$\mathrm{supp}(\beta) \subset F = \cup_{j=1}^{b} B_j \quad (B_j \in \mathcal{B}).$$

Let $c(\beta, \mathcal{B}) = \sum_{j=1}^{b} c(B_j)$*. Let* $\rho_0 = \max_j \rho_+(B_j)$*. Then consider* F *such that* $\forall j : c(B_j \cup F) \geq c(F)$*, we define*

$$\beta = \arg\min_{\beta' \in \mathbb{R}^p} \|X\beta' - \mathbf{y}\|_2^2 \quad \textit{subject to} \quad \mathrm{supp}(\beta') \subset F.$$

If $\|X\beta - \mathbf{y}\|_2^2 \geq \|X\beta - \mathbf{y}\|_2^2$*, we have*

$$\max_j \phi(B_j) \geq \frac{\rho_-(F \cup F)}{\rho_0 c(\beta, \mathcal{B})}[\|X\beta - \mathbf{y}\|_2^2 - \|X\beta - \mathbf{y}\|_2^2],$$

where as in (5), we define

$$\phi(B) = \frac{\|P_{B-F}(X\beta - \mathbf{y})\|_2^2}{c(B \cup F) - c(F)}.$$

Proof For all $\ell \in F$, $\|X\beta + \alpha X\mathbf{e}_\ell - \mathbf{y}\|_2^2$ achieves the minimum at $\alpha = 0$ (where $\mathbf{e}_\ell$ is the vector of zeros except for the ℓ-th component, which is one). This implies that

$$\mathbf{x}_\ell^\top (X\beta - \mathbf{y}) = 0$$

for all $\ell \in F$. Therefore we have

$$\begin{aligned}
&(X\beta - \mathbf{y})^\top \sum_{\ell \in \bar{F} - F} (\beta_\ell - \beta_\ell)\mathbf{x}_\ell \\
=&(X\beta - \mathbf{y})^\top \sum_{\ell \in \bar{F} \cup F} (\beta_\ell - \beta_\ell)\mathbf{x}_\ell = (X\beta - \mathbf{y})^\top (X\beta - X\beta) \\
=&-\frac{1}{2}\|X(\beta - \beta)\|_2^2 + \frac{1}{2}\|X\beta - \mathbf{y}\|_2^2 - \frac{1}{2}\|X\beta - \mathbf{y}\|_2^2.
\end{aligned}$$

Now, let $B'_j \subset B_j - F$ be disjoint sets such that $\cup_j B'_j = \bar{F} - F$. The above inequality leads to the following derivation $\forall \eta > 0$:

$$\begin{aligned}
&-\sum_j \phi(B_j)(c(B_j \cup F) - c(F)) \\
\leq &\sum_j \left[\left\| X\beta + \eta \sum_{\ell \in B'_j} (\beta_\ell - \beta_\ell)\mathbf{x}_\ell - \mathbf{y} \right\|_2^2 - \|X\beta - \mathbf{y}\|_2^2 \right] \\
\leq &\eta^2 \sum_{\ell \in \bar{F} - F} (\beta_\ell - \beta_\ell)^2 \rho_0 n + 2\eta (X\beta - \mathbf{y})^\top \sum_{\ell \in \bar{F} - F} (\beta_\ell - \beta_\ell)\mathbf{x}_\ell \\
\leq &\eta^2 \sum_{\ell \in \bar{F} - F} (\beta_\ell - \beta_\ell)^2 \rho_0 n - \eta \|X(\beta - \beta)\|_2^2 + \eta \|X\beta - \mathbf{y}\|_2^2 - \eta \|X\beta - \mathbf{y}\|_2^2.
\end{aligned}$$

Note that we have used the fact that $\|P_{B-F}(X\beta - \mathbf{y})\|_2^2 \geq \|X\beta - \mathbf{y}\|_2^2 - \|X\beta - \mathbf{y} + X\Delta\beta\|_2^2$ for all $\Delta\beta$ such that $\mathrm{supp}(\Delta\beta) \subset B - F$. By optimizing over η, we obtain

$$\begin{aligned}
\max_j \phi(B_j) \sum_j c(B_j) \geq &\sum_j \phi(B_j)(c(B_j \cup F) - c(F)) \\
\geq &\frac{[\|X(\beta - \beta)\|_2^2 + \|X\beta - \mathbf{y}\|_2^2 - \|X\beta - \mathbf{y}\|_2^2]^2}{4\sum_{\ell \in \bar{F} - F} (\beta_\ell - \beta_\ell)^2 \rho_0 n} \\
\geq &\frac{4\|X(\beta - \beta)\|_2^2 [\|X\beta - \mathbf{y}\|_2^2 - \|X\beta - \mathbf{y}\|_2^2]}{4\sum_{\ell \in \bar{F} - F} (\beta_\ell - \beta_\ell)^2 \rho_0 n} \\
\geq &\frac{\rho_-(F \cup F)}{\rho_0} [\|X\beta - \mathbf{y}\|_2^2 - \|X\beta - \mathbf{y}\|_2^2].
\end{aligned}$$

This leads to the desired bound. In the above derivation, the first inequality is simple algebra; the second inequality is by optimizing over η mentioned earlier; the third inequality is of the form $[a_1 + a_2]^2 \geq 4a_1 a_2$. The last inequality uses the definition of $\rho_-(\cdot)$. ■

E.1 Proof of Theorem 9

Let

$$\nu' = \frac{\nu\rho_-(s+c(F))}{\rho_0(\mathcal{B})c(\beta,\mathcal{B})}.$$

By Lemma 21, we have at any step $k > 0$:

$$\|X\beta^{(k-1)} - \mathbf{y}\|_2^2 - \|X\beta^{(k)} - \mathbf{y}\|_2^2 \geq \nu'[\|X\beta^{(k-1)} - \mathbf{y}\|_2^2 - \|X\beta - \mathbf{y}\|_2^2](c(\beta^{(k)}) - c(\beta^{(k-1)}),$$

which implies that

$$\max[0, \|X\beta^{(k)} - \mathbf{y}\|_2^2 - \|X\beta - \mathbf{y}\|_2^2] \leq \max[0, \|X\beta^{(k-1)} - \mathbf{y}\|_2^2 - \|X\beta - \mathbf{y}\|_2^2]e^{-\nu'(c(\beta^{(k)})-c(\beta^{(k-1)})}.$$

Therefore at stopping, we have

$$\begin{aligned}
&\|X\beta^{(k)} - \mathbf{y}\|_2^2 - \|X\beta - \mathbf{y}\|_2^2 \\
\leq &[\|\mathbf{y}\|_2^2 - \|X\beta - \mathbf{y}\|_2^2]e^{-\nu' c(\beta^{(k)})} \\
\leq &[\|\mathbf{y}\|_2^2 - \|X\beta - \mathbf{y}\|_2^2]e^{-\nu' s} \leq \varepsilon.
\end{aligned}$$

This proves the theorem.

Appendix F. Performance of StructOMP for Weakly Sparse Signals

Theorem 22 *Suppose the coding scheme is sub-additive. Given a sequence of targets β_j such that $\hat{Q}(\beta_0) \leq \hat{Q}(\beta_1) \leq \cdots$ and $c(\beta_j,\mathcal{B}) \leq c(\beta_0,\mathcal{B})/2^j$. If*

$$s \geq \frac{\rho_0(\mathcal{B})}{\nu \min_j \rho_-(s+c(\beta_j))} c(\beta_0,\mathcal{B}) \left[3.4 + \sum_{j=0}^{\infty} 2^{-j} \ln \frac{\hat{Q}(\beta_{j+1}) - \hat{Q}(\beta_0) + \varepsilon}{\hat{Q}(\beta_j) - \hat{Q}(\beta_0) + \varepsilon}\right]$$

for some $\varepsilon > 0$. Then at the stopping time k, we have

$$\hat{Q}(\beta^{(k)}) \leq \hat{Q}(\beta_0) + \varepsilon.$$

Proof For simplicity, let $f_j = \hat{Q}(\beta_j)$. For each $k = 1,2,\ldots$ before the stopping time, let j_k be the largest j such that

$$\hat{Q}(\beta^{(k)}) \geq f_j + f_j - f_0 + \varepsilon.$$

Let $\nu' = (\nu \min_j \rho_-(s+c(\beta_j)))/(\rho_0(\mathcal{B})c(\beta_0,\mathcal{B}))$.

We prove by contradiction. Suppose that the theorem does not hold, then for all k before stopping, we have $j_k \geq 0$.

For each $k > 0$ before stopping, if $j_k = j_{k-1} = j$, then we have from Lemma 21 (with $\beta = \beta_j$)

$$c(\beta^{(k)}) \leq c(\beta^{(k-1)}) + \nu'^{-1}2^{-j} \ln \frac{\|X\beta^{(k-1)} - \mathbf{y}\|_2^2 - f_j}{\|X\beta^{(k)} - \mathbf{y}\|_2^2 - f_j}.$$

Therefore for each $j \geq 0$, we have:

$$\sum_{k:j_k=j_{k-1}=j} [c(\beta^{(k)}) - c(\beta^{(k-1)})] \leq \nu'^{-1}2^{-j} \ln \frac{2(f_{j+1} - f_0 + \varepsilon)}{f_j - f_0 + \varepsilon}.$$

Moreover, for each $j \geq 0$, Lemma 21 (with $\beta = \beta_j$) implies that

$$\sum_{k:j_k=j, j_{k-1}>j} [c(\beta^{(k)}) - c(\beta^{(k-1)})] \leq \nu'^{-1} 2^{-j}.$$

Therefore we have

$$\sum_{k:j_k=j} [c(\beta^{(k)}) - c(\beta^{(k-1)})] \leq \nu'^{-1} 2^{-j} \left[1.7 + \ln \frac{f_{j+1} - f_0 + \varepsilon}{f_j - f_0 + \varepsilon} \right].$$

Now by summing over $j \geq 0$, we have

$$c(\beta^{(k)}) \leq 3.4\nu'^{-1} + \nu'^{-1} \sum_{j=0}^{\infty} 2^{-j} \ln \frac{f_{j+1} - f_0 + \varepsilon}{f_j - f_0 + \varepsilon} \leq s.$$

This is a contradiction because we know at stopping, we should have $c(\beta^{(k)}) > s$. ■

In the above theorem, we can see that if the signal is only weakly sparse, in that $(\hat{Q}(\beta_{j+1}) - \hat{Q}(\beta_0) + \varepsilon)/(\hat{Q}(\beta_j) - \hat{Q}(\beta_0) + \varepsilon)$ grows sub-exponentially in j, then we can choose $s = O(c(\beta_0, \mathcal{B}))$. This means that we can find $\beta^{(k)}$ of complexity $s = O(c(\beta_0, \mathcal{B}))$ to approximate a signal β_0. The worst case scenario is when $\hat{Q}(\beta_1) \approx \hat{Q}(0)$, which reduces to the $s = O(c(\beta_0, \mathcal{B}) \log(1/\varepsilon))$ complexity in Theorem 9.

As an application, we introduce the following concept of weakly sparse compressible target that generalizes the corresponding concept of compressible signal in standard sparsity from the compressive sensing literature (Donoho, 2006). A related extension has also appeared in Baraniuk et al. (2010).

Definition 23 *The target $\mathbb{E}\mathbf{y}$ is (a,q)-compressible with respect to block $\mathcal{B}$ if there exist constants $a, q > 0$ such that for each $s > 0$, $\exists \beta(s)$ such that $c(\beta(s), \mathcal{B}) \leq s$ and*

$$\frac{1}{n} \|X\beta(s) - \mathbb{E}\mathbf{y}\|_2^2 \leq as^{-q}.$$

Corollary 24 *Suppose that the target is (a,q)-compressible with respect to $\mathcal{B}$. Then with probability $1-\eta$, at the stopping time k, we have*

$$\hat{Q}(\beta^{(k)}) \leq \hat{Q}(\beta(s')) + 2na/s'^q + 2\sigma^2[\ln(2/\eta) + 1],$$

where

$$s' \leq \frac{s\,\nu}{(10+3q)\rho_0(\mathcal{B})} \min_{u \leq s'} \rho_-(s + c(\beta(u))).$$

Proof Given s', we consider $f_j = \min_{\ell \geq j} \hat{Q}(\beta(s'/2^\ell))$. We also assume that f_0 is achieved at $\ell_0 \geq 0$. Note that by Lemma 19, we have with probability $1 - 2^{-j-1}\eta$:

$$\begin{aligned} |\hat{Q}(\beta(s'/2^j)) - \|\mathbf{y} - \mathbb{E}\mathbf{y}\|_2^2| \leq & 2\|X\beta(s'/2^j) - \mathbb{E}\mathbf{y}\|_2^2 + 2\sigma^2[j + 1 + \ln(2/\eta)] \\ \leq & 2an2^{qj}/s'^q + 2\sigma^2[j + 1 + \ln(2/\eta)]. \end{aligned}$$

This means the above inequality holds for all j with probability $1-\eta$. Therefore

$$\begin{aligned} f_{j+1} - f_0 \leq & \hat{Q}(\beta(s'/2^{j+1})) - \hat{Q}(\beta(s')) \\ \leq & |\hat{Q}(\beta(s'/2^{j+1})) - \|\mathbf{y}-\mathbb{E}\mathbf{y}\|_2^2| + |\hat{Q}(\beta(s')) - \|\mathbf{y}-\mathbb{E}\mathbf{y}\|_2^2| \\ \leq & 4an2^{q(j+1)}/s'^q + 4\sigma^2[0.5j+1+\ln(2/\eta)]. \end{aligned}$$

Now, by taking $\varepsilon = 2an/s'^q + 2\sigma^2[\ln(2/\eta)+1]$ in Theorem 22, we obtain

$$\begin{aligned} \sum_{j=0}^{\infty} 2^{-j} \ln \frac{f_{j+1}-f_0+\varepsilon}{f_j - f_0 + \varepsilon} \leq & \sum_{j=\ell_0}^{\infty} 2^{-j} \ln(1+(f_{j+1}-f_0)/\varepsilon) \\ \leq & \sum_{j=\ell_0}^{\infty} 2^{-j} \ln(4+2(0.5j+2^{q(j+1)})) \\ \leq & \sum_{j=\ell_0}^{\infty} 2^{-j}(2+0.5j+\ln 2 + q(j+1)\ln 2) \leq 4.4 + 4(0.5+q\ln 2), \end{aligned}$$

where we have used the simple inequality $\ln(\alpha+2\beta) \leq 0.5\alpha + \ln(2\beta)$ when $\alpha, \beta \geq 1$. Therefore,

$$\begin{aligned} s \geq & \frac{\rho_0(\mathcal{B})s'}{\nu \min_{u \leq s'} \rho_-(s+c(\beta(u)))}(10+3q) \\ \geq & \frac{\rho_0(\mathcal{B})s'}{\nu \min_{u \leq s'} \rho_-(s+c(\beta(u)))} \left[3.4 + \sum_{j=0}^{\infty} 2^{-j} \ln \frac{f_{j+1}-f_0+\varepsilon}{f_j - f_0 + \varepsilon} \right]. \end{aligned}$$

This means that Theorem 22 can be applied to obtain the desired bound. ■

If we assume the underlying coding scheme is block coding generated by $\mathcal{B}$, then we have $\min_{u \leq s'} \rho_-(s+c(\beta(u))) \leq \rho_-(s+s')$. The corollary shows that we can approximate a compressible signal of complexity s' with complexity $s = O(qs')$ using greedy algorithm. This means the greedy algorithm obtains optimal rate for weakly-sparse compressible signals. The sample complexity suffers only a constant factor $O(q)$. Combine this result with Theorem 6, and take union bound, we have with probability $1-2\eta$, at stopping time k:

$$\begin{aligned} & \frac{1}{\sqrt{n}} \|X\beta^{(k)} - \mathbb{E}\mathbf{y}\|_2 \leq \sqrt{\frac{a}{s'^q}} + \sigma\sqrt{\frac{2\ln(6/\eta)}{n}} + 2\sigma\sqrt{\frac{7.4(s+c_0(\mathcal{B})) + 6.7\ln(6/\eta)}{n} + \frac{2a}{\sigma^2 s'^q}}, \\ & \|\beta^{(k)} - \beta(s')\|_2^2 \leq \frac{1}{\rho_-(s+s'+c_0(\mathcal{B}))} \left[\frac{15a}{s'^q} + \frac{37\sigma^2(s+c_0(\mathcal{B})) + 34\sigma^2\ln(6/\eta)}{n} \right]. \end{aligned}$$

Given a fixed n, we can obtain a convergence result by choosing s (and thus s') to optimize the right hand side. The resulting rate is optimal for the special case of standard sparsity, which implies that the bound has the optimal form for structured q-compressible targets. In particular, in compressive sensing applications where $\sigma = 0$, we obtain when sample size reaches $n = O(qs')$, the reconstruction performance is

$$\|\beta^{(k)} - \beta\|_2^2 = O(a/s'^q),$$

which matches that of the constrained coding complexity regularization method in (2) up to a constant $O(q)$. Since many real data involve weakly sparse signals, our result provides strong theoretical justification for the use of OMP in such problems. Our experiments are consistent with the theory.

References

A. Argyriou, T. Evgeniou, and M. Pontil. Convex multi-task feature learning. *Machine Learning Journal*, 73:243–272, 2008.

I. Johnstone B. Efron, T. Hastie and R. Tibshirani. Least angle regression. *Annals of Statistics*, 32: 407–499, 2004.

F. R. Bach. Consistency of the group lasso and multiple kernel learning. *Journal of Machine Learning Research*, 9:1179–1225, 2008.

R. Baraniuk, V. Cevher, M. F. Duarte, and C. Hegde. Model based compressive sensing. *IEEE Transactions on Information Theory*, 56:1982–2001, 2010.

F. Bunea, A. B. Tsybakov, and M. H. Wegkamp. Aggregation for Gaussian regression. *Annals of Statistics*, 35:1674–1697, 2007.

E. J. Candes and T. Tao. Decoding by linear programming. *IEEE Transaction on Information Theory*, 51:4203–4215, 2005.

V. Cevher, M. F. Duarte, C. Hegde, and R. Baraniuk. Sparse signal recovery using markov random fields. In *Proceeding of NIPS*, 2009a.

V. Cevher, P. Indyk, C. Hegde, and R. G. Baraniuk. Recovery of clustered sparse signals from compressive measurements. In *Proceeding of the 8th International Conference on Sampling Theory and Applications*, 2009b.

C. Chesneau and M. Hebiri. Some theoretical results on the grouped variables Lasso. *Mathematical Methods of Statistics*, 17:317–326, 2008.

T. M. Cover and J. A. Thomas. *Elements of Information Theory*. Wiley-Interscience, 1991.

A. d'Aspremont, F. Bach, and L. El Ghaoui. Optimal solutions for sparse principal component analysis. *Journal of Machine Learning Research*, 9:1269–1294, 2008.

L. Daudet. Sparse and structured decomposition of audio signals in overcomplete spaces. In *Proceeding of International Conference on Digital Audio Effects*, 2004.

D. Donoho. Compressed sensing. *IEEE Transactions on Information Theory*, 52:1289–1306, 2006.

D. Grimm, T. Netzer, and M. Schweighofer. A note on the representation of positive polynomials with structured sparsity. *Archiv der Mathematik*, 89:399–403, 2007.

J. Haupt and R. Nowak. Signal reconstruction from noisy projections. *IEEE Trans. Information Theory*, 52:4036–4048, 2006.

L. He and L. Carin. Exploiting structure in wavelet-based bayesian compressive sensing. *IEEE Transaction on Signal Processing*, 57:3488–3497, 2009a.

L. He and L. Carin. Exploiting structure in compressive sensing with a jpeg basis. In *Preprint*, 2009b.

J. Huang and T. Zhang. The benefit of group sparsity. *Annals of Statistics*, 38(4):1978–2004, 2010.

L. Jacob, G. Obozinski, and J. Vert. Group lasso with overlap and graph lasso. In *Proceedings of ICML*, 2009.

R. Jenatton, J. Y. Audibert, and F. Bach. Structured variable selection with sparsity-inducing norms. Technical report, Tech Report: arXiv:0904, 2009.

S. Ji, D. Dunson, and L. Carin. Multi-task compressive sensing. *IEEE Transactions on Signal Processing*, 2008. Accepted.

V. Koltchinskii and M. Yuan. Sparse recovery in large ensembles of kernel machines. In *Proceeding of COLT*, 2008.

M. Kowalski and B. Torresani. Structured sparsity: from mixed norms to structured shrinkage. In *Workshop on Signal Processing with Adaptive Sparse Representations*, 2009.

K. Lounici, M. Pontil, A. B. Tsybakov, and S. A. Van De Geer. Taking advantage of sparsity in multi-task learnng. In *Proceeding of COLT*, 2009.

A. Lozano, G. Swirszcz, and N. Abe. Group orthogonal matching pursuit for variable selection and prediction. In *Proceedings of NIPS*, 2009.

Y. M. Lu and M. N. Do. A theory for sampling signals from a union of subspaces. *IEEE Transactions on Signal Processing*, 56(6):2334–2345, 2008.

S. Mallat. *A Wavelet Tour of Signal Processing*. Academic Press, 1999.

Y. Nardi and A. Rinaldo. On the asymptotic properties of the group lasso estimator for linear models. *Electronic Journal of Statistics*, 2:605–633, 2008.

G. Obozinski, M. J. Wainwright, and M. I. Jordan. Union support recovery in high-dimensional multivariate regression. Technical Report 761, UC Berkeley, 2008.

G. Pisier. *The Volume of Convex Bodies and Banach Space Geometry*. Cambridge University Press, 1989.

J. M. Shapiro. Embedded image coding using zerotrees of wavelet coefficients. *IEEE Transactions on Signal Processing*, 41:3445–3462, 1993.

R. Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society*, 58:267–288, 1996.

J. Tropp and A. Gilbert. Signal recovery from random measurements via orthogonal matching pursuit. *IEEE Transactions on Information Theory*, 53(12):4655–4666, 2007.

D. Wipf and B. Rao. An empirical Bayesian strategy for solving the simultaneous sparse approximation problem. *IEEE Transactions on Signal Processing*, 55(7):3704–3716, 2007.

M. Yuan and Y. Lin. Model selection and estimation in regression with grouped variables. *Journal of The Royal Statistical Society Series B*, 68(1):49–67, 2006.

T. Zhang. Adaptive forward-backward greedy algorithm for learning sparse representations. *IEEE Transactions on Information Theory*, 57:4689–4708, 2011.

P. Zhao, G. Rocha, and B. Yu. The composite absolute penalties family for grouped and hierarchical variable selection. *The Annals of Statistics*, 37(6A):3468–3497, 2009.

Z. Zivkovic and F. Heijden. Efficient adaptive density estimation per image pixel for the task of background subtraction. *Pattern Recognition Letters*, 27(7):773–780, 2006.

Journal of Machine Learning Research 12 (2011) 3413-3430 Submitted 10/09; Revised 8/11; Published 12/11

A Simpler Approach to Matrix Completion

Benjamin Recht BRECHT@CS.WISC.EDU
Department of Computer Sciences
University of Wisconsin-Madison
Madison, WI 53706

Editor: Francis Bach

Abstract

This paper provides the best bounds to date on the number of randomly sampled entries required to reconstruct an unknown low-rank matrix. These results improve on prior work by Candès and Recht (2009), Candès and Tao (2009), and Keshavan et al. (2009). The reconstruction is accomplished by minimizing the nuclear norm, or sum of the singular values, of the hidden matrix subject to agreement with the provided entries. If the underlying matrix satisfies a certain incoherence condition, then the number of entries required is equal to a quadratic logarithmic factor times the number of parameters in the singular value decomposition. The proof of this assertion is short, self contained, and uses very elementary analysis. The novel techniques herein are based on recent work in quantum information theory.

Keywords: matrix completion, low-rank matrices, convex optimization, nuclear norm minimization, random matrices, operator Chernoff bound, compressed sensing

1. Introduction

Recovering a low-rank matrix from a partial sampling of its entries is a recurring problem in collaborative filtering (Rennie and Srebro, 2005; Koren et al., 2009) and dimensionality reduction (Weinberger and Saul, 2006; So and Ye, 2007). Estimating of low-rank models also arise in embedding problems (Linial et al., 1995) and multi-class learning (Argyriou et al., 2008; Obozinski et al., 2009). While a variety of heuristics have been developed across many disciplines, the general problem of finding the lowest rank matrix satisfying equality constraints is NP-hard. All known algorithms which can compute the lowest rank solution for all instances require time at least exponential in the dimensions of the matrix in both theory and practice (Chistov and Grigoriev, 1984).

In sharp contrast to such worst case pessimism, Candès and Recht (2009) showed that most low-rank matrices could be recovered from most sufficiently large sets of entries by computing the matrix of minimum *nuclear norm* that agreed with the provided entries, and furthermore the revealed set of entries could comprise a vanishing fraction of the entire matrix. The nuclear norm is equal to the sum of the singular values of a matrix and is the best convex lower bound of the rank function on the set of matrices whose singular values are all bounded by 1. The intuition behind this heuristic is that whereas the rank function counts the number of nonvanishing singular values, the nuclear norm sums their amplitude, much like how the ℓ_1 norm is a useful surrogate for counting the number of nonzeros in a vector. Moreover, the nuclear norm can be minimized subject to equality constraints via semidefinite programming.

Nuclear norm minimization had long been observed to produce very low-rank solutions in practice (see, for example, Beck and D'Andrea, 1998; Fazel, 2002; Fazel et al., 2001; Srebro, 2004;

Mesbahi and Papavassilopoulos, 1997), but only very recently was there any theoretical basis for when it produced the minimum rank solution. The first paper to provide such foundations was Recht et al. (2010), where the authors developed probabilistic techniques to study average case behavior and showed that the nuclear norm heuristic could solve most instances of the linearly-constrained rank-minimization problem assuming the number of linear constraints was sufficiently large. The results in Recht et al. (2010) inspired a groundswell of interest in theoretical guarantees for rank minimization, and these results lay the foundation for Candès and Recht (2009). Candès and Recht's bounds were subsequently improved by Candès and Tao (2009) and Keshavan et al. (2009) to show that one could, in special cases, reconstruct a low-rank matrix by observing a set of entries of size at most a polylogarithmic factor larger than the intrinsic dimension of the variety of rank r matrices.

This paper sharpens the results in Candès and Tao (2009) and Keshavan et al. (2009) to provide a bound on the number of entries required to reconstruct a low-rank matrix which is optimal up to a small numerical constant and one logarithmic factor. The main theorem makes minimal assumptions about the low-rank matrix of interest. Moreover, the proof is very short and relies on mostly elementary analysis.

In order to precisely state the main result, we need one definition. Candès and Recht observed that it is impossible to recover a matrix which is equal to zero in nearly all of its entries unless all of the entries of the matrix are observed (consider, for example, the rank one matrix which is equal to 1 in one entry and zeros everywhere else). In other words, the matrix cannot be mostly equal to zero on the observed entries. This motivated the following definition

Definition 1 *Let U be a subspace of $\mathbb{R}^n$ of dimension r and $\boldsymbol{P}_U$ be the orthogonal projection onto U. Then the* coherence *of U (vis-à-vis the standard basis $(\boldsymbol{e}_i)$) is defined to be*

$$\mu(U) \equiv \frac{n}{r} \max_{1 \le i \le n} \|\boldsymbol{P}_U \boldsymbol{e}_i\|^2.$$

Note that for any subspace, the smallest $\mu(U)$ can be is 1, achieved, for example, if U is spanned by vectors whose entries all have magnitude $1/\sqrt{n}$. The largest possible value for $\mu(U)$ is n/r which would correspond to any subspace that contains a standard basis element. If a matrix has row and column spaces with low coherence, then each entry can be expected to provide about the same amount of information.

Recall that the *nuclear norm* of an $n_1 \times n_2$ matrix $\boldsymbol{X}$ is the sum of the singular values of $\boldsymbol{X}$, $\|\boldsymbol{X}\|_* = \sum_{k=1}^{\min\{n_1,n_2\}} \sigma_k(\boldsymbol{X})$, where, here and below, $\sigma_k(\boldsymbol{X})$ denotes the kth largest singular value of $\boldsymbol{X}$. The main result of this paper is the following

Theorem 2 *Let $\boldsymbol{M}$ be an $n_1 \times n_2$ matrix of rank r with singular value decomposition $\boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^*$. Without loss of generality, impose the conventions $n_1 \le n_2$, $\boldsymbol{\Sigma}$ is $r \times r$, $\boldsymbol{U}$ is $n_1 \times r$ and $\boldsymbol{V}$ is $n_2 \times r$. Assume that*

A0 *The row and column spaces have coherences bounded above by some positive μ_0.*

A1 *The matrix $\boldsymbol{U}\boldsymbol{V}^*$ has a maximum entry bounded by $\mu_1\sqrt{r/(n_1 n_2)}$ in absolute value for some positive μ_1.*

Suppose m entries of $\boldsymbol{M}$ are observed with locations sampled uniformly at random. Then if

$$m \ge 32 \max\{\mu_1^2, \mu_0\}\, r(n_1+n_2)\, \beta \log^2(2n_2) \tag{1}$$

for some $\beta > 1$*, the minimizer to the problem*

$$\begin{array}{ll} \textit{minimize} & \|\boldsymbol{X}\|_* \\ \textit{subject to} & X_{ij} = M_{ij} \quad (i,j) \in \Omega. \end{array} \tag{2}$$

is unique and equal to $\boldsymbol{M}$ *with probability at least* $1 - 6\log(n_2)(n_1+n_2)^{2-2\beta} - n_2^{2-2\beta^{1/2}}$.

The assumptions **A0** and **A1** were introduced in Candès and Recht (2009). Both μ_0 and μ_1 may depend on r, n_1, or n_2. Moreover, note that $\mu_1 \leq \mu_0\sqrt{r}$ by the Cauchy-Schwarz inequality. As shown in Candès and Recht (2009), both subspaces selected from the uniform distribution and spaces constructed as the span of singular vectors with bounded entries are not only incoherent with the standard basis, but also obey **A1** with high probability for values of μ_1 at most logarithmic in n_1 and/or n_2. Applying this theorem to the models studied in Section 2 of Candès and Recht (2009), we find that there is a numerical constant c_u such that $c_u r(n_1+n_2)\log^5(n_2)$ entries are sufficient to reconstruct a rank r matrix whose row and column spaces are sampled from the Haar measure on the Grassmann manifold. If $r > \log(n_2)$, the number of entries can be reduced to $c_u r(n_1+n_2)\log^4(n_2)$. Similarly, there is a numerical constant c_i such that $c_i\mu_0^2 r(n_1+n_2)\log^3(n_2)$ entries are sufficient to recover a matrix of arbitrary rank r whose singular vectors have entries with magnitudes bounded by $\sqrt{\mu_0/n_1}$.

Theorem 2 greatly improves upon prior results. First of all, it has the weakest assumptions on the matrix to be recovered. In addition to assumption **A1**, Candès and Tao (2009) require a "strong incoherence condition" which is considerably more restrictive than the assumption **A0** in Theorem 2. Many of their results also require restrictions on the rank of $\boldsymbol{M}$, and their bounds depend superlinearly on μ_0. Keshavan et al. (2009) require the matrix rank to be no more than $\log(n_2)$, and require bounds on the maximum magnitude of the entries in $\boldsymbol{M}$ and the ratios $\sigma_1(\boldsymbol{M})/\sigma_r(\boldsymbol{M})$ and n_2/n_1. Theorem 2 makes no such assumptions about the rank, aspect ratio, nor condition number of $\boldsymbol{M}$. Moreover, (1) has a smaller log factor than Candès and Tao (2009), and features numerical constants that are both explicit and small.

Also note that there is not much room for improvement in the bound for m. It is a consequence of the coupon collector's problem that at least $n_2 \log n_2$ uniformly sampled entries are necessary just to guarantee that at least one entry in every row and column is observed with high probability. In addition, rank r matrices have $r(n_1+n_2-r)$ parameters, a fact that can be verified by counting the number of degrees of freedom in the singular value decomposition. Interestingly, Candès and Tao (2009) showed that $C\mu_0 n_2 r\log(n_2)$ entries were *necessary* for completion when the entries are sampled uniformly at random. Hence, (1) is optimal up to a small numerical constant times $\log(n_2)$.

Most importantly, the proof of Theorem 2 is short and straightforward. Candès and Recht employed sophisticated tools from the study of random variables on Banach spaces including decoupling tools and powerful moment inequalities for the norms of random matrices. Candès and Tao rely on intricate moment calculations spanning over 30 pages. The present work only uses basic matrix analysis, elementary large deviation bounds, and a noncommutative version of Bernstein's Inequality proven here in the Appendix.

The proof of Theorem 2 is adapted from a recent paper by Gross et al. (2010) in quantum information which considered the problem of reconstructing the density matrix of a quantum ensemble using as few measurements as possible. Their work extended results from Candès and Recht (2009) to the quantum regime by using special algebraic properties of quantum measurements. Their proof

followed a methodology analogous to the approach of Candès and Recht but had three main differences: they used a sampling with replacement model as a proxy for uniform sampling, deployed a powerful noncommutative Chernoff bound developed by Ahlswede and Winter (2002) for use in quantum information theory, and devised a simplified appeal to convex duality to guarantee exact recovery. In this paper, I adapt these strategies from Gross et al. (2010) to the matrix completion problem. In Section 3 I show how the sampling with replacement model bounds probabilities in the uniform sampling model, and present very short proofs of some of the main results in Candès and Recht (2009). Surprisingly, this yields a simple proof of Theorem 2, provided in Section 4, which has the least restrictive assumptions of any assertion proven thus far.[1]

2. Preliminaries and Notation

Before continuing, let us survey the notations used throughout the paper. I closely follow the conventions established in Candès and Recht (2009), and invite the reader to consult this reference for a more thorough discussion of the matrix completion problem and the associated convex geometry. A thorough introduction to the necessary matrix analysis used in this paper can be found in Recht et al. (2010).

Matrices are bold capital, vectors are bold lowercase and scalars or entries are not bold. For example, $\boldsymbol{X}$ is a matrix, and X_{ij} its (i,j)th entry. Likewise $\boldsymbol{x}$ is a vector, and x_i its ith component. If $\boldsymbol{u}_k \in \mathbb{R}^n$ for $1 \leq k \leq d$ is a collection of vectors, $[\boldsymbol{u}_1, \ldots, \boldsymbol{u}_d]$ will denote the $n \times d$ matrix whose kth column is $\boldsymbol{u}_k$. $\boldsymbol{e}_k$ will denote the kth standard basis vector in $\mathbb{R}^d$, equal to 1 in component k and 0 everywhere else. The dimension of $\boldsymbol{e}_k$ will always be clear from context. $\boldsymbol{X}^*$ and $\boldsymbol{x}^*$ denote the transpose of matrices $\boldsymbol{X}$ and vectors $\boldsymbol{x}$ respectively.

A variety of norms on matrices will be discussed. The spectral norm of a matrix is denoted by $\|\boldsymbol{X}\|$. The Euclidean inner product between two matrices is $\langle \boldsymbol{X}, \boldsymbol{Y} \rangle = \mathrm{Tr}(\boldsymbol{X}^* \boldsymbol{Y})$, and the corresponding Euclidean norm, called the Frobenius or Hilbert-Schmidt norm, is denoted $\|\boldsymbol{X}\|_F$. That is, $\|\boldsymbol{X}\|_F = \langle \boldsymbol{X}, \boldsymbol{X} \rangle^{1/2}$. The nuclear norm of a matrix $\boldsymbol{X}$ is $\|\boldsymbol{X}\|_*$. The maximum entry of $\boldsymbol{X}$ (in absolute value) is denoted by $\|\boldsymbol{X}\|_\infty \equiv \max_{ij} |X_{ij}|$. For vectors, the only norm applied is the usual Euclidean ℓ_2 norm, simply denoted as $\|\boldsymbol{x}\|$.

Linear transformations that act on matrices will be denoted by calligraphic letters. In particular, the identity operator will be denoted by $\mathcal{I}$. The spectral norm (the top singular value) of such an operator will be denoted by $\|\mathcal{A}\| = \sup_{\boldsymbol{X}:\|\boldsymbol{X}\|_F \leq 1} \|\mathcal{A}(\boldsymbol{X})\|_F$.

Fix once and for all a matrix $\boldsymbol{M}$ obeying the assumptions of Theorem 2. Let $\boldsymbol{u}_k$ (respectively $\boldsymbol{v}_k$) denote the kth column of $\boldsymbol{U}$ (respectively $\boldsymbol{V}$). Set $U \equiv \mathrm{span}\,(\boldsymbol{u}_1, \ldots, \boldsymbol{u}_r)$, and $V \equiv \mathrm{span}\,(\boldsymbol{v}_1, \ldots, \boldsymbol{v}_r)$. Also assume, without loss of generality, that $n_1 \leq n_2$. It is convenient to introduce the orthogonal decomposition $\mathbb{R}^{n_1 \times n_2} = T \oplus T^\perp$ where T is the linear space spanned by elements of the form $\boldsymbol{u}_k \boldsymbol{y}^*$ and $\boldsymbol{x} \boldsymbol{v}_k^*$, $1 \leq k \leq r$, where $\boldsymbol{x}$ and $\boldsymbol{y}$ are arbitrary, and $T^\perp$ is its orthogonal complement. $T^\perp$ is the subspace of matrices spanned by the family $(\boldsymbol{x}\boldsymbol{y}^*)$, where $\boldsymbol{x}$ (respectively $\boldsymbol{y}$) is any vector orthogonal to U (respectively V).

The orthogonal projection $\mathcal{P}_T$ onto T is given by

$$\mathcal{P}_T(\boldsymbol{Z}) = \boldsymbol{P}_U \boldsymbol{Z} + \boldsymbol{Z} \boldsymbol{P}_V - \boldsymbol{P}_U \boldsymbol{Z} \boldsymbol{P}_V, \tag{3}$$

1. Shortly after the appearance of a preprint of this manuscript, Gross (2011) announced a far reaching generalization of the techniques in Gross et al. (2010), providing bounds on recovering low-rank matrices in almost any basis. This work is more general than the work presented here, but the present paper achieves tighter constants and bounds and work directly with non-Hermitian matrices. The interested reader should consult Gross (2011) for more details.

where $\boldsymbol{P}_U$ and $\boldsymbol{P}_V$ are the orthogonal projections onto U and V respectively. Note here that while $\boldsymbol{P}_U$ and $\boldsymbol{P}_V$ are matrices, $\mathcal{P}_T$ is a linear operator mapping matrices to matrices. The orthogonal projection onto $T^\perp$ is given by

$$\mathcal{P}_{T^\perp}(\boldsymbol{Z}) = (I - \mathcal{P}_T)(\boldsymbol{Z}) = (\boldsymbol{I}_{n_1} - \boldsymbol{P}_U)\boldsymbol{Z}(\boldsymbol{I}_{n_2} - \boldsymbol{P}_V)$$

where $\boldsymbol{I}_d$ denotes the $d \times d$ identity matrix. It follows from the definition (3) of $\mathcal{P}_T$ that

$$\mathcal{P}_T(\boldsymbol{e}_a\boldsymbol{e}_b^*) = (\boldsymbol{P}_U\boldsymbol{e}_a)\boldsymbol{e}_b^* + \boldsymbol{e}_a(\boldsymbol{P}_V\boldsymbol{e}_b)^* - (\boldsymbol{P}_U\boldsymbol{e}_a)(\boldsymbol{P}_V\boldsymbol{e}_b)^*.$$

This gives

$$\|\mathcal{P}_T(\boldsymbol{e}_a\boldsymbol{e}_b^*)\|_F^2 = \langle \mathcal{P}_T(\boldsymbol{e}_a\boldsymbol{e}_b^*), \boldsymbol{e}_a\boldsymbol{e}_b^* \rangle = \|\boldsymbol{P}_U\boldsymbol{e}_a\|^2 + \|\boldsymbol{P}_V\boldsymbol{e}_b\|^2 - \|\boldsymbol{P}_U\boldsymbol{e}_a\|^2 \|\boldsymbol{P}_V\boldsymbol{e}_b\|^2 .$$

Since $\|\boldsymbol{P}_U\boldsymbol{e}_a\|^2 \le \mu(U)r/n_1$ and $\|\boldsymbol{P}_V\boldsymbol{e}_b\|^2 \le \mu(V)r/n_2$,

$$\|\mathcal{P}_T(\boldsymbol{e}_a\boldsymbol{e}_b^*)\|_F^2 \le \max\{\mu(U),\mu(V)\} r \frac{n_1+n_2}{n_1 n_2} \le \mu_0 r \frac{n_1+n_2}{n_1 n_2} . \tag{4}$$

I will make frequent use of this calculation throughout the sequel.

3. Sampling with Replacement

As discussed above, the main contribution of this work is an analysis of uniformly sampled sets of entries via the study of a sampling with replacement model. All of the previous work (e.g., Candès and Recht, 2009; Candès and Tao, 2009; Keshavan et al., 2009) studied a Bernoulli sampling model as a proxy for uniform sampling. There, each entry was revealed independently with probability equal to p. In all of these results, the theorem statements concerned sampling sets of m entries uniformly, but it was shown that probability of failure under Bernoulli sampling with $p = \frac{m}{n_1 n_2}$ closely approximated the probability of failure under uniform sampling. The present work will analyze the situation where each entry index is sampled independently from the uniform distribution on $\{1,\dots,n_1\} \times \{1,\dots,n_2\}$. This modification of the sampling model gives rise to all of the simplifications below.

It would appear that sampling with replacement is not suitable for analyzing matrix completion as one might encounter duplicate entries. However, just as is the case with Bernoulli sampling, bounding the likelihood of error when sampling with replacement allows us to bound the probability of the nuclear norm heuristic failing under uniform sampling.

Proposition 3 *The probability that the nuclear norm heuristic fails when the set of observed entries is sampled uniformly from the collection of sets of size m is less than or equal to the probability that the heuristic fails when m entries are sampled independently with replacement.*

Proof The proof follows the argument in Section II.C of Candés et al. (2006). Let Ω' be a collection of m entries, each sampled independently from the uniform distribution on $\{1,\dots,n_1\} \times \{1,\dots,n_2\}$. Let Ω_k denote a set of entries of size k sampled uniformly from all collections of entries of size k.

It follows that

$$\begin{aligned}\mathbb{P}(\text{Failure}(\Omega')) &= \sum_{k=0}^{m} \mathbb{P}(\text{Failure}(\Omega') \mid |\Omega'| = k)\,\mathbb{P}(|\Omega'| = k) \\ &= \sum_{k=0}^{m} \mathbb{P}(\text{Failure}(\Omega_k))\,\mathbb{P}(|\Omega'| = k) \\ &\geq \mathbb{P}(\text{Failure}(\Omega_m)) \sum_{k=0}^{m} \mathbb{P}(|\Omega'| = k) = \mathbb{P}(\text{Failure}(\Omega_m))\,.\end{aligned}$$

Where the inequality follows because $\mathbb{P}(\text{Failure}(\Omega_m)) \geq \mathbb{P}(\text{Failure}(\Omega_{m'}))$ if $m \leq m'$. That is, the probability decreases as the number of entries revealed is increased. ■

Surprisingly, changing the sampling model makes most of the theorems from Candès and Recht (2009) simple consequences of a noncommutative variant of Bernstein's Inequality.

Theorem 4 (Noncommutative Bernstein Inequality) *Let $\boldsymbol{X}_1, \ldots, \boldsymbol{X}_L$ be independent zero-mean random matrices of dimension $d_1 \times d_2$. Suppose $\rho_k^2 = \max\{\|\mathbb{E}[\boldsymbol{X}_k \boldsymbol{X}_k^*]\|, \|\mathbb{E}[\boldsymbol{X}_k^* \boldsymbol{X}_k]\|\}$ and $\|\boldsymbol{X}_k\| \leq M$ almost surely for all k. Then for any $\tau > 0$,*

$$\mathbb{P}\left[\left\|\sum_{k=1}^{L} \boldsymbol{X}_k\right\| > \tau\right] \leq (d_1 + d_2) \exp\left(\frac{-\tau^2/2}{\sum_{k=1}^{L} \rho_k^2 + M\tau/3}\right).$$

Note that in the case that $d_1 = d_2 = 1$, this is precisely the two sided version of the standard Bernstein Inequality. When the $\boldsymbol{X}_k$ are diagonal, this bound is the same as applying the standard Bernstein Inequality and a union bound to the diagonal of the matrix summation. Furthermore, observe that the right hand side is less than $(d_1 + d_2)\exp(-\frac{3}{8}\tau^2/(\sum_{k=1}^{L} \rho_k^2))$ as long as $\tau \leq \frac{1}{M}\sum_{k=1}^{L} \rho_k^2$. This condensed form of the inequality will be used exclusively throughout. Theorem 4 is a corollary of an Chernoff bound for finite dimensional operators developed by Ahlswede and Winter (2002). A similar inequality for symmetric i.i.d. matrices is proposed in Gross et al. (2010). The proof is provided in the Appendix.

Let us now record two theorems, proven for the Bernoulli model in Candès and Recht (2009), that admit very simple proofs in the sampling with replacement model. The theorem statements requires some additional notation. Let $\Omega = \{(a_k, b_k)\}_{k=1}^{l}$ be a collection of indices sampled uniformly with replacement. Set $\mathcal{R}_\Omega$ to be the operator

$$\mathcal{R}_\Omega(\boldsymbol{Z}) = \sum_{k=1}^{|\Omega|} \langle \boldsymbol{e}_{a_k} \boldsymbol{e}_{b_k}^*, \boldsymbol{Z} \rangle \boldsymbol{e}_{a_k} \boldsymbol{e}_{b_k}^*\,.$$

Note that the (i,j)th component of $\mathcal{R}_\Omega(\boldsymbol{X})$ is zero unless $(i,j) \in \Omega$. For $(i,j) \in \Omega$, $\mathcal{R}_\Omega(\boldsymbol{X})$ is equal to X_{ij} times the multiplicity of $(i,j) \in \Omega$. Unlike in previous work on matrix completion, $\mathcal{R}_\Omega$ is not a projection operator if there are duplicates in Ω. Nonetheless, this does not adversely affect the argument, and $\mathcal{R}_\Omega(\boldsymbol{X}) = 0$ if and only if $X_{ab} = 0$ for all $(a,b) \in \Omega$. Moreover, we can show that the maximum duplication of any entry is always less than $\frac{8}{3}\log(n_2)$ with very high probability.

Proposition 5 *With probability at least $1 - n_2^{2-2\beta}$, the maximum number of repetitions of any entry in Ω is less than $\frac{8}{3}\beta\log(n_2)$ for $n_2 \geq 9$ and $\beta > 1$.*

Proof This assertion can be proven by applying a standard Chernoff bound for the Bernoulli distribution. Note that for a fixed entry, the probability it is sampled more than t times is equal to the probability of more than t heads occurring in a sequence of m tosses where the probability of a head is $\frac{1}{n_1 n_2}$. This probability can be upper bounded by

$$\mathbb{P}[\text{more than } t \text{ heads in } m \text{ trials}] \leq \left(\frac{m}{n_1 n_2 t}\right)^t \exp\left(t - \frac{m}{n_1 n_2}\right)$$

(see Hagerup and Rüb, 1990, for example). Applying the union bound over all of the $n_1 n_2$ entries and the fact that $\frac{m}{n_1 n_2} < 1$, we have

$$\begin{aligned}\mathbb{P}[&\text{any entry is selected more than } \tfrac{8}{3}\beta\log(n_2) \text{ times}] \\ &\leq n_1 n_2 \left(\tfrac{8}{3}\beta\log(n_2)\right)^{-\frac{8}{3}\beta\log(n_2)} \exp\left(\tfrac{8}{3}\beta\log(n_2)\right) \\ &\leq n_2^{2-2\beta}\end{aligned}$$

when $n_2 \geq 9$. ■

This application of the Chernoff bound is very crude, and much tighter bounds can be derived using more careful analysis. For example in Gonnet (1981), the maximum oversampling is shown to be bounded by $O(\frac{\log(n_2)}{\log\log(n_2)})$. For our purposes here, the loose upper bound provided by Proposition 5 will be more than sufficient.

In addition to this bound on the norm of $\mathcal{R}_\Omega$, the following theorem asserts that the operator $\mathcal{P}_T \mathcal{R}_\Omega \mathcal{P}_T$ is also very close to an isometry on T if the number of sampled entries is sufficiently large. This result is analgous to the Theorem 4.1 in Candès and Recht (2009) for the Bernoulli model, whose proof uses several powerful theorems from the study of probability in Banach spaces. Here, one only needs to compute a few low order moments and then apply Theorem 4.

Theorem 6 *Suppose Ω is a set of entries of size m sampled independently and uniformly with replacement. Then for all $\beta > 1$,*

$$\frac{n_1 n_2}{m}\left\|\mathcal{P}_T \mathcal{R}_\Omega \mathcal{P}_T - \frac{m}{n_1 n_2}\mathcal{P}_T\right\| \leq \sqrt{\frac{16\mu_0 r(n_1+n_2)\beta\log(n_2)}{3m}}$$

with probability at least $1 - 2n_2^{2-2\beta}$ provided that $m > \frac{16}{3}\mu_0 r(n_1+n_2)\beta\log(n_2)$.

Proof Decompose any matrix $\boldsymbol{Z}$ as $\boldsymbol{Z} = \sum_{ab}\langle \boldsymbol{Z}, \boldsymbol{e}_a \boldsymbol{e}_b^*\rangle \boldsymbol{e}_a \boldsymbol{e}_b^*$ so that

$$\mathcal{P}_T(\boldsymbol{Z}) = \sum_{ab}\langle \mathcal{P}_T(\boldsymbol{Z}), \boldsymbol{e}_a \boldsymbol{e}_b^*\rangle \boldsymbol{e}_a \boldsymbol{e}_b^* = \sum_{ab}\langle \boldsymbol{Z}, \mathcal{P}_T(\boldsymbol{e}_a \boldsymbol{e}_b^*)\rangle \boldsymbol{e}_a \boldsymbol{e}_b^*. \tag{5}$$

For $k = 1, \ldots, m$ sample (a_k, b_k) from $\{1, \ldots, n_1\} \times \{1, \ldots, n_2\}$ uniformly with replacement. Then $\mathcal{R}_\Omega \mathcal{P}_T(\boldsymbol{Z}) = \sum_{k=1}^m \langle \boldsymbol{Z}, \mathcal{P}_T(\boldsymbol{e}_{a_k} \boldsymbol{e}_{b_k}^*)\rangle \, \boldsymbol{e}_{a_k} \boldsymbol{e}_{b_k}^*$ which gives

$$(\mathcal{P}_T \mathcal{R}_\Omega \mathcal{P}_T)(\boldsymbol{Z}) = \sum_{k=1}^m \langle \boldsymbol{Z}, \mathcal{P}_T(\boldsymbol{e}_{a_k} \boldsymbol{e}_{b_k}^*)\rangle \, \mathcal{P}_T(\boldsymbol{e}_{a_k} \boldsymbol{e}_{b_k}^*).$$

Now the fact that the operator $\mathcal{P}_T \mathcal{R}_\Omega \mathcal{P}_T$ does not deviate from its expected value

$$\mathbb{E}(\mathcal{P}_T \mathcal{R}_\Omega \mathcal{P}_T) = \mathcal{P}_T(\mathbb{E}\,\mathcal{R}_\Omega)\mathcal{P}_T = \mathcal{P}_T(\frac{m}{n_1 n_2} I)\mathcal{P}_T = \frac{m}{n_1 n_2}\mathcal{P}_T$$

in the spectral norm can be proven using the Noncommutative Bernstein Inequality.

To proceed, define the operator $\mathcal{T}_{ab}$ which maps $\boldsymbol{Z}$ to $\langle \mathcal{P}_T(\boldsymbol{e}_a \boldsymbol{e}_b^*), \boldsymbol{Z}\rangle \mathcal{P}_T(\boldsymbol{e}_a \boldsymbol{e}_b^*)$. This operator is rank one, has operator norm $\|\mathcal{T}_{ab}\| = \|\mathcal{P}_T(\boldsymbol{e}_a \boldsymbol{e}_b^*)\|_F^2$, and we have $\mathcal{P}_T = \sum_{a,b} \mathcal{T}_{ab}$ by (5). Hence, for $k = 1, \ldots, m$, $\mathbb{E}[\mathcal{T}_{a_k b_k}] = \frac{1}{n_1 n_2}\mathcal{P}_T$.

Observe that if $\boldsymbol{A}$ and $\boldsymbol{B}$ are positive semidefinite, we have $\|\boldsymbol{A} - \boldsymbol{B}\| \leq \max\{\|\boldsymbol{A}\|, \|\boldsymbol{B}\|\}$. Using this fact, we can compute the bound

$$\|\mathcal{T}_{a_k b_k} - \tfrac{1}{n_1 n_2}\mathcal{P}_T\| \leq \max\{\|\mathcal{P}_T(\boldsymbol{e}_{a_k} \boldsymbol{e}_{b_k}^*)\|_F^2, \tfrac{1}{n_1 n_2}\} \leq \mu_0 r \frac{n_1 + n_2}{n_1 n_2},$$

where the final inequality follows from (4). We also have

$$\begin{aligned}
\|\,\mathbb{E}[(\mathcal{T}_{a_k b_k} - \tfrac{1}{n_1 n_2}\mathcal{P}_T)^2]\| &= \|\,\mathbb{E}[\|\mathcal{P}_T(\boldsymbol{e}_{a_k} \boldsymbol{e}_{b_k}^*)\|_F^2 \mathcal{T}_{a_k b_k}] - \frac{1}{n_1^2 n_2^2}\mathcal{P}_T]\| \\
&\leq \max\{\|\,\mathbb{E}[\|\mathcal{P}_T(\boldsymbol{e}_{a_k} \boldsymbol{e}_{b_k}^*)\|_F^2 \mathcal{T}_{a_k b_k}]\|, \frac{1}{n_1^2 n_2^2}\} \\
&\leq \max\{\|\,\mathbb{E}[\mathcal{T}_{a_k b_k}]\|\,\mu_0 r \frac{n_1 + n_2}{n_1 n_2}, \frac{1}{n_1^2 n_2^2}\} \leq \mu_0 r \frac{n_1 + n_2}{n_1^2 n_2^2}.
\end{aligned}$$

The theorem now follows by applying the Noncommutative Bernstein Inequality. ■

The next theorem is an analog of Theorem 6.3 in Candès and Recht (2009) or Lemma 3.2 in Keshavan et al. (2009). This theorem asserts that for a fixed matrix, if one sets all of the entries not in Ω to zero it remains close to a multiple of the original matrix in the operator norm.

Theorem 7 *Suppose Ω is a set of entries of size m sampled independently and uniformly with replacement and let $\boldsymbol{Z}$ be a fixed $n_1 \times n_2$ matrix. Assume without loss of generality that $n_1 \leq n_2$, Then for all $\beta > 1$,*

$$\left\| \left(\frac{n_1 n_2}{m} \mathcal{R}_\Omega - I \right)(\boldsymbol{Z}) \right\| \leq \sqrt{\frac{8\beta n_1 n_2^2 \log(n_1 + n_2)}{3m}} \|\boldsymbol{Z}\|_\infty$$

with probability at least $1 - (n_1 + n_2)^{1-\beta}$ provided that $m > 6\beta n_1 \log(n_1 + n_2)$.

Proof First observe that the operator norm can be upper bounded by a multiple of the matrix infinity norm

$$\begin{aligned}
\|\boldsymbol{Z}\| = \sup_{\substack{\|\boldsymbol{x}\|=1 \\ \|\boldsymbol{y}\|=1}} \sum_{a,b} Z_{ab} y_a x_b &\leq \left(\sum_{a,b} Z_{ab}^2 y_a^2 \right)^{1/2} \left(\sum_{a,b} x_b^2 \right)^{1/2} \\
&\leq \sqrt{n_2} \max_a \left(\sum_b Z_{ab}^2 \right)^{1/2} \\
&\leq \sqrt{n_1 n_2} \|\boldsymbol{Z}\|_\infty .
\end{aligned}$$

Note that $\frac{n_1 n_2}{m}\mathcal{R}_\Omega(\boldsymbol{Z}) - \boldsymbol{Z} = \frac{1}{m}\sum_{k=1}^m n_1 n_2 Z_{a_k b_k} \boldsymbol{e}_{a_k}\boldsymbol{e}_{b_k}^* - \boldsymbol{Z}$. This is a sum of zero-mean random matrices, and $\|n_1 n_2 Z_{a_k b_k}\boldsymbol{e}_{a_k}\boldsymbol{e}_{b_k}^* - \boldsymbol{Z}\| \leq \|n_1 n_2 Z_{a_k b_k}\boldsymbol{e}_{a_k}\boldsymbol{e}_{b_k}^*\| + \|\boldsymbol{Z}\| < \frac{3}{2} n_1 n_2 \|\boldsymbol{Z}\|_\infty$ for $n_1 \geq 2$. We also have

$$\begin{aligned}
\|\,\mathbb{E}[(n_1 n_2 Z_{a_k b_k}\boldsymbol{e}_{a_k}\boldsymbol{e}_{b_k}^* - \boldsymbol{Z})^*(n_1 n_2 Z_{a_k b_k}\boldsymbol{e}_{a_k}\boldsymbol{e}_{b_k}^* - \boldsymbol{Z})]\| & \\
&= \left\| n_1 n_2 \sum_{c,d} Z_{cd}^2 \boldsymbol{e}_d \boldsymbol{e}_d^* - \boldsymbol{Z}^*\boldsymbol{Z} \right\| \\
&\leq \max\left\{ \left\| n_1 n_2 \sum_{c,d} Z_{cd}^2 \boldsymbol{e}_d \boldsymbol{e}_d^* \right\|, \|\boldsymbol{Z}^*\boldsymbol{Z}\| \right\} \\
&\leq n_1 n_2^2 \|\boldsymbol{Z}\|_\infty^2
\end{aligned}$$

where we again use the fact that $\|\boldsymbol{A} - \boldsymbol{B}\| \leq \max\{\|\boldsymbol{A}\|, \|\boldsymbol{B}\|\}$ for positive semidefinite $\boldsymbol{A}$ and $\boldsymbol{B}$. A similar calculation holds for $(n_1 n_2 Z_{a_k b_k}\boldsymbol{e}_{a_k}\boldsymbol{e}_{b_k}^* - \boldsymbol{Z})(n_1 n_2 Z_{a_k b_k}\boldsymbol{e}_{a_k}\boldsymbol{e}_{b_k}^* - \boldsymbol{Z})^*$. The theorem now follows by the Noncommutative Bernstein Inequality. ■

Finally, the following Lemma is required to prove Theorem 2. Succinctly, it says that for a fixed matrix in T, the operator $\mathcal{P}_T\mathcal{R}_\Omega$ does not increase the matrix infinity norm.

Lemma 8 *Suppose Ω is a set of entries of size m sampled independently and uniformly with replacement and let $\boldsymbol{Z} \in T$ be a fixed $n_1 \times n_2$ matrix. Assume without loss of generality that $n_1 \leq n_2$. Then for all $\beta > 2$,*

$$\left\| \frac{n_1 n_2}{m}\mathcal{P}_T\mathcal{R}_\Omega(\boldsymbol{Z}) - \boldsymbol{Z} \right\|_\infty \leq \sqrt{\frac{8\beta\mu_0 r(n_1+n_2)\log n_2}{3m}}\|\boldsymbol{Z}\|_\infty$$

with probability at least $1 - 2n_2^{2-\beta}$ provided that $m > \frac{8}{3}\beta\mu_0 r(n_1+n_2)\log n_2$.

Proof This lemma can be proven using the standard Bernstein Inequality. For each matrix index (c,d), sample (a,b) uniformly at random to define the random variable $\xi_{cd} = \langle \boldsymbol{e}_c\boldsymbol{e}_d^*, n_1 n_2 \langle \boldsymbol{e}_a\boldsymbol{e}_b^*, \boldsymbol{Z}\rangle \mathcal{P}_T(\boldsymbol{e}_a\boldsymbol{e}_b^*) - \boldsymbol{Z}\rangle$. We have $\mathbb{E}[\xi_{cd}] = 0$, $|\xi_{cd}| \leq \mu_0 r(n_1+n_2)\|\boldsymbol{Z}\|_\infty$, and

$$\begin{aligned}
\mathbb{E}[\xi_{cd}^2] &= \frac{1}{n_1 n_2}\sum_{a,b}\langle \boldsymbol{e}_c\boldsymbol{e}_d^*, n_1 n_2\langle \boldsymbol{e}_a\boldsymbol{e}_b^*, \boldsymbol{Z}\rangle \mathcal{P}_T(\boldsymbol{e}_a\boldsymbol{e}_b^*) - \boldsymbol{Z}\rangle^2 \\
&= n_1 n_2 \sum_{a,b}\langle \mathcal{P}_T(\boldsymbol{e}_c\boldsymbol{e}_d^*), \boldsymbol{e}_a\boldsymbol{e}_b^*\rangle^2 \langle \boldsymbol{e}_a\boldsymbol{e}_b^*, \boldsymbol{Z}\rangle^2 - Z_{cd}^2 \\
&\leq n_1 n_2 \|\mathcal{P}_T(\boldsymbol{e}_c\boldsymbol{e}_d^*)\|_F^2 \|\boldsymbol{Z}\|_\infty^2 \leq \mu_0 r(n_1+n_2)\|\boldsymbol{Z}\|_\infty^2 .
\end{aligned}$$

Since the (c,d) entry of $\frac{n_1 n_2}{m}\mathcal{P}_T\mathcal{R}_\Omega(\boldsymbol{Z}) - \boldsymbol{Z}$ is identically distributed to $\frac{1}{m}\sum_{k=1}^m \xi_{cd}^{(k)}$, where $\xi_{cd}^{(k)}$ are i.i.d. copies of ξ_{cd}, we have by Bernstein's Inequality and the union bound:

$$\Pr\left[\left\| \frac{n_1 n_2}{m}\mathcal{P}_T\mathcal{R}_\Omega(\boldsymbol{Z}) - \boldsymbol{Z} \right\|_\infty > \sqrt{\frac{8\beta\mu_0 r(n_1+n_2)\log(n_2)}{3m}}\|\boldsymbol{Z}\|_\infty \right] \leq 2n_2^{2-\beta}.$$

■

4. Proof of Theorem 2

The proof follows the program developed in Gross et al. (2010) which itself adapted the strategy proposed in Candès and Recht (2009). The main idea is to approximate a dual feasible solution of (2) which certifies that $\boldsymbol{M}$ is the unique minimum nuclear norm solution. In Candès and Recht (2009) such a certificate was constructed via an infinite series using a construction developed in the compressed sensing literature (See, for example Candés et al., 2006; Fuchs, 2004). The terms in this series were then analyzed individually using the decoupling inequalities of de la Peña and Montgomery-Smith (1995). Truncating the infinite series after 4 terms gave their result. In Candès and Tao (2009), the authors bounded the contribution of $O(\log(n_2))$ terms in this series using intensive combinatorial analysis of each term. The insight in Gross et al. (2010) was that, when sampling observations with replacement, a dual feasible solution could be closely approximated by a modified series where each term involved the product of independent random variables. This change in the sampling model allows one to avoid decoupling inequalities and gives rise to the dramatic simplification here.

To proceed, recall again that by Proposition 3 it suffices to consider the scenario when the entries are sampled independently and uniformly with replacement. I will first develop the main argument of the proof assuming many conditions hold with high probability. The proof is completed by subsequently bounding probability that all of these events hold. Suppose that

$$\frac{n_1 n_2}{m}\left\|\mathcal{P}_T\mathcal{R}_\Omega\mathcal{P}_T - \frac{m}{n_1 n_2}\mathcal{P}_T\right\| \leq \frac{1}{2}, \qquad \|\mathcal{R}_\Omega\| \leq \tfrac{8}{3}\beta^{1/2}\log(n_2)\,. \tag{6}$$

Also suppose there exists a $\boldsymbol{Y}$ in the range of $\mathcal{R}_\Omega$ such that

$$\|\mathcal{P}_T(\boldsymbol{Y}) - \boldsymbol{U}\boldsymbol{V}^*\|_F \leq \sqrt{\frac{r}{2n_2}}, \qquad \|\mathcal{P}_{T^\perp}(\boldsymbol{Y})\| < \frac{1}{2}\,. \tag{7}$$

If (6) holds, then for any $\boldsymbol{Z} \in \ker\mathcal{R}_\Omega$, $\mathcal{P}_T(\boldsymbol{Z})$ cannot be too large. Indeed, we have

$$0 = \|\mathcal{R}_\Omega(\boldsymbol{Z})\|_F \geq \|\mathcal{R}_\Omega\mathcal{P}_T(\boldsymbol{Z})\|_F - \|\mathcal{R}_\Omega\mathcal{P}_{T^\perp}(\boldsymbol{Z})\|_F\,.$$

Now observe that

$$\|\mathcal{R}_\Omega\mathcal{P}_T(\boldsymbol{Z})\|_F^2 = \langle \boldsymbol{Z}, \mathcal{P}_T\mathcal{R}_\Omega^2\mathcal{P}_T(\boldsymbol{Z})\rangle \geq \langle \boldsymbol{Z}, \mathcal{P}_T\mathcal{R}_\Omega\mathcal{P}_T(\boldsymbol{Z})\rangle \geq \frac{m}{2n_1n_2}\|\mathcal{P}_T(\boldsymbol{Z})\|_F^2$$

and $\|\mathcal{R}_\Omega\mathcal{P}_{T^\perp}(\boldsymbol{Z})\|_F \leq \frac{8}{3}\beta^{1/2}\log(n_2)\|\mathcal{P}_{T^\perp}(\boldsymbol{Z})\|_F$. Collecting these facts gives that for any $\boldsymbol{Z} \in \ker\mathcal{R}_\Omega$,

$$\|\mathcal{P}_{T^\perp}(\boldsymbol{Z})\|_F \geq \sqrt{\frac{9m}{128\beta n_1 n_2\log^2(n_2)}}\|\mathcal{P}_T(\boldsymbol{Z})\|_F > \sqrt{\frac{2r}{n_2}}\|\mathcal{P}_T(\boldsymbol{Z})\|_F\,.$$

Now recall that $\|\boldsymbol{A}\|_* = \sup_{\|B\|\leq 1}\langle \boldsymbol{A}, \boldsymbol{B}\rangle$. For $\boldsymbol{Z} \in \ker\mathcal{R}_\Omega$, pick $\boldsymbol{U}_\perp$ and $\boldsymbol{V}_\perp$ such that $[\boldsymbol{U}, \boldsymbol{U}_\perp]$ and $[\boldsymbol{V}, \boldsymbol{V}_\perp]$ are unitary matrices and that $\langle \boldsymbol{U}_\perp\boldsymbol{V}_\perp^*, \mathcal{P}_{T^\perp}(\boldsymbol{Z})\rangle = \|\mathcal{P}_{T^\perp}(\boldsymbol{Z})\|_*$. Then it follows that

$$\begin{aligned}
\|\boldsymbol{M}+\boldsymbol{Z}\|_* &\geq \langle \boldsymbol{U}\boldsymbol{V}^* + \boldsymbol{U}_\perp\boldsymbol{V}_\perp^*, \boldsymbol{M}+\boldsymbol{Z}\rangle \\
&= \|\boldsymbol{M}\|_* + \langle \boldsymbol{U}\boldsymbol{V}^* + \boldsymbol{U}_\perp\boldsymbol{V}_\perp^*, \boldsymbol{Z}\rangle \\
&= \|\boldsymbol{M}\|_* + \langle \boldsymbol{U}\boldsymbol{V}^* - \mathcal{P}_T(\boldsymbol{Y}), \mathcal{P}_T(\boldsymbol{Z})\rangle + \langle \boldsymbol{U}_\perp\boldsymbol{V}_\perp^* - \mathcal{P}_{T^\perp}(\boldsymbol{Y}), \mathcal{P}_{T^\perp}(\boldsymbol{Z})\rangle \\
&> \|\boldsymbol{M}\|_* - \sqrt{\frac{r}{2n_2}}\|\mathcal{P}_T(\boldsymbol{Z})\|_F + \frac{1}{2}\|\mathcal{P}_{T^\perp}(\boldsymbol{Z})\|_* \geq \|\boldsymbol{M}\|_*\,.
\end{aligned}$$

The first inequality holds from the variational characterization of the nuclear norm. We also used the fact that $\langle \boldsymbol{Y}, \boldsymbol{Z} \rangle = 0$ for all $\boldsymbol{Z} \in \ker \mathcal{R}_\Omega$. Thus, if a $\boldsymbol{Y}$ exists obeying (7), we have that for any $\boldsymbol{X}$ obeying $\mathcal{R}_\Omega(\boldsymbol{X} - \boldsymbol{M}) = \boldsymbol{0}$, $\|\boldsymbol{X}\|_* > \|\boldsymbol{M}\|_*$. That is, any if $\boldsymbol{X}$ has $M_{ab} = X_{ab}$ for all $(a,b) \in \Omega$, $\boldsymbol{X}$ has strictly larger nuclear norm than $\boldsymbol{M}$, and hence $\boldsymbol{M}$ is the unique minimizer of (2). The remainder of the proof shows that such a $\boldsymbol{Y}$ exists with high probability.

To this end, partition $1, \ldots, m$ into p partitions of size q. By assumption, we may choose

$$q \geq \frac{128}{3} \max\{\mu_0, \mu_1^2\} r(n_1 + n_2) \beta \log(n_1 + n_2) \quad \text{and} \quad p \geq \frac{3}{4} \log(2n_2).$$

Let Ω_j denote the set of indices corresponding to the jth partition. Note that each of these partitions are independent of one another when the indices are sampled with replacement. Assume that

$$\frac{n_1 n_2}{q} \left\| \mathcal{P}_T \mathcal{R}_{\Omega_k} \mathcal{P}_T - \frac{q}{n_1 n_2} \mathcal{P}_T \right\| \leq \frac{1}{2} \tag{8}$$

for all k. Define $\boldsymbol{W}_0 = \boldsymbol{U}\boldsymbol{V}^*$ and set $\boldsymbol{Y}_k = \frac{n_1 n_2}{q} \sum_{j=1}^{k} \mathcal{R}_{\Omega_j}(\boldsymbol{W}_{j-1})$, $\boldsymbol{W}_k = \boldsymbol{U}\boldsymbol{V}^* - \mathcal{P}_T(\boldsymbol{Y}_k)$ for $k = 1, \ldots, p$. Then

$$\begin{aligned} \|\boldsymbol{W}_k\|_F &= \left\| \boldsymbol{W}_{k-1} - \frac{n_1 n_2}{q} \mathcal{P}_T \mathcal{R}_{\Omega_k}(\boldsymbol{W}_{k-1}) \right\|_F \\ &= \left\| (\mathcal{P}_T - \frac{n_1 n_2}{q} \mathcal{P}_T \mathcal{R}_{\Omega_k} \mathcal{P}_T)(\boldsymbol{W}_{k-1}) \right\|_F \\ &\leq \frac{1}{2} \|\boldsymbol{W}_{k-1}\|_F, \end{aligned}$$

and it follows that $\|\boldsymbol{W}_k\|_F \leq 2^{-k} \|\boldsymbol{W}_0\|_F = 2^{-k}\sqrt{r}$. Since $p \geq \frac{3}{4}\log(2n_2) \geq \frac{1}{2}\log_2(2n_2) = \log_2 \sqrt{2n_2}$, then $\boldsymbol{Y} = \boldsymbol{Y}_p$ will satisfy the first inequality of (7). Also suppose that

$$\left\| \boldsymbol{W}_{k-1} - \frac{n_1 n_2}{q} \mathcal{P}_T \mathcal{R}_{\Omega_k}(\boldsymbol{W}_{k-1}) \right\|_\infty \leq \frac{1}{2} \|\boldsymbol{W}_{k-1}\|_\infty, \tag{9}$$

$$\left\| \left(\frac{n_1 n_2}{q} \mathcal{R}_{\Omega_j} - I \right) (\boldsymbol{W}_{j-1}) \right\| \leq \sqrt{\frac{8 n_1 n_2^2 \beta \log(n_1 + n_2)}{3q}} \|\boldsymbol{W}_{j-1}\|_\infty \tag{10}$$

for $k = 1, \ldots, p$.

To see that $\|\mathcal{P}_{T^\perp}(\boldsymbol{Y}_p)\| \leq \frac{1}{2}$ when (9) and (10) hold, observe $\|\boldsymbol{W}_k\|_\infty \leq 2^{-k}\|\boldsymbol{U}\boldsymbol{V}^*\|_\infty$, and it follows that

$$\begin{aligned}
\|\mathcal{P}_{T^\perp}\boldsymbol{Y}_p\| &\leq \sum_{j=1}^{p} \|\tfrac{n_1 n_2}{q}\mathcal{P}_{T^\perp}\mathcal{R}_{\Omega_j}\boldsymbol{W}_{j-1}\| \\
&= \sum_{j=1}^{p} \|\mathcal{P}_{T^\perp}(\tfrac{n_1 n_2}{q}\mathcal{R}_{\Omega_j}\boldsymbol{W}_{j-1} - \boldsymbol{W}_{j-1})\| \\
&\leq \sum_{j=1}^{p} \|(\tfrac{n_1 n_2}{q}\mathcal{R}_{\Omega_j} - I)(\boldsymbol{W}_{j-1})\| \\
&\leq \sum_{j=1}^{p} \sqrt{\frac{8n_1 n_2^2 \beta \log(n_1+n_2)}{3q}}\|\boldsymbol{W}_{j-1}\|_\infty \\
&= 2\sum_{j=1}^{p} 2^{-j}\sqrt{\frac{8n_1 n_2^2 \beta \log(n_1+n_2)}{3q}}\|\boldsymbol{U}\boldsymbol{V}^*\|_\infty < \sqrt{\frac{32\mu_1^2 r n_2 \beta \log(n_1+n_2)}{3q}} < 1/2
\end{aligned}$$

since $q > \frac{128}{3}\mu_1^2 r n_2 \beta \log(n_1+n_2)$. The first inequality follows from the triangle inequality. The second line follows because $\boldsymbol{W}_{j-1} \in T$ for all j. The third line follows because, for any $\boldsymbol{Z}$,

$$\|\mathcal{P}_{T^\perp}(\boldsymbol{Z})\| = \|(\boldsymbol{I}_{n_1} - \boldsymbol{P}_U)\boldsymbol{Z}(\boldsymbol{I}_{n_2} - \boldsymbol{P}_V)\| \leq \|\boldsymbol{Z}\|.$$

The fourth line applies (10). The next line follows from (9). The final line follows from the assumption **A1**.

All that remains is to bound the probability that all of the invoked events hold. With m satisfying the bound in the main theorem statement, the first inequality in (6) fails to hold with probability at most $2n_2^{2-2\beta}$ by Theorem 6, and the second inequality fails to hold with probability at most $n_2^{2-2\beta^{1/2}}$ by Proposition 5. For all k, (8) fails to hold with probability at most $2n_2^{2-2\beta}$, (9) fails to hold with probability at most $2n_2^{2-2\beta}$, and (10) fails to hold with probability at most $(n_1+n_2)^{1-2\beta}$. Summing these all together, all of the events hold with probability at least

$$1 - 6\log(n_2)(n_1+n_2)^{2-2\beta} - n_2^{2-2\beta^{1/2}}$$

by the union bound. This completes the proof.

5. Discussion and Conclusions

The results proven here are nearly optimal, but small improvements can possibly be made. The numerical constant 32 in the statement of the theorem may be reducible by more clever bookkeeping, and it may be possible to derive a linear dependence on the logarithm of the matrix dimensions. But further reduction is not possible because of the necessary conditions provided by Candès and Tao. One minor improvement that could be made would be to remove the assumption **A1**. For instance, while μ_1 is known to be small in most of the models of low-rank matrices that have been analyzed, no one has shown that an assumption of the form **A1** is necessary for completion. Nonetheless, all prior results on matrix completion have imposed an assumption like **A1** (i.e., Candès and Recht, 2009; Candès and Tao, 2009; Keshavan et al., 2009), and it would be interesting to see if it can be removed as a requirement, or if it is somehow necessary.

In many matrix completion scenarios of interest in machine learning, the provided entries are corrupted by noise. While Theorem 2 only addresses the noise-free case, we can immediately extend our results to the noisy case. Specifically, suppose we observe $X_{ij} = M_{ij} + \nu_{ij}$ on the set Ω and we are guaranteed that

$$\sum_{(i,j)\in\Omega} \nu_{ij}^2 \leq \delta^2 .$$

Then if we solve the quadratically constrained problem

$$\begin{array}{ll} \text{minimize} & \|\boldsymbol{X}\|_* \\ \text{subject to} & \sum_{(i,j)\in\Omega}(X_{ij} - M_{ij})^2 \leq \delta^2 . \end{array} \tag{11}$$

we will have that any optimal solution, $\hat{\boldsymbol{M}}$ of (11) satisfies

$$\|\hat{\boldsymbol{M}} - \boldsymbol{M}\|_F \leq \left(2 + \sqrt{\frac{48 n_1^2 n_2}{m}}\right) \delta .$$

This claim follows directly from the argument of Candès and Plan (2009). Indeed, the only necessary requirements for such stable recovery is that (6) and (7) hold. Hence, under the sampling assumptions of Theorem 2, low-rank matrices can be approximated from noisy data by solving a quadratically constrained nuclear norm problem.

We conclude by noting that much of the simplicity of the argument presented here arises from an application of new large deviation inequalities for matrices. The noncommutative versions of Chernoff and Bernstein' s Inequalities may be useful throughout machine learning and statistical signal processing, and a fruitful line of inquiry would examine how to apply these tools from quantum information to the study of classical signals and systems.

Acknowledgments

B.R. would like to thank Aram Harrow for introducing him to the operator Chernoff bound and many helpful clarifying conversations, Silvia Gandy for pointing out several typos in the original version of this manuscript, and Yi-Kai Liu, Rob Nowak, Ali Rahimi, and Stephen Wright for many fruitful discussions about this paper.

Appendix A. Operator Chernoff Bounds

In this section, I present a proof of 4 based on foundational results needed from Quantum Information Theory. For completeness, I also provide proofs of Theorems 9 and 10 which were originally proven in Ahlswede and Winter (2002). I have made minor modifications to the original arguments, but the assertions remain the same.

To review, a symmetric matrix $\boldsymbol{A}$ is positive semidefinite if all of its eigenvalues are nonnegative. If $\boldsymbol{A}$ and $\boldsymbol{B}$ are positive semidefinite matrices, $\boldsymbol{A} \preceq \boldsymbol{B}$ means $\boldsymbol{B} - \boldsymbol{A}$ is positive semidefinite. For square matrices $\boldsymbol{A}$, the matrix exponential will be denoted $\exp(\boldsymbol{A})$ and is given by the power series

$$\exp(\boldsymbol{A}) = \sum_{k=0}^{\infty} \frac{\boldsymbol{A}^k}{k!} .$$

The following theorem is a generalization of Markov's inequality originally proven in Ahlswede and Winter (2002). Unlike the original proof, the following argument closely follows the standard proof of the traditional Markov inequality and does not rely on discrete summations.

Theorem 9 (Operator Markov Inequality) *Let $\boldsymbol{X}$ be a random positive semidefinite matrix and $\boldsymbol{A}$ a fixed positive definite matrix. Then*

$$\mathbb{P}[\boldsymbol{X} \not\preceq \boldsymbol{A}] \leq \mathrm{Tr}(\mathbb{E}[\boldsymbol{X}]\boldsymbol{A}^{-1}).$$

Proof Note that if $\boldsymbol{X} \not\preceq \boldsymbol{A}$, then $\boldsymbol{A}^{-1/2}\boldsymbol{X}\boldsymbol{A}^{-1/2} \not\preceq \boldsymbol{I}$, and hence $\|\boldsymbol{A}^{-1/2}\boldsymbol{X}\boldsymbol{A}^{-1/2}\| > 1$. Let $I_{\boldsymbol{X}\not\preceq\boldsymbol{A}}$ denote the indicator of the event $\boldsymbol{X} \not\preceq \boldsymbol{A}$. Then $I_{\boldsymbol{X}\not\preceq\boldsymbol{A}} \leq \mathrm{Tr}(\boldsymbol{A}^{-1/2}\boldsymbol{X}\boldsymbol{A}^{-1/2})$ as the right hand side is always nonnegative, and, if the left hand side equals 1, the trace of the right hand side must exceed the norm of the right hand side which is greater than 1. Thus we have

$$\mathbb{P}[\boldsymbol{X} \not\preceq \boldsymbol{A}] = \mathbb{E}[I_{\boldsymbol{X}\not\preceq\boldsymbol{A}}] \leq \mathbb{E}[\mathrm{Tr}(\boldsymbol{A}^{-1/2}\boldsymbol{X}\boldsymbol{A}^{-1/2})] = \mathrm{Tr}(\mathbb{E}[\boldsymbol{X}]\boldsymbol{A}^{-1}).$$

where the last equality follows from the linearity and cyclic properties of the trace. ■

Next I will derive a noncommutative version of the Chernoff bound. This was also proven in Ahlswede and Winter (2002) for i.i.d. matrices. The version stated here is more general in that the random matrices need not be identically distributed, but the proof is essentially the same.

Theorem 10 (Noncommutative Chernoff Bound) *Let $\boldsymbol{X}_1, \ldots, \boldsymbol{X}_n$ be independent symmetric random matrices in $\mathbb{R}^{d\times d}$. Let $\boldsymbol{A}$ be an arbitrary symmetric matrix. Then for any invertible $d \times d$ matrix $\boldsymbol{T}$*

$$\mathbb{P}\left[\sum_{k=1}^{n} \boldsymbol{X}_k \not\preceq n\boldsymbol{A}\right] \leq d \prod_{k=1}^{n} \|\mathbb{E}[\exp(\boldsymbol{T}\boldsymbol{X}_k\boldsymbol{T}^* - \boldsymbol{T}\boldsymbol{A}\boldsymbol{T}^*)]\|.$$

Proof The proof relies on an estimate of Golden (1965) and Thompson (1965) which is stated here without proof.

Lemma 11 (Golden-Thompson inequality) *For any symmetric matrices $\boldsymbol{A}$ and $\boldsymbol{B}$,*

$$\mathrm{Tr}(\exp(\boldsymbol{A}+\boldsymbol{B})) \leq \mathrm{Tr}((\exp\boldsymbol{A})(\exp\boldsymbol{B})).$$

Much like the proof of the standard Chernoff bound, the theorem now follows from a long chain of inequalities.

$$\begin{aligned}
\mathbb{P}\left[\sum_{k=1}^{n} \boldsymbol{X}_k \not\preceq n\boldsymbol{A}\right] &= \mathbb{P}\left[\sum_{k=1}^{n} (\boldsymbol{X}_k - \boldsymbol{A}) \not\preceq 0\right] \\
&= \mathbb{P}\left[\sum_{k=1}^{n} \boldsymbol{T}(\boldsymbol{X}_k - \boldsymbol{A})\boldsymbol{T}^* \not\preceq 0\right] \\
&= \mathbb{P}\left[\exp\left(\sum_{k=1}^{n} \boldsymbol{T}(\boldsymbol{X}_k - \boldsymbol{A})\boldsymbol{T}^*\right) \not\preceq \boldsymbol{I}_d\right] \\
&\leq \operatorname{Tr}\left(\mathbb{E}\left[\exp\left(\sum_{k=1}^{n} \boldsymbol{T}(\boldsymbol{X}_k - \boldsymbol{A})\boldsymbol{T}^*\right)\right]\right) \\
&= \mathbb{E}\left[\operatorname{Tr}\left(\exp\left(\sum_{k=1}^{n} \boldsymbol{T}(\boldsymbol{X}_k - \boldsymbol{A})\boldsymbol{T}^*\right)\right)\right] \\
&\leq \mathbb{E}\left[\operatorname{Tr}\left(\exp\left(\sum_{k=1}^{n-1} \boldsymbol{T}(\boldsymbol{X}_k - \boldsymbol{A})\boldsymbol{T}^*\right)\exp\left(\boldsymbol{T}(\boldsymbol{X}_n - \boldsymbol{A})\boldsymbol{T}^*\right)\right)\right] \\
&\leq \mathbb{E}_{1,\ldots,n-1}\left[\operatorname{Tr}\left(\exp\left(\sum_{k=1}^{n-1} \boldsymbol{T}(\boldsymbol{X}_k - \boldsymbol{A})\boldsymbol{T}^*\right)\mathbb{E}[\exp\left(\boldsymbol{T}(\boldsymbol{X}_n - \boldsymbol{A})\boldsymbol{T}^*\right)]\right)\right] \\
&\leq \|\mathbb{E}[\exp\left(\boldsymbol{T}(\boldsymbol{X}_n - \boldsymbol{A})\boldsymbol{T}^*\right)]\|\,\mathbb{E}_{1,\ldots,n-1}\left[\operatorname{Tr}\left(\exp\left(\sum_{k=1}^{n-1} \boldsymbol{T}(\boldsymbol{X}_k - \boldsymbol{A})\boldsymbol{T}^*\right)\right)\right] \\
&\leq \prod_{k=2}^{n} \|\mathbb{E}[\exp\left(\boldsymbol{T}(\boldsymbol{X}_k - \boldsymbol{A})\boldsymbol{T}^*\right)]\|\,\mathbb{E}\left[\operatorname{Tr}\left(\exp\left(\boldsymbol{T}(\boldsymbol{X}_1 - \boldsymbol{A})\boldsymbol{T}^*\right)\right)\right] \\
&\leq d\prod_{k=1}^{n} \|\mathbb{E}[\exp\left(\boldsymbol{T}(\boldsymbol{X}_k - \boldsymbol{A})\boldsymbol{T}^*\right)]\| \,.
\end{aligned}$$

Here, the first three lines follow from standard properties of the semidefinite ordering. The fourth line invokes the Operator Markov Inequality. The sixth line follows from the Golden-Thompson inequality. The seventh line follows from independence of the $\boldsymbol{X}_k$. The eighth line follows because for positive definite matrices $\operatorname{Tr}(\boldsymbol{AB}) \leq \operatorname{Tr}(\boldsymbol{A})\|\boldsymbol{B}\|$. This is just another statement of the duality between the nuclear and operator norms. The ninth line iteratively repeats the previous two steps. The final line follows because for a positive definite matrix $\boldsymbol{A}$, $\operatorname{Tr}(\boldsymbol{A})$ is the sum of the eigenvalues of $\boldsymbol{A}$, and all of the eigenvalues are at most $\|\boldsymbol{A}\|$. ■

Let us now turn to proving the Noncommutative Bernstein Inequality presented in Section 3. Gross et al. (2010) proposed a similar inequality for symmetric i.i.d. random matrices with a slightly worse constant. The proof here is more general and follows the standard derivation of Bernstein's inequality.

Proof [of Theorem 4] Set

$$\boldsymbol{Y}_k = \begin{bmatrix} \boldsymbol{0} & \boldsymbol{X}_k \\ \boldsymbol{X}_k^* & \boldsymbol{0} \end{bmatrix}.$$

Then $\boldsymbol{Y}_k$ are symmetric random variables, and for all k

$$\|\mathbb{E}[\boldsymbol{Y}_k^2]\| = \left\|\mathbb{E}\left[\begin{bmatrix} \boldsymbol{X}_k\boldsymbol{X}_k^* & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{X}_k^*\boldsymbol{X}_k \end{bmatrix}\right]\right\| = \max\{\|\mathbb{E}[\boldsymbol{X}_k\boldsymbol{X}_k^*]\|, \|\mathbb{E}[\boldsymbol{X}_k^*\boldsymbol{X}_k]\|\} = \rho_k^2 .$$

Moreover, the maximum singular value of $\sum_{k=1}^L \boldsymbol{X}_k$ is equal to the maximum eigenvalue of $\sum_{k=1}^L \boldsymbol{Y}_k$. By Theorem 10, we have for all $\lambda > 0$

$$\mathbb{P}\left[\left\|\sum_{k=1}^L \boldsymbol{X}_k\right\| > Lt\right] = \mathbb{P}\left[\sum_{k=1}^L \boldsymbol{Y}_k \not\preceq Lt\boldsymbol{I}\right] \leq (d_1+d_2)\exp(-L\lambda t)\prod_{k=1}^L \|\mathbb{E}[\exp(\lambda \boldsymbol{Y}_k)]\| .$$

For each k, let $\boldsymbol{Y}_k = \boldsymbol{U}_k\boldsymbol{\Lambda}_k\boldsymbol{U}_k^*$ be an eigenvalue decomposition, where $\boldsymbol{\Lambda}_k$ is the diagonal matrix of the eigenvalues of $\boldsymbol{Y}_k$. In turn, it follows that for $s > 0$

$$-M^s\boldsymbol{Y}_k^2 \preceq -\boldsymbol{U}_kM^s\boldsymbol{\Lambda}_k^2\boldsymbol{U}_k^* \preceq \boldsymbol{U}_k\boldsymbol{\Lambda}_k^{2+s}\boldsymbol{U}_k^* = \boldsymbol{Y}_k^{2+s} \preceq \boldsymbol{U}_kM^s\boldsymbol{\Lambda}_k^2\boldsymbol{U}_k^* \preceq M^s\boldsymbol{Y}_k^2 ,$$

which then implies

$$\|\mathbb{E}[\boldsymbol{Y}_k^{s+2}]\| \leq M^s\|\mathbb{E}[\boldsymbol{Y}_k^2]\| . \tag{12}$$

For fixed k, we have

$$\begin{aligned}
\|\mathbb{E}[\exp(\lambda\boldsymbol{Y}_k)]\| &\leq \|\boldsymbol{I}\| + \sum_{j=2}^\infty \frac{\lambda^j}{j!}\|\mathbb{E}[\boldsymbol{Y}_k^j]\| \\
&\leq 1 + \sum_{j=2}^\infty \frac{\lambda^j}{j!}\|\mathbb{E}[\boldsymbol{Y}_k^2]\|M^{j-2} \\
&= 1 + \frac{\rho_k^2}{M^2}\sum_{j=2}^\infty \frac{\lambda^j}{j!}M^j = 1 + \frac{\rho_k^2}{M^2}(\exp(\lambda M) - 1 - \lambda M) \\
&\leq \exp\left(\frac{\rho_k^2}{M^2}(\exp(\lambda M) - 1 - \lambda M)\right) .
\end{aligned}$$

The first inequality follows from the triangle inequality and the fact that $\mathbb{E}[\boldsymbol{Y}_k] = 0$, the second inequality follows from (12), and the final inequality follows from the fact that $1 + x \leq \exp(x)$ for all x. Putting this together gives

$$\mathbb{P}\left[\left\|\sum_{k=1}^L \boldsymbol{X}_k\right\| > Lt\right] \leq (d_1+d_2)\exp\left(-\lambda Lt + \frac{\sum_{k=1}^L \rho_k^2}{M^2}(\exp(\lambda M) - 1 - \lambda M)\right) .$$

This final expression is now just a real number, and only has to be minimized as a function of λ. The theorem now follows by algebraic manipulation: the right hand side is minimized by setting $\lambda = \frac{1}{M}\log(1 + \frac{tLM}{\sum_{k=1}^L \rho_k^2})$, then basic approximations can be employed to complete the argument (see, for example, Panchenko, 2007, Lectures 4 and 5). ■

References

Rudolf Ahlswede and Andreas Winter. Strong converse for identification via quantum channels. *IEEE Transactions on Information Theory*, 48(3):569–579, 2002.

Andreas Argyriou, Charles A. Micchelli, and Massimiliano Pontil. Convex multi-task feature learning. *Machine Learning*, 2008. Published online first at `http://www.springerlink.com/`.

Carolyn Beck and Raffaello D'Andrea. Computational study and comparisons of LFT reducibility methods. In *Proceedings of the American Control Conference*, 1998.

Emmanuel Candès and Benjamin Recht. Exact matrix completion via convex optimization. *Foundations of Computational Mathematics*, 9(6):717–772, 2009.

Emmanuel J. Candès and Yaniv Plan. Matrix completion with noise. *Proceedings of the IEEE*, 98 (6):925–936, 2009.

Emmanuel J. Candès and Terence Tao. The power of convex relaxation: Near-optimal matrix completion. *IEEE Transactions on Information Theory*, 56(5):2053–2080, 2009.

Emmanuel J. Candés, Justin Romberg, and Terence Tao. Robust uncertainty principles: Exact signal reconstruction from highly incomplete frequency information. *IEEE Transactions on Information Theory*, 52(2):489–509, 2006. ISSN 0018-9448.

Alexander L. Chistov and Dima Yu. Grigoriev. Complexity of quantifier elimination in the theory of algebraically closed fields. In *Proceedings of the 11th Symposium on Mathematical Foundations of Computer Science*, volume 176 of *Lecture Notes in Computer Science*, pages 17–31. Springer Verlag, 1984.

Victor H. de la Peña and Stephen J. Montgomery-Smith. Decoupling inequalities for the tail probabilities of multivariate U-statistics. *Annals of Probability*, 23(2):806–816, 1995. ISSN 0091-1798.

Maryam Fazel. *Matrix Rank Minimization with Applications*. PhD thesis, Stanford University, 2002.

Maryam Fazel, Haitham Hindi, and Stephen Boyd. A rank minimization heuristic with application to minimum order system approximation. In *Proceedings of the American Control Conference*, 2001.

Jean Jacques Fuchs. On sparse representations in arbitrary redundant bases. *IEEE Transactions on Information Theory*, 50:1341–1344, 2004.

Sideny Golden. Lower bounds for the Helmholtz function. *Physical Review*, 137B(4):B1127–1128, 1965.

Gaston H. Gonnet. Expected length of the longest probe sequence in hash code searching. *Journal of the Association for Computing Machinery*, 28(2):289–304, 1981.

David Gross. Recovering low-rank matrices from few coefficients in any basis. *IEEE Transactions on Information Theory*, 57:1548–1566, 2011.

David Gross, Yi-Kai Liu, Steven T. Flammia, Stephen Becker, and Jens Eisert. Quantum state tomography via compressed sensing. *Physical Review Letters*, 105(15):150401, 2010.

Torben Hagerup and Christine Rüb. A guided tour of Chernoff bounds. *Information Processing Letters*, 33:305–308, 1990.

Raghunandan H. Keshavan, Andrea Montanari, and Sewoong Oh. Matrix completion from a few entries. *IEEE Transactions on Information Theory*, 56(6):2980–2998, 2009.

Yehuda Koren, Robert Bell, and Chris Volinsky. Matrix factorization techniques for recommender systems. *Computer*, 42(8):30–37, 2009.

Nathan Linial, Eran London, and Yuri Rabinovich. The geometry of graphs and some of its algorithmic applications. *Combinatorica*, 15:215–245, 1995.

Mehran Mesbahi and George P. Papavassilopoulos. On the rank minimization problem over a positive semidefinite linear matrix inequality. *IEEE Transactions on Automatic Control*, 42(2):239–243, 1997.

Guillaume Obozinski, Ben Taskar, and Michael I. Jordan. Joint covariate selection and joint subspace selection for multiple classification problems. *Statistics and Computing*, pages 1–22, 2009.

Dmitry Panchenko. MIT 18.465: Statistical Learning Theory. MIT Open Courseware `http://ocw.mit.edu/OcwWeb/Mathematics/18-465Spring-2007/CourseHome/`, 2007.

Benjamin Recht, Maryam Fazel, and Pablo Parrilo. Guaranteed minimum rank solutions of matrix equations via nuclear norm minimization. *SIAM Review*, 52(3):471–501, 2010.

Jason D. M. Rennie and Nathan Srebro. Fast maximum margin matrix factorization for collaborative prediction. In *Proceedings of the International Conference of Machine Learning*, 2005.

Anthony Man-Cho So and Yinyu Ye. Theory of semidefinite programming for sensor network localization. *Mathematical Programming, Series B*, 109:367–384, 2007.

Nathan Srebro. *Learning with Matrix Factorizations*. PhD thesis, Massachusetts Institute of Technology, 2004.

Colin J. Thompson. Inequality with applications in statistical mechanics. *Journal of Mathematical Physics*, 6(11):1812–1823, 1965.

Kilian Q. Weinberger and Lawrence K. Saul. Unsupervised learning of image manifolds by semidefinite programming. *International Journal of Computer Vision*, 70(1):77–90, 2006.

Journal of Machine Learning Research 12 (2011) 3431-3466 Submitted 12/10; Revised 11/11; Published 12/11

Convergence of Distributed Asynchronous Learning Vector Quantization Algorithms

Benoît Patra[*] BENOIT.PATRA@UPMC.FR
LSTA
Université Pierre et Marie Curie – Paris VI
Tour 15-25, 4 place Jussieu
75252 Paris cedex 05, France

Editor: Gabor Lugosi

Abstract

Motivated by the problem of effectively executing clustering algorithms on very large data sets, we address a model for large scale distributed clustering methods. To this end, we briefly recall some standards on the quantization problem and some results on the almost sure convergence of the competitive learning vector quantization (CLVQ) procedure. A general model for linear distributed asynchronous algorithms well adapted to several parallel computing architectures is also discussed. Our approach brings together this scalable model and the CLVQ algorithm, and we call the resulting technique the distributed asynchronous learning vector quantization algorithm (DALVQ). An in-depth analysis of the almost sure convergence of the DALVQ algorithm is performed. A striking result is that we prove that the multiple versions of the quantizers distributed among the processors in the parallel architecture asymptotically reach a consensus almost surely. Furthermore, we also show that these versions converge almost surely towards the same nearly optimal value for the quantization criterion.

Keywords: k-means, vector quantization, distributed, asynchronous, stochastic optimization, scalability, distributed consensus

1. Introduction

Distributed algorithms arise in a wide range of applications, including telecommunications, distributed information processing, scientific computing, real time process control and many others. Parallelization is one of the most promising ways to harness greater computing resources, whereas building faster serial computers is increasingly expensive and also faces some physical limits such as transmission speeds and miniaturization. One of the challenges proposed for machine learning is to build scalable applications that quickly process large amounts of data in sophisticated ways. Building such large scale algorithms attacks several problems in a distributed framework, such as communication delays in the network or numerous problems caused by the lack of shared memory.

Clustering algorithms are one of the primary tools of unsupervised learning. From a practical perspective, clustering plays an outstanding role in data mining applications such as text mining, web analysis, marketing, medical diagnostics, computational biology and many others. Clustering is a separation of data into groups of similar objects. As clustering represents the data with fewer clusters, there is a necessary loss of certain fine details, but simplification is achieved. The popular

∗. Also at LOKAD SAS, 70 rue Lemercier, 75017 Paris, France, email: benoit.patra@lokad.com.

competitive learning vector quantization (CLVQ) algorithm (see Gersho and Gray, 1992) provides a technique for building reliable clusters characterized by their prototypes. As pointed out by Bottou and Bengio (1995), the CLVQ algorithm can also be viewed as the on-line version of the widespread Lloyd's method (see Lloyd 2003, for the definition) which is referred to as batch k-means in Bottou and Bengio (1995). The CLVQ also belongs to the class of stochastic gradient descent algorithms (for more information on stochastic gradient descent procedures we refer the reader to Benveniste et al. 1990).

The analysis of parallel stochastic gradient procedures in a machine learning context has recently received a great deal of attention (see for instance Zinkevich et al. 2009 and McDonald et al. 2010). In the present paper, we go further by introducing a model that brings together the original CLVQ algorithm and the comprehensive theory of asynchronous parallel linear algorithms developed by Tsitsiklis (1984), Tsitsiklis et al. (1986) and Bertsekas and Tsitsiklis (1989). The resulting model will be called distributed asynchronous learning vector quantization (DALVQ for short). At a high level, the DALVQ algorithm parallelizes several executions of the CLVQ method concurrently on different processors while the results of these algorithms are broadcast through the distributed framework asynchronously and efficiently. Here, the term processor refers to any computing instance in a distributed architecture (see Bullo et al. 2009, chap. 1, for more details). Let us remark that there is a series of publications similar in spirit to this paper. Indeed in Frasca et al. (2009) and in Durham et al. (2009), a coverage control problem is formulated as an optimization problem where the functional cost to be minimized is the same of the quantization problem stated in this manuscript.

Let us provide a brief mathematical introduction to the CLVQ technique and DALVQ algorithms. The first technique computes quantization scheme for d dimensional samples $\mathbf{z}_1, \mathbf{z}_2, \ldots$ using the following iterations on a $\left(\mathbb{R}^d\right)^\kappa$ vector,

$$w(t+1) = w(t) - \varepsilon_{t+1} H\left(\mathbf{z}_{t+1}, w(t)\right), \quad t \geq 0.$$

In the equation above, $w(0) \in \left(\mathbb{R}^d\right)^\kappa$ and the ε_t are positive reals. The vector $H(\mathbf{z}, w)$ is the opposite of the difference between the sample $\mathbf{z}$ and its nearest component in w. Assume that there are M computing entities, the data are split among the memory of these machines: $\mathbf{z}_1^i, \mathbf{z}_2^i, \ldots$, where $i \in \{1, \ldots, M\}$. Therefore, the DALVQ algorithms are defined by the M iterations $\{w^i(t)\}_{t=0}^\infty$, called versions, satisfying (with slight simplifications)

$$w^i(t+1) = \sum_{j=1}^{M} a^{i,j}(t) w^j(\tau^{i,j}(t)) - \varepsilon_{t+1}^i H\left(\mathbf{z}_{t+1}^i, w^i(t)\right), \quad i \in \{1, \ldots, M\} \text{ and } t \geq 0.$$

The time instants $\tau^{i,j}(t) \geq 0$ are deterministic but unknown and the delays satisfy the inequality $t - \tau^{i,j}(t) \geq 0$. The families $\{a^{i,j}(t)\}_{j=1}^M$ define the weights of convex combinations.

As a striking result, we prove that multiple versions of the quantizers, distributed among the processors in a parallel architecture, asymptotically reach a consensus almost surely. Using the materials introduced above, it writes

$$w^i(t) - w^j(t) \xrightarrow[t \to \infty]{} 0, \quad (i, j) \in \{1, \ldots, M\}^2, \text{ almost surely (a.s.).}$$

Furthermore, we also show that these versions converge almost surely towards (the same) nearly optimal value for the quantization criterion. These convergence results are similar in spirit to the

most satisfactory almost sure convergence theorem for the CLVQ algorithm obtained by Pagès (1997).

For a given time span, our parallel DALVQ algorithm is able to process much more data than a single processor execution of the CLVQ procedure. Moreover, DALVQ is also asynchronous. This means that local algorithms do not have to wait at preset points for messages to become available. This allows some processors to compute faster and execute more iterations than others, and it also allows communication delays to be substantial and unpredictable. The communication channels are also allowed to deliver messages out of order, that is, in a different order than the one in which they were transmitted. Asynchronism can provide two major advantages. First, a reduction of the synchronization penalty, which could bring a speed advantage over a synchronous execution. Second, for potential industrialization, asynchronism has greater implementation flexibility. Tolerance to system failures and uncertainty can also be increased. As in the case with any on-line algorithm, DALVQ also deals with variable data loads over time. In fact, on-line algorithms avoid tremendous and non scalable batch requests on all data sets. Moreover, with an on-line algorithm, new data may enter the system and be taken into account while the algorithm is already running.

The paper is organized as follows. In Section 2 we review some standard facts on the clustering problem. We extract the relevant material from Pagès (1997) without proof, thus making our exposition self-contained. In Section 3 we give a brief exposition of the mathematical framework for parallel asynchronous gradient methods introduced by Tsitsiklis (1984), Tsitsiklis et al. (1986) and Bertsekas and Tsitsiklis (1989). The results of Blondel et al. (2005) on the asymptotic consensus in asynchronous parallel averaging problems are also recalled. In Section 4, our main results are stated and proved.

2. Quantization and CLVQ Algorithm

In this section, we describe the mathematical quantization problem and the CLVQ algorithm. We also recall some convergence results for this technique found by Pagès (1997).

2.1 Overview

Let μ be a probability measure on $\mathbb{R}^d$ with finite second-order moment. The quantization problem consists in finding a "good approximation" of μ by a set of κ vectors of $\mathbb{R}^d$ called quantizer. Throughout the document the κ quantization points (or prototypes) will be seen as the components of a $\left(\mathbb{R}^d\right)^\kappa$-dimensional vector $w = (w_1, \ldots, w_\kappa)$. To measure the correctness of a quantization scheme given by w, one introduces a cost function called distortion, defined by

$$C_\mu(w) = \frac{1}{2} \int_{\mathbb{R}^d} \min_{1 \leq \ell \leq \kappa} \|\mathbf{z} - w_\ell\|^2 \, d\mu(\mathbf{z}).$$

Under some minimal assumptions, the existence of an optimal $\left(\mathbb{R}^d\right)^\kappa$-valued quantizer vector $w^\circ \in \operatorname{argmin}_{w \in \left(\mathbb{R}^d\right)^\kappa} C_\mu(w)$ has been established by Pollard (1981) (see also Sabin and Gray 1986, Appendix 2).

In a statistical context, the distribution μ is only known through n independent random observations $\mathbf{z}_1, \ldots, \mathbf{z}_n$ drawn according to μ. Denote by μ_n the empirical distribution based on $\mathbf{z}_1, \ldots, \mathbf{z}_n$,

that is, for every Borel subset A of $\mathbb{R}^d$

$$\mu_n(A) = \frac{1}{n}\sum_{i=1}^{n} \mathbb{1}_{\{\mathbf{z}_i \in A\}}.$$

Much attention has been devoted to the convergence study of the quantization scheme provided by the empirical minimizers

$$w_n^\circ \in \underset{w\in\left(\mathbb{R}^d\right)^\kappa}{\operatorname{argmin}} C_{\mu_n}(w).$$

The almost sure convergence of $C_\mu\left(w_n^\circ\right)$ towards $\min_{w\in\left(\mathbb{R}^d\right)^\kappa} C_\mu(w)$ was proved by Pollard (1981, 1982a) and Abaya and Wise (1984). Rates of convergence and nonasymptotic performance bounds have been considered by Pollard (1982b), Chou (1994), Linder et al. (1994), Bartlett et al. (1998), Linder (2001, 2000), Antos (2005) and Antos et al. (2005). Convergence results have been established by Biau et al. (2008) where μ is a measure on a Hilbert space. It turns out that the minimization of the empirical distortion is a computationally hard problem. As shown by Inaba et al. (1994), the computational complexity of this minimization problem is exponential in the number of quantizers κ and the dimension of the data d. Therefore, exact computations are intractable for most of the practical applications.

Based on this, our goal in this document is to investigate effective methods that produce accurate quantizations with data samples. One of the most popular procedure is Lloyd's algorithm (see Lloyd, 2003) sometimes refereed to as batch k-means. A convergence theorem for this algorithm is provided by Sabin and Gray (1986). Another celebrated quantization algorithm is the competitive learning vector quantization (CLVQ), also called on-line k-means. The latter acronym outlines the fact that data arrive over time while the execution of the algorithm and their characteristics are unknown until their arrival times. The main difference between the CLVQ and the Lloyd's algorithm is that the latter run in batch training mode. This means that the whole training set is presented before performing an update, whereas the CLVQ algorithm uses each item of the training sequence at each update.

The CLVQ procedure can be seen as a stochastic gradient descent algorithm. In the more general context of gradient descent methods, one cannot hope for the convergence of the procedure towards global minimizers with a non convex objective function (see for instance Benveniste et al. 1990). In our quantization context, the distortion mapping C_μ is not convex (see for instance Graf and Luschgy 2000). Thus, just as in Lloyd's method, the iterations provided by the CLVQ algorithm converge towards local minima of C_μ.

Assuming that the distribution μ has a compact support and a bounded density with respect to the Lebesgue measure, Pagès (1997) states a result regarding the almost sure consistency of the CLVQ algorithm towards critical points of the distortion C_μ. The author shows that the set of critical points necessarily contains the global and local optimal quantizers. The main difficulties in the proof arise from the fact that the gradient of the distortion is singular on κ-tuples having equal components and the distortion function C_μ is not convex. This explains why standard theories for stochastic gradient algorithm do not apply in this context.

2.2 The Quantization Problem, Basic Properties

In the sequel, we denote by $\mathcal{G}$ the closed convex hull of $\operatorname{supp}(\mu)$, where $\operatorname{supp}(\mu)$ stands for the support of the distribution. Observe that, with this notation, the distortion mapping is the function

$C : \left(\mathbb{R}^d\right)^\kappa \longrightarrow [0,\infty)$ defined by

$$C(w) \triangleq \frac{1}{2}\int_{\mathcal{G}} \min_{1\leq \ell \leq \kappa} \|\mathbf{z}-w_\ell\|^2 \, d\mu(\mathbf{z}), \quad w=(w_1,\ldots,w_\kappa) \in \left(\mathbb{R}^d\right)^\kappa.$$

Throughout the document, with a slight abuse of notation, $\|.\|$ means both the Euclidean norm of $\mathbb{R}^d$ or $\left(\mathbb{R}^d\right)^\kappa$. In addition, the notation $\mathcal{D}_*^\kappa$ stands for the set of all vector of $\left(\mathbb{R}^d\right)^\kappa$ with pairwise distinct components, that is,

$$\mathcal{D}_*^\kappa \triangleq \left\{ w \in \left(\mathbb{R}^d\right)^\kappa \mid w_\ell \neq w_k \text{ if and only if } \ell \neq k \right\}.$$

Under some extra assumptions on μ, the distortion function can be rewritten using space partition set called Voronoï tessellation.

Definition 1 *Let $w \in \left(\mathbb{R}^d\right)^\kappa$, the Voronoï tessellation of $\mathcal{G}$ related to w is the family of open sets $\{W_\ell(w)\}_{1\leq\ell\leq\kappa}$ defined as follows:*

- *If $w \in \mathcal{D}_*^\kappa$, for all $1 \leq \ell \leq \kappa$,*

$$W_\ell(w) = \left\{ v \in \mathcal{G} \;\middle|\; \|w_\ell - v\| < \min_{k\neq\ell} \|w_k - v\| \right\}.$$

- *If $w \in \left(\mathbb{R}^d\right)^\kappa \setminus \mathcal{D}_*^\kappa$, for all $1 \leq \ell \leq \kappa$,*
  - *if $\ell = \min\{k \mid w_k = w_\ell\}$,*

$$W_\ell(w) = \left\{ v \in \mathcal{G} \;\middle|\; \|w_\ell - v\| < \min_{w_k\neq w_\ell} \|w_k - v\| \right\}$$

  - *otherwise, $W_\ell(w) = \emptyset$.*

As an illustration, Figure 1 shows Voronoï tessellations associated to a vector w lying in $([0,1]\times[0,1])^{50}$ whose components have been drawn independently and uniformly. This figure also highlights a remarkable property of the cell borders, which are portions of hyperplanes (see Graf and Luschgy, 2000).

Observe that if $\mu(H)$ is zero for any hyperplane H of $\mathbb{R}^d$ (a property which is sometimes referred to as strong continuity) then using Definition 1, it is easy to see that the distortion takes the form:

$$C(w) = \frac{1}{2}\sum_{\ell=1}^{\kappa} \int_{W_\ell(w)} \|\mathbf{z}-w_\ell\|^2 \, d\mu(\mathbf{z}), \quad w \in \left(\mathbb{R}^d\right)^\kappa.$$

The following assumption will be needed throughout the paper. This assumption is similar to the peak power constraint (see Chou 1994 and Linder 2000). Note that most of the results of this subsection are still valid if μ satisfies the weaker strong continuity property.

Assumption 1 (Compact supported density) *The probability measure μ has a bounded density with respect to the Lebesgue measure on $\mathbb{R}^d$. Moreover, the support of μ is equal to its convex hull $\mathcal{G}$, which in turn, is compact.*

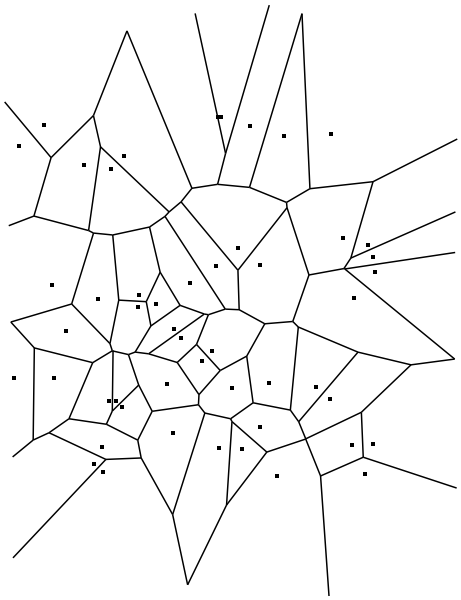


Figure 1: Voronoï tessellation of 50 points of $\mathbb{R}^2$ drawn uniformly in a square.

The next proposition states the differentiability of the distortion C, and provides an explicit formula for the gradient ∇C whenever the distortion is differentiable.

Proposition 1 (Pagès 1997) *Under Assumption 1, the distortion C is continuously differentiable at every $w = (w_1, \ldots, w_\kappa) \in \mathcal{D}_*^\kappa$. Furthermore, for all $1 \leq \ell \leq \kappa$,*

$$\nabla_\ell C(w) = \int_{W_\ell(w)} (w_\ell - \mathbf{z})\, d\mu(\mathbf{z}).$$

Some necessary conditions on the location of the minimizers of C can be derived from its differentiability properties. Therefore, Proposition 2 below states that the minimizers of C have parted components and that they are contained in the support of the density. Thus, the gradient is well defined and these minimizers are necessarily some zeroes of ∇C. For the sequel it is convenient to let $\overset{\circ}{A}$ be the interior of any subset A of $\left(\mathbb{R}^d\right)^\kappa$.

Proposition 2 (Pagès 1997) *Under Assumption 1, we have*

$$\operatorname*{argmin}_{w \in \left(\mathbb{R}^d\right)^\kappa} C(w) \subset \operatorname*{argminloc}_{w \in \mathcal{G}^\kappa} C(w) \subset \overset{\circ}{\mathcal{G}^\kappa} \cap \{\nabla C = 0\} \cap \mathcal{D}_*^\kappa,$$

where $\operatorname{argminloc}_{w \in \mathcal{G}^\kappa} C(w)$ *stands for the set of local minimizers of C over $\mathcal{G}^\kappa$.*

For any $\mathbf{z} \in \mathbb{R}^d$ and $w \in \left(\mathbb{R}^d\right)^\kappa$, let us define the following vector of $\left(\mathbb{R}^d\right)^\kappa$

$$H(\mathbf{z}, w) \triangleq \left((w_\ell - \mathbf{z})\, \mathbb{1}_{\{\mathbf{z} \in W_\ell(w)\}}\right)_{1 \leq \ell \leq \kappa}. \tag{1}$$

On $\mathcal{D}_*^\kappa$, the function H may be interpreted as an observation of the gradient. With this notation, Proposition 1 states that

$$\nabla C(w) = \int_{\mathcal{G}} H(\mathbf{z}, w) d\mu(\mathbf{z}), \quad w \in \mathcal{D}_*^\kappa.$$

Let $\complement A$ stands for the complementary in $\left(\mathbb{R}^d\right)^\kappa$ of a subset $A \subset \left(\mathbb{R}^d\right)^\kappa$. Clearly, for all $w \in \complement \mathcal{D}_*^\kappa$, the mapping $H(.,w)$ is integrable. Therefore, ∇C can be extended on $\left(\mathbb{R}^d\right)^\kappa$ *via* the formula

$$h(w) \triangleq \int_{\mathcal{G}} H(\mathbf{z}, w) d\mu(\mathbf{z}), \quad w \in \left(\mathbb{R}^d\right)^\kappa. \tag{2}$$

Note however that the function h, which is sometimes called the average function of the algorithm, is not continuous.

Remark 1 *Under Assumption 1, a computation for all $w \in \mathcal{D}_*^\kappa$ of the Hessian matrix $\nabla^2 C(w)$ can be deduced from Theorem 4 of (Fort and Pagès, 1995). In fact, the formula established in this theorem is valid for cost functions which are more complex than C (they are associated to Kohonen Self Organizing Maps, see Kohonen 1982 for more details). In Theorem 4, letting $\sigma(k) = \mathbb{1}_{\{k=0\}}$, provides the result for our distortion C. The resulting formula shows that h is singular on $\complement \mathcal{D}_*^\kappa$ and, consequently, that this function cannot be Lipschitz on $\mathcal{G}^\kappa$.*

2.3 Convergence of the CLVQ Algorithm

The problem of finding a reliable clustering scheme for a data set is equivalent to find optimal (or at least nearly optimal) minimizers for the mapping C. A minimization procedure by a usual gradient descent method cannot be implemented as long as ∇C is unknown. Thus, the gradient is approximated by a single example extracted from the data. This leads to the following stochastic gradient descent procedure

$$w(t+1) = w(t) - \varepsilon_{t+1} H\left(\mathbf{z}_{t+1}, w(t)\right), \quad t \geq 0, \tag{3}$$

where $w(0) \in \overset{\circ}{\mathcal{G}}{}^\kappa \cap \mathcal{D}_*^\kappa$ and $\mathbf{z}_1, \mathbf{z}_2 \ldots$ are independent observations distributed according to the probability measure μ.

The algorithm defined by the iterations (3) is known as the CLVQ algorithm in the data analysis community. It is also called the Kohonen Self Organizing Map algorithm with 0 neighbor (see for instance Kohonen 1982) or the on-line k-means procedure (see MacQueen 1967 and Bottou 1998) in various fields related to statistics. As outlined by Pagès in Pagès (1997), this algorithm belongs to the class of stochastic gradient descent methods. However, the almost sure convergence of this type of algorithm cannot be obtained by general tools such as Robbins-Monro method (see Robbins and Monro, 1951) or the Kushner-Clark's Theorem (see Kushner and Clark, 1978). Indeed, the main difficulty essentially arises from the non convexity of the function C, its non coercivity and the singularity of h at $\complement \mathcal{D}_*^\kappa$ (we refer the reader to Pagès 1997, Section 6, for more details).

The following assumption set is standard in a gradient descent context. It basically upraises constraints on the decreasing speed of the sequence of steps $\{\varepsilon_t\}_{t=0}^\infty$.

Assumption 2 (Decreasing steps) *The $(0,1)$-valued sequence $\{\varepsilon_t\}_{t=0}^\infty$ satisfies the following two constraints:*

1. $\sum_{t=0}^{\infty} \varepsilon_t = \infty$.

2. $\sum_{t=0}^{\infty} \varepsilon_t^2 < \infty$.

An examination of identities (3) and (1) reveals that if $\mathbf{z}_{t+1} \in W_{\ell_0}(w(t))$, where the integer $\ell_0 \in \{1,\ldots,M\}$ then

$$w_{\ell_0}(t+1) = (1-\varepsilon_{t+1})\, w_{\ell_0}(t) + \varepsilon_{t+1}\mathbf{z}_{t+1}.$$

The component $w_{\ell_0}(t+1)$ can be viewed as the image of $w_{\ell_0}(t)$ by a $\mathbf{z}_{t+1}$-centered homothety with ratio $1-\varepsilon_{t+1}$ (Figure 2 provides an illustration of this fact). Thus, under Assumptions 1 and 2, the trajectories of $\{w(t)\}_{t=0}^{\infty}$ stay in $\overset{\circ}{\mathcal{G}}{}^{\kappa} \cap \mathcal{D}_*^{\kappa}$. More precisely, if

$$w(0) \in \overset{\circ}{\mathcal{G}}{}^{\kappa} \cap \mathcal{D}_*^{\kappa}$$

then

$$w(t) \in \overset{\circ}{\mathcal{G}}{}^{\kappa} \cap \mathcal{D}_*^{\kappa}, \quad t \geq 0, \text{a.s.}$$

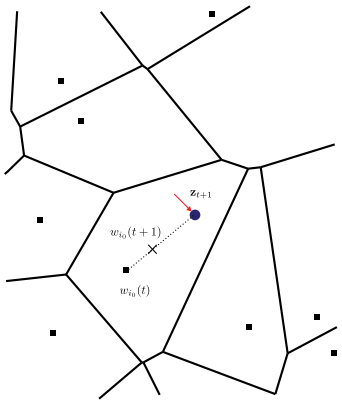


Figure 2: Drawing of a portion of a 2-dimensional Voronoï tessellation. For $t \geq 0$, if the vector $\mathbf{z}_{t+1} \in W_{\ell_0}(w(t))$ then $w_\ell(t+1) = w_\ell(t)$ for all $\ell \neq \ell_0$ and $w_{\ell_0}(t+1)$ lies in the segment $[w_{\ell_0}(t), \mathbf{z}_{t+1}]$. The update of the vector $w_{\ell_0}(t)$ can also be viewed as a $\mathbf{z}_{t+1}$-centered homothety with ratio $1-\varepsilon_{t+1}$.

Although ∇C is not continuous some regularity can be obtained. To this end, we need to introduce the following materials. For any $\delta > 0$ and any compact set $L \subset \mathbb{R}^d$, let the compact set $L_\delta^{\kappa} \subset \left(\mathbb{R}^d\right)^{\kappa}$ be defined as

$$L_\delta^{\kappa} \triangleq \left\{ w \in L^{\kappa} \mid \min_{k \neq \ell} \|w_\ell - w_k\| \geq \delta \right\}. \tag{4}$$

The next lemma that states on the regularity of ∇C will prove to be extremely useful in the proof of Theorem 4 and throughout Section 4.

Lemma 2 (Pagès 1997) *Assume that μ satisfies Assumption 1 and let L be a compact set of $\mathbb{R}^d$. Then, there is some constant P_δ such that for all w and v in L_δ^κ with $[w,v] \subset \mathcal{D}_*^\kappa$,*

$$\|\nabla C(w) - \nabla C(v)\| \leq P_\delta \|w - v\|.$$

The following lemma, called G-lemma in Pagès (1997) is an easy-to-apply convergence results on stochastic algorithms. It is particularly adapted to the present situation of the CLVQ algorithm where the average function of the algorithm h is singular.

Theorem 3 (G-lemma, Fort and Pagès 1996) *Assume that the iterations (3) of the CLVQ algorithm satisfy the following conditions:*

1. $\sum_{t=1}^{\infty} \varepsilon_t = \infty$ *and* $\varepsilon_t \xrightarrow[t\to\infty]{} 0$.
2. *The sequences* $\{w(t)\}_{t=0}^{\infty}$ *and* $\{h(w(t))\}_{t=0}^{\infty}$ *are bounded a.s.*
3. *The series* $\sum_{t=0}^{\infty} \varepsilon_{t+1} (H(\mathbf{z}_{t+1}, w(t)) - h(w(t)))$ *converge a.s. in* $\left(\mathbb{R}^d\right)^\kappa$.
4. *There exists a lower semi-continuous function* $G : \left(\mathbb{R}^d\right)^\kappa \longrightarrow [0,\infty)$ *such that*

$$\sum_{t=0}^{\infty} \varepsilon_{t+1} G(w(t)) < \infty, \quad a.s.$$

Then, there exists a random connected component Ξ of $\{G = 0\}$ such that

$$\operatorname{dist}(w(t), \Xi) \xrightarrow[t\to\infty]{} 0, \quad a.s.,$$

where the symbol dist *denotes the usual distance function between a vector and a subset of* $\left(\mathbb{R}^d\right)^\kappa$. *Note also that if the connected components of $\{G = 0\}$ are singletons then there exists $\xi \in \{G = 0\}$ such that $w(t) \xrightarrow[t\to\infty]{} \xi$ a.s.*

For a definition of the topological concept of connected component, we refer the reader to Choquet (1966). The interest of the G-lemma depends upon the choice of G. In our context, a suitable lower semi-continuous function is $\widehat{G}$ defined by

$$\widehat{G}(w) \triangleq \liminf_{v \in \mathcal{G}^\kappa \cap \mathcal{D}_*^\kappa,\, v \to w} \|\nabla C(v)\|^2, \qquad w \in \mathcal{G}^\kappa. \tag{5}$$

The next theorem is, as far as we know, the first almost sure convergence theorem for the stochastic algorithm CLVQ.

Theorem 4 (Pagès 1997) *Under Assumptions 1 and 2, conditioned on the event*

$$\left\{\liminf_{t\to\infty} \operatorname{dist}\left(w(t), \complement \mathcal{D}_*^\kappa\right) > 0\right\}, \quad \textit{one has}$$

$$\operatorname{dist}(w(t), \Xi_\infty) \xrightarrow[t\to\infty]{} 0, \quad a.s.,$$

where Ξ_∞ is some random connected component of $\{\nabla C = 0\}$.

The proof is an application of the above G-lemma with the mapping $\widehat{G}$ defined by Equation (5). Theorem 4 states that the iterations of the CLVQ necessarily converge towards some critical points (zeroes of ∇C). From Proposition 2 we deduce that the set of critical points necessarily contains optimal quantizers. Recall that without more assumption than $w(0) \in \overset{\circ}{\mathcal{G}}{}^{\kappa} \cap \mathcal{D}_*^{\kappa}$, we have already discussed the fact that the components of $w(t)$ are almost surely parted for all $t \geq 0$. Thus, it is easily seen that the two following events only differ on a set of zero probability

$$\left\{ \liminf_{t\to\infty} \operatorname{dist}\left(w(t), \complement \mathcal{D}_*^{\kappa}\right) > 0 \right\}$$

and

$$\left\{ \inf_{t\geq 0} \operatorname{dist}\left(w(t), \complement \mathcal{D}_*^{\kappa}\right) > 0 \right\}.$$

Some results are provided by Pagès (1997) for asymptotically stuck components but, as pointed out by the author, they are less satisfactory.

3. General Distributed Asynchronous Algorithm

We present in this section some materials and results of the asynchronous parallel linear algorithms theory.

3.1 Model Description

Let $s(t)$ be any $\left(\mathbb{R}^d\right)^{\kappa}$-valued vector and consider the following iterations

$$w(t+1) = w(t) + s(t), \quad t \geq 0. \tag{6}$$

Here, the model of discrete time described by iterations (6) can only be performed by a single computing entity. Therefore, if the computations of the vectors $s(t)$ are relatively time consuming then not many iterations can be achieved for a given time span. Consequently, a parallelization of this computing scheme should be investigated. The aim of this section is to discuss a precise mathematical description of a distributed asynchronous model for the iterations (6). This model for distributed computing was originally proposed by Tsitsiklis et al. (1986) and was revisited in Bertsekas and Tsitsiklis (1989, Section 7.7).

Assume that we dispose of a distributed architecture with M computing entities called processors (or agents, see for instance Bullo et al. 2009). Each processor is labeled, for simplicity of notation, by a natural number $i \in \{1,\ldots,M\}$. Throughout the paper, we will add the superscript i on the variables possessed by the processor i. In the model we have in mind, each processor has a buffer where its current version of the iterated vector is kept, that is, local memory. Thus, for agent i such iterations are represented by the $\left(\mathbb{R}^d\right)^{\kappa}$-valued sequence $\left\{w^i(t)\right\}_{t=0}^{\infty}$.

Let $t \geq 0$ denote the current time. For any pair of processors $(i,j) \in \{1,\ldots,M\}^2$, the value kept by agent j and available for agent i at time t is not necessarily the most recent one, $w^j(t)$, but more probably and outdated one, $w^j(\tau^{i,j}(t))$, where the deterministic time instant $\tau^{i,j}(t)$ satisfy $0 \leq \tau^{i,j}(t) \leq t$. Thus, the difference $t - \tau^{i,j}(t)$ can be seen as a communication delay. This is a modeling of some aspects of the network: latency and bandwidth finiteness.

We insist on the fact that there is a distinction between "global" and "local" time. The time variable we refer above to as t corresponds to a global clock. Such a global clock is needed only for

analysis purposes. The processors work without knowledge of this global clock. They have access to a local clock or to no clock at all.

The algorithm is initialized at $t = 0$, where each processor $i \in \{1, \ldots, M\}$ has an initial version $w^i(0) \in \left(\mathbb{R}^d\right)^\kappa$ in its buffer. We define the general distributed asynchronous algorithm by the following iterations

$$w^i(t+1) = \sum_{j=1}^{M} a^{i,j}(t) w^j(\tau^{i,j}(t)) + s^i(t), \quad i \in \{1, \ldots, M\} \text{ and } t \geq 0. \tag{7}$$

The model can be interpreted as follows: at time $t \geq 0$, processor i receives messages from other processors containing $w^j(\tau^{i,j}(t))$. Processor i incorporates these new vectors by forming a convex combination and incorporates the vector $s^i(t)$ resulting from its own "local" computations. The coefficients $a^{i,j}(t)$ are nonnegative numbers which satisfy the constraint

$$\sum_{j=1}^{M} a^{i,j}(t) = 1, \quad i \in \{1, \ldots, M\} \text{ and } t \geq 0. \tag{8}$$

As the combining coefficients $a^{i,j}(t)$ depend on t, the network communication topology is sometimes referred to as time-varying. The sequences $\left\{\tau^{i,j}(t)\right\}_{t=0}^{\infty}$ need not to be known in advance by any processor. In fact, their knowledge is not required to execute iterations defined by Equation (7). Thus, we do not necessary dispose of a shared global clock or synchronized local clocks at the processors.

As for now the descent terms $\left\{s^i(t)\right\}_{t=0}^{\infty}$ will be arbitrary $\left(\mathbb{R}^d\right)^\kappa$-valued sequences. In Section 4, when we define the distributed asynchronous learning vector quantization (DALVQ), the definition of the descent terms will be made more explicit.

3.2 The Agreement Algorithm

This subsection is devoted to a short survey of the results, found by Blondel et al. (2005), for a natural simplification of the general distributed asynchronous algorithm (7). This simplification is called agreement algorithm by Blondel et al. and is defined by

$$x^i(t+1) = \sum_{j=1}^{M} a^{i,j}(t) x^j(\tau^{i,j}(t)), \quad i \in \{1, \ldots, M\} \text{ and } t \geq 0. \tag{9}$$

where $x^i(0) \in \left(\mathbb{R}^d\right)^\kappa$. An observation of these equations reveals that they are similar to iterations (7), the only difference being that all descent terms equal 0.

In order to analyse the convergence of the agreement algorithm (9), Blondel et al. (2005) define two sets of assumptions that enforce some weak properties on the communication delays and the network topology. As shown in Blondel et al. (2005), if the assumptions contained in one of these two set hold, then the distributed versions of the agreement algorithm, namely the x^i, reach an asymptotical consensus. This latter statement means that there exists a vector $x^\star$ (independent of i) such that

$$x^i(t) \xrightarrow[t \to \infty]{} x^\star, \quad i \in \{1, \ldots, M\}.$$

The agreement algorithm (9) is essentially driven by the communication times $\tau^{i,j}(t)$ assumed to be deterministic but do not need to be known *a priori* by the processors. The following Assumption

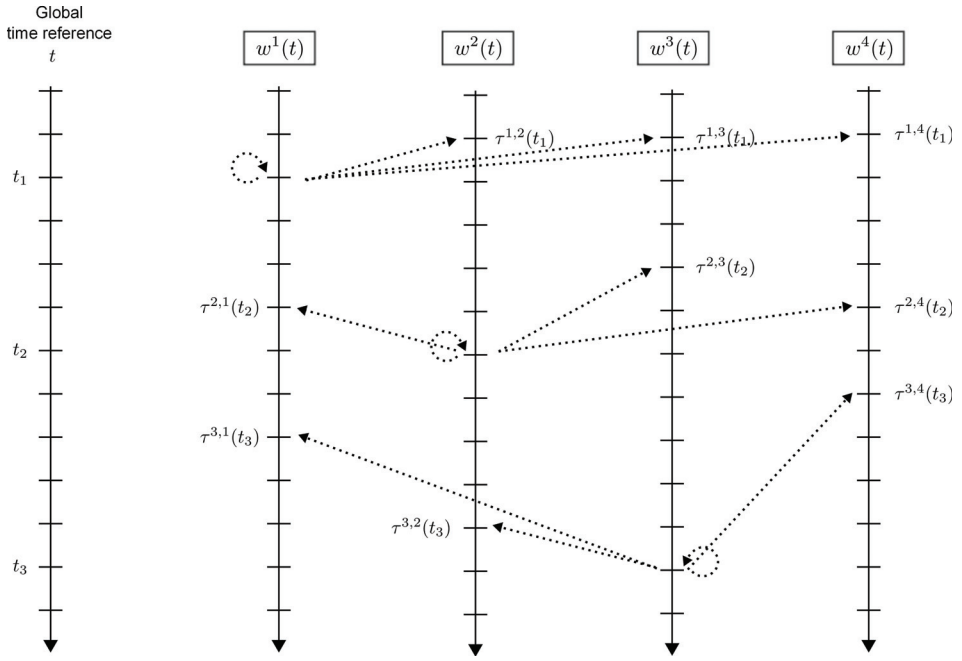


Figure 3: Illustration of the time delays introduced in the general distributed asynchronous algorithm. Here, there are $M = 4$ different processors with their own computations of the vectors $w^{(i)}, i \in \{1,2,3,4\}$. Three arbitrary values of the global time t are represented (t_1, t_2 and t_3), with $\tau^{i,i}(t_k) = t_k$ for all $i \in \{1,2,3,4\}$ and $1 \leq k \leq 3$. The dashed arrows head towards the versions available at time t_k for an agent $i \in \{1,2,3,4\}$ represented by the tail of the arrow.

3 essentially ensures, in its third statement, that the communication delays $t - \tau^{i,j}(t)$ are bounded. This assumption prevents some processor from taking into account some arbitrarily old values computed by others processors. Assumption 3 1. is just a convention: when $a^{i,j}(t) = 0$ the value $\tau^{i,j}(t)$ has no effect on the update. Assumption 3 2. is rather natural because processors have access to their own most recent value.

Assumption 3 (Bounded communication delays)

1. *If* $a^{i,j}(t) = 0$ *then one has* $\tau^{i,j}(t) = t, \quad (i,j) \in \{1,\ldots,M\}^2$ *and* $t \geq 0$,
2. $\tau^{i,i}(t) = t, \quad i \in \{1,\ldots,M\}$ *and* $t \geq 0$.
3. *There exists a positive integer* B_1 *such that*

$$t - B_1 < \tau^{i,j}(t) \leq t, \quad (i,j) \in \{1,\ldots,M\}^2 \text{ and } t \geq 0.$$

The next Assumption 4 states that the value possessed by agent i at time $t+1$, namely $x^i(t+1)$, is a weighted average of its own value and the values that it has just received from other agents.

Assumption 4 (Convex combination and threshold) *There exists a positive constant* $\alpha > 0$ *such that the following three properties hold:*

1. $a^{i,i}(t) \geq \alpha, \quad i \in \{1,\ldots,M\}$ *and* $t \geq 0$.
2. $a^{i,j}(t) \in \{0\} \cup [\alpha, 1], \quad (i,j) \in \{1,\ldots,M\}^2$ *and* $t \geq 0$.
3. $\sum_{j=1}^{M} a^{i,j}(t) = 1, \quad i \in \{1,\ldots,M\}$ *and* $t \geq 0$.

Let us mention one particular relevant case for the choice of the combining coefficients $a^{i,j}(t)$. Let $i \in \{1,\ldots,M\}$ and $t \geq 0$, the set

$$N^i(t) \triangleq \left\{ j \in \{1,\ldots,M\} \in \{1,\ldots,M\} \mid a^{i,j}(t) \neq 0 \right\}$$

corresponds to the set of agents whose version is taken into account by processor i at time t. For all $(i,j) \in \{1,\ldots,M\}^2$ and $t \geq 0$, the weights $a^{i,j}(t)$ are defined by

$$a^{i,j}(t) = \begin{cases} 1/\#N^i(t) & \text{if } j \in N^i(t); \\ 0 & \text{otherwise;} \end{cases}$$

where $\#A$ denotes the cardinal of any finite set A. The above definition on the combining coefficients appears to be relevant for practical implementations of the model DALVQ introduced in Section 4. For a discussion on others special interest cases regarding the choices of the coefficients $a^{i,j}(t)$ we refer the reader to Blondel et al. (2005).

The communication patterns, sometimes refereed to as the network communication topology, can be expressed in terms of directed graph. For a thorough introduction to graph theory, (see Jungnickel, 1999).

Definition 5 (Communication graph) *Let us fix* $t \geq 0$*, the communication graph at time* t*,* $(\mathcal{V}, E(t))$*, is defined by*

- *the set of vertices* $\mathcal{V}$ *is formed by the set of processors* $\mathcal{V} = \{1,\ldots,M\}$,
- *the set of edges* $E(t)$ *is defined via the relationship*

$$(j,i) \in E(t) \text{ if and only if } a^{i,j}(t) > 0.$$

Assumption 5 is a minimal condition required for a consensus among the processors. More precisely, it states that for any pair of agents $(i,j) \in \{1,\ldots,M\}^2$ there is a sequence of communications where the values computed by agent i will influence (directly or indirectly) the future values kept by agent j.

Assumption 5 (Graph connectivity) *The graph* $(\mathcal{V}, \cup_{s \geq t} E(s))$ *is strongly connected for all* $t \geq 0$.

Finally, we define two supplementary assumptions. The combination of one of the two following assumptions with the three previous ones will ensure the convergence of the agreement algorithm. As mentioned above, if Assumption 5 holds then there is a communication path between any pair of agents. Assumption 6 below expresses the fact that there is a finite upper bound for the length of such paths.

Assumption 6 (Bounded communication intervals) *If i communicates with j an infinite number of times then there is a positive integer B_2 such that*

$$(i,j) \in E(t) \cup E(t+1) \cup \ldots \cup E(t+B_2-1), \quad t \geq 0.$$

Assumption 7 is a symmetry condition: if agent $i \in \{1,\ldots,M\}$ communicates with agent $j \in \{1,\ldots,M\}$ then j has communicated or will communicate with i during the time interval $(t-B_3, t+B_3)$ where $B_3 > 0$.

Assumption 7 (Symmetry) *There exists some $B_3 > 0$ such that whenever the pair $(i,j) \in E(t)$, there exists some τ that satisfies $|t-\tau| < B_3$ and $(j,i) \in E(\tau)$.*

To shorten the notation, we set

$$(\mathbf{AsY})_\mathbf{1} \equiv \begin{cases} \text{Assumption 3;} \\ \text{Assumption 4;} \\ \text{Assumption 5;} \\ \text{Assumption 6.} \end{cases} \qquad (\mathbf{AsY})_\mathbf{2} \equiv \begin{cases} \text{Assumption 3;} \\ \text{Assumption 4;} \\ \text{Assumption 5;} \\ \text{Assumption 7;} \end{cases}$$

We are now in a position to state the main result of this section. The Theorem 6 expresses the fact that, for the agreement algorithm, a consensus is asymptotically reached by the agents.

Theorem 6 (Blondel et al. 2005) *Under the set of Assumptions $(\mathbf{AsY})_\mathbf{1}$ or $(\mathbf{AsY})_\mathbf{2}$, there is a consensus vector $x^\star \in \left(\mathbb{R}^d\right)^\kappa$ (independent of i) such that*

$$\lim_{t\to\infty} \left\| x^i(t) - x^\star \right\| = 0, \quad i \in \{1,\ldots,M\}.$$

Besides, there exist $\rho \in [0,1)$ and $L > 0$ such that

$$\left\| x^i(t) - x^i(\tau) \right\| \leq L\rho^{t-\tau}, \quad i \in \{1,\ldots,M\} \text{ and } t \geq \tau \geq 0.$$

3.3 Asymptotic Consensus

This subsection is devoted to the analysis of the general distributed asynchronous algorithm (7). For this purpose, the study of the agreement algorithm defined by Equations (9) will be extremely fruitful. The following lemma states that the version possessed by agent $i \in \{1,\ldots,M\}$ at time $t \geq 0$, namely $w^i(t)$, depends linearly on the others initialization vectors $w^j(0)$ and the descent subsequences $\left\{s^j(\tau)\right\}_{\tau=-1}^{t-1}$, where $j \in \{1,\ldots,M\}$.

Lemma 7 (Tsitsiklis 1984) *For all $(i,j) \in \{1,\ldots,M\}^2$ and $t \geq 0$, there exists a real-valued sequence $\left\{\phi^{i,j}(t,\tau)\right\}_{\tau=-1}^{t-1}$ such that*

$$w^i(t) = \sum_{j=1}^{M} \phi^{i,j}(t,-1)\, w^j(0) + \sum_{\tau=0}^{t-1} \sum_{j=1}^{M} \phi^{i,j}(t,\tau)\, s^j(\tau).$$

For all $(i,j) \in \{1,\ldots,M\}^2$ and $t \geq 0$, the real-valued sequences $\left\{\phi^{i,j}(t,\tau)\right\}_{\tau=-1}^{t-1}$ do not depend on the value taken by the descent terms $s^i(t)$. The real numbers $\phi^{i,j}(t,\tau)$ are determined by the sequences $\left\{\tau^{i,j}(\tau)\right\}_{\tau=0}^{t}$ and $\left\{a^{i,j}(\tau)\right\}_{\tau=0}^{t}$ which do not depend on w. These last sequences are unknown in general, but some useful qualitative properties can be derived, as expressed in Lemma 8 below.

Lemma 8 (Tsitsiklis 1984) *For all $(i,j) \in \{1,\ldots,M\}^2$, let $\left\{\phi^{i,j}(t,\tau)\right\}_{\tau=-1}^{t-1}$ be the sequences defined in Lemma 7.*

1. *Under Assumption 4,*

$$0 \leq \phi^{i,j}(t,\tau) \leq 1, \quad (i,j) \in \{1,\ldots,M\}^2 \text{ and } t > \tau \geq -1.$$

2. *Under Assumptions $(\mathbf{AsY})_\mathbf{1}$ or $(\mathbf{AsY})_\mathbf{2}$, we have:*

   (a) *For all $(i,j) \in \{1,\ldots,M\}^2$ and $\tau \geq -1$, the limit of $\phi^{i,j}(t,\tau)$ as t tends to infinity exists and is independent of j. It will be denoted $\phi^i(\tau)$.*

   (b) *There exists some $\eta > 0$ such that*

$$\phi^i(\tau) > \eta, \quad i \in \{1,\ldots,M\} \text{ and } \tau \geq -1.$$

   (c) *There exist a constant $A > 0$ and $\rho \in (0,1)$ such that*

$$\left|\phi^{i,j}(t,\tau) - \phi^i(\tau)\right| \leq A\rho^{t-\tau}, \quad (i,j) \in \{1,\ldots,M\}^2 \text{ and } t > \tau \geq -1.$$

Take $t' \geq 0$ and assume that the agents stop performing update after time t', but keep communicating and merging the results. This means that $s^j(t) = 0$ for all $t \geq t'$. Then, Equations (7) write

$$w^i(t+1) = \sum_{j=1}^{M} a^{i,j}(t) w^j\left(\tau^{i,j}(t)\right), \quad i \in \{1,\ldots,M\} \text{ and } t \geq t'.$$

If Assumptions $(\mathbf{AsY})_\mathbf{1}$ or $(\mathbf{AsY})_\mathbf{2}$ are satisfied then Theorem 6 shows that there is a consensus vector, depending on the time instant t'. This vector will be equal to $w^\star(t')$ defined below (see Figure 4). Lemma 8 provides a good way to define the sequence $\{w^\star(t)\}_{t=0}^{\infty}$ as shown in Definition 9. Note that this definition does not involve any assumption on the descent terms.

Definition 9 (Agreement vector) *Assume that Assumptions $(\mathbf{AsY})_\mathbf{1}$ or $(\mathbf{AsY})_\mathbf{2}$ are satisfied. The agreement vector sequence $\{w^\star(t)\}_{t=0}^{\infty}$ is defined by*

$$w^\star(t) \triangleq \sum_{j=1}^{M} \phi^j(-1)\, w^j(0) + \sum_{\tau=0}^{t-1} \sum_{j=1}^{M} \phi^j(\tau)\, s^j(\tau), \quad t \geq 0.$$

It is noteworthy that the agreement vector sequence $w^\star$ satisfies the following recursion formula

$$w^\star(t+1) = w^\star(t) + \sum_{j=1}^{M} \phi^j(t) s^j(t), \quad t \geq 0. \tag{10}$$

4. Distributed Asynchronous Learning Vector Quantization

This section is devoted to the distributed asynchronous learning vector quantization techniques. We provide a definition and investigate the almost sure convergence properties of the techniques.

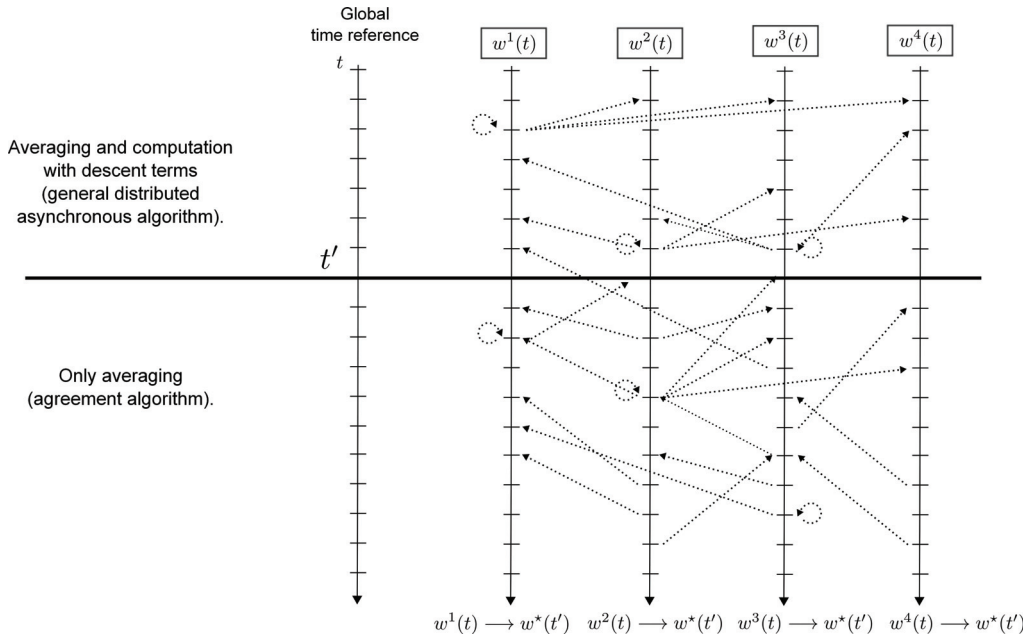


Figure 4: The agreement vector at time t', $w^\star(t')$ corresponds to the common value asymptotically achieved by all processors if computations integrating descent terms have stopped after t', that is, $s^j(t) = 0$ for all $t \geq t'$.

4.1 Introduction, Model Presentation

From now on, and until the end of the paper, we assume that one of the two set of Assumptions $(\mathbf{AsY})_\mathbf{1}$ or $(\mathbf{AsY})_\mathbf{2}$ holds, as well as the compact-supported density Assumption 1. In addition, we will also assume that $0 \in \mathcal{G}$. For the sake of clarity, all the proofs of the main theorems as well as the lemmas needed for these proofs have been postponed at the end of the paper, in Annex.

Tsitsiklis (1984), Tsitsiklis et al. (1986) and Bertsekas and Tsitsiklis (1989) studied distributed asynchronous stochastic gradient optimization algorithms. In this series of publications, for the distributed minimization of a cost function $F : \left(\mathbb{R}^d\right)^\kappa \longrightarrow \mathbb{R}$, the authors considered the general distributed asynchronous algorithm defined by Equation (7) with specific choices for stochastic descent terms s^i. Using the notation of Section 3, the algorithm writes

$$w^i(t+1) = \sum_{j=1}^{M} a^{i,j}(t) w^j(\tau^{i,j}(t)) + s^i(t), \quad i \in \{1,\ldots,M\} \text{ and } t \geq 0,$$

with stochastic descent terms $s^i(t)$ satisfying

$$\mathbb{E}\left\{ s^i(t) \,\middle|\, s^j(\tau),\ j \in \{1,\ldots,M\} \text{ and } t > \tau \geq 0 \right\} = -\varepsilon_{t+1}^i \nabla F\left(w^i(t)\right), \quad i \in \{1,\ldots,M\} \text{ and } t \geq 0. \tag{11}$$

where $\left\{\varepsilon_t^i\right\}_{t=0}^\infty$ are decreasing steps sequences. The definition of the descent terms in Bertsekas and Tsitsiklis (1989) and Tsitsiklis et al. (1986) is more general than the one appearing in Equation

(11). We refer the reader to Assumption 3.2 and 3.3 in Tsitsiklis et al. (1986) and Assumption 8.2 in Bertsekas and Tsitsiklis (1989) for the precise definition of the descent terms. As discussed in Section 2, the CLVQ algorithm is also a stochastic gradient descent procedure. Unfortunately, the results from Tsitisklis et al. do not apply with our distortion function, C, since the authors assume that F is continuously differentiable and ∇F is Lipschitz. Therefore, the aim of this section is to extend the results of Tsitsiklis et al. to the context of vector quantization and on-line clustering.

We first introduce the distributed asynchronous learning vector quantization (DALVQ) algorithm. To prove its almost sure consistency, we will need an asynchronous G-lemma, which is inspired from the G-lemma, Theorem 3, presented in Section 2. This theorem may be seen as an easy-to-apply tool for the almost sure consistency of a distributed asynchronous system where the average function is not necessary regular. Our approach sheds also some new light on the convergence of distributed asynchronous stochastic gradient descent algorithms. Precisely, Proposition 8.1 in Tsitsiklis et al. (1986) claims that the next asymptotic equality holds: $\liminf_{t\to\infty} \left\|\nabla F(w^i(t))\right\| = 0$, while our main Theorem 12 below states that $\lim_{t\to\infty} \left\|\nabla C(w^i(t))\right\| = 0$. However, there is a price to pay for this more precise result with the non Lipschitz gradient ∇C. Similarly to Pagès (1997), who assumes that the trajectory of the CLVQ algorithm has almost surely asymptotically parted components (see Theorem 4 in Section 2), we will suppose that the agreement vector sequence has, almost surely, asymptotically parted component trajectories.

Recall that the goal of the DALVQ is to provide a well designed distributed algorithm that processes quickly (in term of wall clock time) very large data sets to produce accurate quantization. The data sets (or streams of data) are distributed among several queues sending data to the different processors of our distributed framework. Thus, in this context the sequence $\mathbf{z}_1^i, \mathbf{z}_2^i, \ldots$ stands for the data available for processor, where $i \in \{1, \ldots, M\}$. The random variables

$$\mathbf{z}_1^1, \mathbf{z}_2^1, \ldots, \mathbf{z}_1^2, \mathbf{z}_2^2, \ldots$$

are assumed to be independent and identically distributed according to μ.

In the definition of the CLVQ procedure (3), the term $H(\mathbf{z}_{t+1}, w(t))$ can be seen as an observation of the gradient $\nabla C(w(t))$. Therefore, in our DALVQ algorithm, each processor $i \in \{1, \ldots, M\}$ is able to compute such observations using its own data $\mathbf{z}_1^i, \mathbf{z}_2^i, \ldots$. Thus, the DALVQ procedure is defined by Equation (7) with the following choice for the descent term s^i:

$$s^i(t) = \begin{cases} -\varepsilon_{t+1}^i H\left(\mathbf{z}_{t+1}^i, w^i(t)\right) & \text{if } t \in T^i; \\ 0 & \text{otherwise;} \end{cases} \tag{12}$$

where $\left\{\varepsilon_t^i\right\}_{t=0}^{\infty}$ are $(0,1)$-valued sequences. The sets T^i contain the time instants where the version w^i, kept by processor i, is updated with the descent terms. This fine grain description of the algorithm allows some processors to be idle for computing descent terms (when $t \notin T^i$). This reflects the fact that the computing operations might not take the same time for all processors, which is precisely the core of asynchronous algorithms analysis. Similarly to time delays and combining coefficients, the sets T^i are supposed to be deterministic but do not need to be known *a priori* for the execution of the algorithm.

In the DALVQ model, randomness arises from the data $\mathbf{z}$. Therefore, it is natural to let $\{\mathcal{F}_t\}_{t=0}^{\infty}$ be the filtration built on the σ-algebras

$$\mathcal{F}_t \triangleq \sigma\left(\mathbf{z}_s^i,\ i \in \{1, \ldots, M\} \text{ and } t \geq s \geq 0\right), \quad t \geq 0.$$

An easy verification shows that, for all $j \in \{1,\ldots,M\}$ and $t \geq 0$, $w^\star(t)$ and $w^j(t)$ are $\mathcal{F}_t$-measurable random variables.

For simplicity, the assumption on the decreasing speed of the sequences $\left\{\varepsilon_t^i\right\}_{t=0}^\infty$ is strengthened as follows. The notation $a \vee b$ stands for the maximum of two reals a and b.

Assumption 8 *There exist two real numbers $K_1 > 0$ and $K_2 \geq 1$ such that*

$$\frac{K_1}{t \vee 1} \leq \varepsilon_{t+1}^i \leq \frac{K_2}{t \vee 1}, \quad i \in \{1,\ldots,M\} \text{ and } t \geq 0.$$

If Assumption 8 holds then the sequences $\left\{\varepsilon_t^i\right\}_{t=0}^\infty$ satisfy the standard Assumption 2 for stochastic optimization algorithms. Note that the choice of steps proportional to $1/t$ has been proved to be a satisfactory learning rate, theoretically speaking and also for practical implementations (see for instance Murata 1998 and Bottou and LeCun 2004).

For practical implementation, the sequences $\left\{\varepsilon_{t+1}^i\right\}_{t=0}^\infty$ satisfying Assumption 8 can be implemented without a global clock, that is, without assuming that the current value of t is known by the agents. This assumption is satisfied, for example, by taking the current value of ε_t^i proportional to $1/n_t^i$, where n_t^i is the number of times that processor i as performed an update, that is, the cardinal of the set $T^i \cap \{0,\ldots,t\}$. For a given processor, if the time span between consecutive updates is bounded from above and from below, a straightforward examination shows that the sequence of steps satisfy Assumption 8.

Finally, the next assumption is essentially technical in nature. It enables to avoid time instants where all processors are idle. It basically requires that, at any time $t \geq 0$, there is at least one processor $i \in \{1,\ldots,M\}$ satisfying $s^i(t) \neq 0$.

Assumption 9 *One has $\sum_{j=1}^M \mathbb{1}_{\{t \in T^j\}} \geq 1$ for all $t \geq 0$.*

4.2 The Asynchronous G-lemma

The aim of this subsection is to state a useful theorem similar to Theorem 3, but adapted to our asynchronous distributed context. The precise Definition 9 of the agreement vector sequence should not cast aside the intuitive definition. The reader should keep in mind that the vector $w^\star(t)$ is also the asymptotical consensus if descent terms are zero after time t. Consequently, even if the agreement vector $\{w^\star(t)\}_{t=0}^\infty$ is adapted to the filtration $\{\mathcal{F}_t\}_{t=0}^\infty$, the vector $w^\star(t)$ cannot be accessible for a user at time t. Nevertheless, the agreement vector $w^\star(t)$ can be interpreted as a "probabilistic state" of the whole distributed quantization scheme at time t. This explains why the agreement vector is a such convenient tool for the analysis of the DALVQ convergence and will be central in our adaptation of G-lemma, Theorem 10.

Let us remark that Equation (10), writes for all $t \geq 0$,

$$\begin{aligned} w^\star(t+1) &= w^\star(t) + \sum_{j=1}^M \phi^j(t) s^j(t) \\ &= w^\star(t) - \sum_{j=1}^M \mathbb{1}_{\{t \in T^j\}} \phi^j(t) \varepsilon_{t+1}^j H\left(\mathbf{z}_{t+1}^j, w^j(t)\right). \end{aligned}$$

We recall the reader that the $[0,1]$-valued functions ϕ^j's are defined in Lemma 7.

Using the function h defined by identity (2) and the fact that the random variables $w^\star(t)$ and $w^j(t)$ are $\mathcal{F}_t$-measurable then it holds

$$h(w^\star(t)) = \mathbb{E}\left\{H\left(\mathbf{z}, w^\star(t)\right) \mid \mathcal{F}_t\right\}, \quad t \geq 0.$$

and

$$h(w^j(t)) = \mathbb{E}\left\{H\left(\mathbf{z}, w^j(t)\right) \mid \mathcal{F}_t\right\}, \quad j \in \{1, \ldots, M\} \text{ and } t \geq 0.$$

where $\mathbf{z}$ is a random variable of law μ independent of $\mathcal{F}_t$.

For all $t \geq 0$, set

$$\varepsilon_{t+1}^\star \triangleq \sum_{j=1}^{M} \mathbb{1}_{\{t \in T^j\}} \phi^j(t) \varepsilon_{t+1}^j. \tag{13}$$

Clearly, the real numbers $\varepsilon_t^\star$ are nonnegative. Their strictly positiveness will be discussed in Proposition 3.

Set

$$\Delta M_t^{(1)} \triangleq \sum_{j=1}^{M} \mathbb{1}_{\{t \in T^j\}} \phi^j(t) \varepsilon_{t+1}^j \left(h(w^\star(t)) - h(w^j(t))\right), \quad t \geq 0, \tag{14}$$

and

$$\Delta M_t^{(2)} \triangleq \sum_{j=1}^{M} \mathbb{1}_{\{t \in T^j\}} \phi^j(t) \varepsilon_{t+1}^j \left(h(w^j(t)) - H\left(\mathbf{z}_{t+1}^j, w^j(t)\right)\right), \quad t \geq 0. \tag{15}$$

Note that $\mathbb{E}\left\{\Delta M_t^{(2)}\right\} = 0$ and, consequently, that the random variables $\Delta M_t^{(2)}$ can be seen as the increments of a martingale with respect to the filtration $\{\mathcal{F}_t\}_{t=0}^\infty$.

Finally, with this notation, equation (10) takes the form

$$w^\star(t+1) = w^\star(t) - \varepsilon_{t+1}^\star h(w^\star(t)) + \Delta M_t^{(1)} + \Delta M_t^{(2)}, \quad t \geq 0. \tag{16}$$

We are now in a position to state our most useful tool, which is similar in spirit to the G-lemma, but adapted to the context of distributed asynchronous stochastic gradient descent algorithm.

Theorem 10 (Asynchronous G-lemma) *Assume that* $(\mathbf{AsY})_\mathbf{1}$ *or* $(\mathbf{AsY})_\mathbf{2}$ *and Assumption 1 hold and that the following conditions are satisfied:*

1. $\sum_{t=0}^\infty \varepsilon_t^\star = \infty$ *and* $\varepsilon_t^\star \xrightarrow[t\to\infty]{} 0$.
2. *The sequences* $\{w^\star(t)\}_{t=0}^\infty$ *and* $\{h(w^\star(t))\}_{t=0}^\infty$ *are bounded a.s.*
3. *The series* $\sum_{t=0}^\infty \Delta M_t^{(1)}$ *and* $\sum_{t=0}^\infty \Delta M_t^{(2)}$ *converge a.s. in* $\left(\mathbb{R}^d\right)^\kappa$.
4. *There exists a lower semi-continuous function* $G : \left(\mathbb{R}^d\right)^\kappa \longrightarrow [0, \infty)$ *such that*

$$\sum_{t=0}^{\infty} \varepsilon_{t+1}^\star G(w^\star(t)) < \infty, \quad a.s.$$

Then, there exists a random connected component Ξ *of* $\{G = 0\}$ *such that*

$$\operatorname{dist}(w^\star(t), \Xi) \xrightarrow[t\to\infty]{} 0, \quad a.s.$$

4.3 Trajectory Analysis

The Pagès's proof in Pagès (1997) on the almost sure convergence of the CLVQ procedure required a careful examination of the trajectories of the process $\{w(t)\}_{t=0}^{\infty}$. Thus, in this subsection we investigate similar properties and introduce the assumptions that will be needed to prove our main convergence result, Theorem 12.

The next Assumption 10 ensures that, for each processor, the quantizers stay in the support of the density.

Assumption 10 *One has*

$$\mathbb{P}\left\{w^j(t) \in \mathcal{G}^{\kappa}\right\} = 1, \quad j \in \{1,\ldots,M\} \text{ and } t \geq 0.$$

Firstly, let us mention that since the set $\mathcal{G}^{\kappa}$ is convex, if Assumption 10 holds then

$$\mathbb{P}\left\{w^{\star}(t) \in \mathcal{G}^{\kappa}\right\} = 1, \quad t \geq 0.$$

Secondly, note that the Assumption 10 is not particularly restrictive. This assumption is satisfied under the condition: for each processor, no descent term is added while a combining computation is performed. This writes

$$a_{i,j}(t) = \delta_{i,j} \text{ and } \tau^{i,i}(t) = t, \quad (i,j) \in \{1,\ldots,M\}^2 \text{ and } t \in T^i.$$

This requirement makes sense for practical implementations.

Recall that if $t \notin T^i$, then $s^i(t) = 0$. Thus, Equation (7) takes the form

$$w^i(t+1) = \begin{cases} w^i(t+1) & = w^i(t) - \varepsilon_{t+1}^i \left(w^i(t) - \mathbf{z}_{t+1}^i\right) \\ & = \left(1 - \varepsilon_{t+1}^i\right) w^i(t) + \varepsilon_{t+1}^i \mathbf{z}_{t+1}^i \\ w^i(t+1) & = \sum_{j=1}^{M} a^{i,j}(t) w^j(\tau^{i,j}(t)) \end{cases} \begin{array}{l} \text{if } t \in T^i; \\ \\ \text{otherwise.} \end{array}$$

Since $\mathcal{G}^{\kappa}$ is a convex set, it follows easily that if $w^j(0) \in \mathcal{G}^{\kappa}$, then $w^j(t) \in \mathcal{G}^{\kappa}$ for all $j \in \{1,\ldots,M\}$ and $t \geq 0$ and, consequently, that Assumption 10 holds.

The next Lemma 11 provides a deterministic upper bound on the differences between the distributed versions w^i and the agreement vector. For any subset A of $\left(\mathbb{R}^d\right)^{\kappa}$, the notation $\operatorname{diam}(A)$ stands for the usual diameter defined by

$$\operatorname{diam}(A) = \sup_{x,y \in A} \{\|x - y\|\}.$$

Lemma 11 *Assume* $(\mathbf{AsY})_{\mathbf{1}}$ *or* $(\mathbf{AsY})_{\mathbf{2}}$ *holds and that Assumptions 1, 8 and 10 are satisfied then*

$$\|w^{\star}(t) - w^i(t)\| \leq \sqrt{\kappa} M \operatorname{diam}(\mathcal{G}) A K_2 \theta_t, \quad i \in \{1,\ldots,M\} \text{ and } t \geq 0, \text{ a.s.},$$

where $\theta_t \triangleq \sum_{\tau=-1}^{t-1} \frac{1}{\tau \vee 1} \rho^{t-\tau}$, *A and* ρ *are the constants introduced in Lemma 8,* K_2 *is defined in Assumption 8.*

The sequence $\{\theta_t\}_{t=0}^{\infty}$ defined in Lemma 11 satisfies

$$\theta_t \xrightarrow[t \to \infty]{} 0 \text{ and } \sum_{t=0}^{\infty} \frac{\theta_t}{t} < \infty. \tag{17}$$

We give some calculations justifying the statements at the end of the Annex.

Thus, under Assumptions 8 and 10, it follows easily that

$$w^\star(t) - w^i(t) \xrightarrow[t\to\infty]{} 0, \quad i \in \{1,\ldots,M\}, \text{a.s.},$$

and

$$w^i(t) - w^j(t) \xrightarrow[t\to\infty]{} 0, \quad (i,j) \in \{1,\ldots,M\}^2, \text{a.s.} \tag{18}$$

This shows that the trajectories of the distributed versions of the quantizers reach asymptotically a consensus with probability 1. In other words, if one of the sequences $\left\{w^i(t)\right\}_{t=0}^{\infty}$ converges then they all converge towards the same value. The rest of the paper is devoted to prove that this common value is in fact a zero of ∇C, that is, a critical point.

To prove the result mentioned above, we will need the following assumption, which basically states that the components of $w^\star$ are parted, for every time t but also asymptotically. This assumption is similar in spirit to the main requirement of Theorem 4.

Assumption 11 *One has*

1. $\mathbb{P}\left\{w^\star(t) \in \mathcal{D}_*^\kappa\right\} = 1, \quad t \geq 0.$
2. $\mathbb{P}\left\{\liminf_{t\to\infty} \operatorname{dist}\left(w^\star(t), \complement\mathcal{D}_*^\kappa\right) > 0\right\} = 1, \quad t \geq 0.$

4.4 Consistency of the DALVQ

In this subsection we state our main theorem on the consistency of the DALVQ. Its proof is based on the asynchronous G-lemma, Theorem 10. The goal of the next proposition is to ensure that the first assumption of Theorem 10 holds.

Proposition 3 *Assume* $(\mathbf{AsY})_\mathbf{1}$ *or* $(\mathbf{AsY})_\mathbf{2}$ *holds and that Assumptions 1, 8 and 9 are satisfied then* $\varepsilon_t^\star > 0, t \geq 0, \varepsilon_t^\star \xrightarrow[t\to\infty]{} 0$ *and* $\sum_{t=0}^{\infty} \varepsilon_t^\star = \infty$.

The second condition required in Theorem 10 deals with the convergence of the two series defined by Equations (14) and (15). The next Proposition 4 provides sufficient condition for the almost sure convergence of these series.

Proposition 4 *Assume* $(\mathbf{AsY})_\mathbf{1}$ *or* $(\mathbf{AsY})_\mathbf{2}$ *holds and that Assumptions 1, 8, 10 and 11 are satisfied then the series* $\sum_{t=0}^{\infty} \Delta M_t^{(1)}$ *and* $\sum_{t=0}^{\infty} \Delta M_t^{(2)}$ *converge almost surely in* $\left(\mathbb{R}^d\right)^\kappa$.

This next proposition may be considered has the most important step in the proof of the convergence of the DALVQ. It establishes the convergence of a series of the form $\sum_{t=0}^{\infty} \varepsilon_{t+1} \left\|\nabla C\left(w(t)\right)\right\|^2$. The analysis of the convergence of this type of series is standard for the analysis of stochastic gradient method (see for instance Benveniste et al. 1990 and Bottou 1991). In our context, we pursue the fruitful use of the agreement vector sequence, $\{w^\star(t)\}_{t=0}^{\infty}$, and its related "steps", $\{\varepsilon_t^\star\}_{t=0}^{\infty}$.

Note that under Assumption 11, we have $h\left(w^\star(t)\right) = \nabla C\left(w^\star(t)\right)$ for all $t \geq 0$, almost surely, therefore the sequence $\{\nabla C\left(w^\star(t)\right)\}_{t=0}^{\infty}$ below is well defined.

Proposition 5 *Assume* $(\mathbf{AsY})_\mathbf{1}$ *or* $(\mathbf{AsY})_\mathbf{2}$ *holds and that Assumptions 1, 8, 10 and 11 are satisfied then*

1. $C(w^\star(t)) \xrightarrow[t\to\infty]{} C_\infty, \quad a.s.,$
   where C_∞ is a $[0,\infty)$-valued random variable,

2.
$$\sum_{t=0}^{\infty} \varepsilon_{t+1}^\star \left\|\nabla C(w^\star(t))\right\|^2 < \infty, \quad a.s. \tag{19}$$

Remark that from the convergence of the series given by Equation (19) one can only deduce that $\liminf_{t\to\infty} \|\nabla C(w^\star(t))\| = 0$.

We are now in a position to state the main theorem of this paper, which expresses the convergence of the distributed version towards some zero of the gradient of the distortion. In addition, the convergence results (18) imply that if a version converges then all the versions converge towards this value.

Theorem 12 (Asynchronous theorem) *Assume $(\mathbf{AsY})_\mathbf{1}$ or $(\mathbf{AsY})_\mathbf{2}$ holds and that Assumptions 1, 8, 9, 10 and 11 are satisfied then*

1. $w^*(t) - w^i(t) \xrightarrow[t\to\infty]{} 0, \quad i \in \{1,\ldots,M\}, a.s.,$
2. $w^i(t) - w^j(t) \xrightarrow[t\to\infty]{} 0, \quad (i,j) \in \{1,\ldots,M\}^2, a.s.,$
3. $\operatorname{dist}(w^\star(t), \Xi_\infty) \xrightarrow[t\to\infty]{} 0, \quad a.s.,$
4. $\operatorname{dist}\left(w^i, \Xi_\infty\right) \xrightarrow[t\to\infty]{} 0, \quad i \in \{1,\ldots,M\}, a.s.,$

where Ξ_∞ is some random connected component of the set $\{\nabla C = 0\} \cap \mathcal{G}^\kappa$.

4.5 Annex

Sketch of the proof of asynchronous G-lemma 10. The proof is an adaptation of the one found by Fort and Pagès, Theorem 4 in Fort and Pagès (1996). The recursive equation (16) satisfied by the sequence $\{w^\star(t)\}_{t=0}^\infty$ is similar to the iterations (2) in Fort and Pagès (1996), with the notation of this paper:

$$X^{t+1} = X^t - \varepsilon_{t+1} h\left(X^t\right) + \varepsilon_{t+1}\left(\Delta M^{t+1} + \eta^{t+1}\right), \quad t \geq 0.$$

Thus, similarly, we define a family of continuous time stepwise function $\{u \mapsto \check{w}(t,u)\}_{t=1}^\infty$.

$$\check{w}^\star(0,u) \triangleq w^\star(s), \text{ if } u \in [\varepsilon_1^\star + \ldots + \varepsilon_s^\star, \varepsilon_1^\star + \ldots + \varepsilon_{s+1}^\star), \quad u \in [0,\infty).$$

and if $u < \varepsilon_1^\star$, $\check{w}^\star(0,u) = w^\star(0)$.

$$\check{w}^\star(t,u) \triangleq \check{w}^\star\left(0, \varepsilon_1^\star + \ldots + \varepsilon_t^\star + u\right), \quad t \geq 1 \text{ and } u \in [0,\infty).$$

Hence, for every $t \in \mathbb{N}$,

$$\check{w}^\star(t,u) = \check{w}^\star(0,t) - \int_0^u h\left(\check{w}^\star(t,v)\right) dv + R_u(t), \quad u \in [0,\infty),$$

where, for every $t \geq 1$ and $u \in [\varepsilon_1^\star + \ldots + \varepsilon_{t+t'}^\star, \varepsilon_1^\star + \ldots + \varepsilon_{t+t'+1}^\star)$,

$$R_u(t) \triangleq \int_{\varepsilon_t^\star + \ldots + \varepsilon_{t+t'}^\star}^{\varepsilon_1^\star + \ldots + \varepsilon_t^\star + u} \breve{w}^\star(0,v)dv + \sum_{s=t+1}^{t+t'} \left(\Delta M_s^{(1)} + \Delta M_s^{(2)} \right).$$

The only difference between the families of continuous time functions $\{\breve{w}(t,u)\}_{t=1}^\infty$ and $\left\{X^{(t)}\right\}_{t=1}^\infty$ defined in Fort and Pagès (1996) is the remainder term $R_u(t)$. The convergence

$$\sup_{u \in [0,T]} \|R_u(t)\| \xrightarrow[t\to\infty]{} 0, \quad T > 0.$$

follows easily from the third assumption of Theorem 10. The rest of the proof follows similarly as in Fort and Pagès (1996, Theorem 4).

Proof of Lemma 11 For all $i \in \{1, \ldots, M\}$, and all $t \geq 0$, and all $1 \leq \ell \leq \kappa$, we may write

$$\begin{aligned}
&\left\| w_\ell^i(t) - w_\ell^\star(t) \right\| \\
&= \left\| \sum_{j=1}^M \left(\left(\phi^{i,j}(t,-1) - \phi^j(-1) \right) w_\ell^j(0) + \sum_{\tau=0}^{t-1} \left(\phi^{i,j}(t,\tau) - \phi^j(t) \right) s_\ell^j(\tau) \right) \right\| \\
&\qquad \text{(by Definition 9 and Lemma 7)} \\
&\leq \sum_{j=1}^M \left| \phi^{i,j}(t,-1) - \phi^j(-1) \right| \left\| w_\ell^j(0) \right\| + \sum_{\tau=0}^{t-1} \sum_{j=1}^M \left| \phi^{i,j}(t,\tau) - \phi^j(t) \right| \left\| s_\ell^j(\tau) \right\| \\
&\leq A\rho^{t+1} \sum_{j=1}^M \left\| w_\ell^j(0) \right\| + A \sum_{\tau=0}^{t-1} \sum_{j=1}^M \rho^{t-\tau} \left\| s_\ell^j(\tau) \right\| \\
&\qquad \text{(by Lemma 8).}
\end{aligned}$$

Thus,

$$\begin{aligned}
&\left\| w_\ell^i(t) - w_\ell^\star(t) \right\| \\
&\quad \leq A\rho^{t+1} \sum_{j=1}^M \left\| w_\ell^j(0) \right\| + A \sum_{\tau=0}^{t-1} \sum_{j=1}^M \rho^{t-\tau} \varepsilon_{\tau+1}^j \mathbb{1}_{\{\tau \in T^j\}} \left\| H(\mathbf{z}_{\tau+1}^j, w^j(\tau))_\ell \right\| \\
&\qquad \text{(by Equation (12))} \\
&\quad \leq A\rho^{t+1} \sum_{j=1}^M \left\| w_\ell^j(0) \right\| \\
&\qquad + A \sum_{\tau=0}^{t-1} \sum_{j=1}^M \rho^{t-\tau} \varepsilon_{\tau+1}^j \mathbb{1}_{\tau \in T^j} \mathbb{1}_{\left\{ \mathbf{z}_{\tau+1}^j \in W_\ell(w^j(\tau)) \right\}} \left\| w_\ell^j(\tau) - \mathbf{z}_{\tau+1}^j \right\|.
\end{aligned}$$

Therefore,

$$\begin{aligned}
&\left\|w_\ell^i(t) - w_\ell^\star(t)\right\| \\
&\quad \le AM\,\mathrm{diam}(\mathcal{G})\rho^{t+1} + A\,\mathrm{diam}(\mathcal{G})K_2M\sum_{\tau=0}^{t-1}\frac{1}{\tau\vee 1}\rho^{t-\tau} \\
&\qquad \text{(because } 0\in\mathcal{G} \text{ and by Assumptions 8 and 10)} \\
&\quad \le A\,\mathrm{diam}(\mathcal{G})K_2M\sum_{\tau=-1}^{t-1}\frac{1}{\tau\vee 1}\rho^{t-\tau}.
\end{aligned}$$

Consequently,

$$\begin{aligned}
&\left\|w^\star(t) - w^i(t)\right\| \\
&\quad = \sqrt{\sum_{\ell=1}^{\kappa}\left\|w_\ell^i(t) - w_\ell^\star(t)\right\|^2} \\
&\quad \le \sqrt{\kappa}M\,\mathrm{diam}(\mathcal{G})AK_2\sum_{\tau=-1}^{t-1}\frac{1}{\tau\vee 1}\rho^{t-\tau}.
\end{aligned}$$

This proves the desired result. ■

Let us now introduce the following events: for any $\delta > 0$ and $t \ge 0$,

$$A_\delta^t \triangleq \left\{w^\star(\tau) \in \mathcal{G}_\delta^\kappa\,,\ t \ge \tau \ge 0\right\}.$$

Recall that the $\mathcal{G}_\delta^\kappa$ is a compact subset of $\mathcal{G}^\kappa$ defined by Equality (4). The next lemma establishes a detailed analysis of security regions for the parted components of the sequences $\{w^\star(t)\}_{t=0}^\infty$ and $\left\{w^j(t)\right\}_{t=0}^\infty$.

Lemma 13 *Let Assumptions 8 and 10 hold. Then,*

1. *there exists an integer $t_\delta^1 \ge 1$ such that*

$$A_\delta^t \subset A_{\delta/2}^{t+1}, \quad t \ge t_\delta^1.$$

Moreover,

$$w^\star(t) \in \mathcal{G}_\delta^\kappa \Rightarrow [w^\star(t), w^\star(t+1)] \subset \mathcal{G}_{\delta/2}^\kappa, \quad t \ge t_\delta^1.$$

2. *There exists an integer $t_\delta^2 \ge 1$ such that*

$$w^\star(t) \in \mathcal{G}_\delta^\kappa \Rightarrow [w^\star(t), w^i(t)] \subset \mathcal{G}_{\delta/2}^\kappa, \quad i \in \{1,\ldots,M\} \text{ and } t \ge t_\delta^2.$$

Proof of Lemma 13 *Proof of statement 1.* The proof starts with the observation that under Assumption 10 we have $w^j(t) \in \mathcal{G}^\kappa$, for all $i \in \{1,\ldots,M\}$ and $t \ge 0$. It follows that, for any $1 \le \ell \le \kappa$,

$$\begin{aligned}
\left\|H\left(\mathbf{z}_{t+1}^j, w^j(t)\right)_\ell\right\| &\le \left\|\mathbf{z}_{t+1}^j - w_\ell^j(t)\right\| \\
&\le \mathrm{diam}(\mathcal{G}).
\end{aligned}$$

Let us now provide an upper bound on the norm of the differences between two consecutive values of the agreement vector sequence. We may write, for all $t \geq 0$ and all $1 \leq \ell \leq M$,

$$\begin{aligned}
&\|w_\ell^\star(t+1) - w_\ell^\star(t)\| \\
&\quad = \left\| \sum_{j=1}^{M} \phi^j(t) s_\ell^j(t) \right\| \\
&\quad \leq \sum_{j=1}^{M} \phi^j(t) \left\| s_\ell^j(t) \right\| \\
&\quad \leq \sum_{j=1}^{M} \varepsilon_{t+1}^j \mathbb{1}_{\{t \in T^j\}} \left\| H\left(\mathbf{z}_{t+1}^j, w^j(t)\right)_\ell \right\| \\
&\qquad \text{(by Equation (12) and statement 1. of Lemma 8)} \\
&\quad \leq \frac{M \operatorname{diam}(\mathcal{G}) K_2}{t \vee 1} \qquad (20) \\
&\qquad \text{(by Assumption 8).}
\end{aligned}$$

Take $t \geq \frac{4}{\delta} M \operatorname{diam}(\mathcal{G}) K_2$ and $1 \leq k \neq \ell \leq M$. Let α be a real number in the interval $[0,1]$. If $w^\star(t) \in \mathcal{G}_\delta^\kappa$ then

$$\begin{aligned}
&\|(1-\alpha)w_\ell^\star(t) + \alpha w_\ell^\star(t+1) - (1-\alpha)w_k^\star(t) - \alpha w_k^\star(t+1)\| \\
&\quad = \|w_\ell^\star(t) - w_k^\star(t) + \alpha\left(w_\ell^\star(t+1) - w_\ell^\star(t)\right) + \alpha\left(w_k^\star(t) - w_k^\star(t+1)\right)\| \\
&\quad \geq \|w_\ell^\star(t) - w_k^\star(t)\| - \|\alpha\left(w_\ell^\star(t+1) - w_\ell^\star(t)\right) + \alpha\left(w_k^\star(t) - w_k^\star(t+1)\right)\| \\
&\quad \geq \|w_\ell^\star(t) - w_k^\star(t)\| - \alpha\|w_\ell^\star(t+1) - w_\ell^\star(t)\| - \alpha\|w_k^\star(t) - w_k^\star(t+1)\| \\
&\quad \geq \delta - 2\alpha\frac{\delta}{4} \\
&\quad \geq \delta/2.
\end{aligned}$$

This proves that the whole segment $[w^\star(t), w^\star(t+1)]$ is contained in $\mathcal{G}_{\delta/2}^\kappa$.

Proof of statement 2. Take $t \geq 1$ and $1 \leq \ell \leq M$. If $w^\star(t) \in \mathcal{G}_\delta^\kappa$ then by Lemma 11, there exists t_δ^2 such that

$$\left\|w_\ell^\star(t) - w_\ell^i(t)\right\| \leq \frac{\delta}{4}, \quad i \in \{1,\ldots,M\} \text{ and } t \geq t_\delta^2.$$

Let k and ℓ two distinct integers between 1 and M. For any $t \geq t_\delta^2$,

$$\begin{aligned}
&\left\|\alpha w_k^i(t) + (1-\alpha)w_k^\star(t) - \alpha w_\ell^i(t) - (1-\alpha)w_\ell^\star(t)\right\| \\
&\quad = \left\|w_k^\star(t) - w_\ell^\star(t) + \alpha(w_k^i(t) - w_k^\star(t)) + \alpha(w_\ell^\star(t) - w_\ell^i(t))\right\| \\
&\quad \geq \|w_k^\star(t) - w_\ell^\star(t)\| - \alpha\left\|w_k^i(t) - w_k^\star(t)\right\| - \alpha\left\|w_\ell^\star(t) - w_\ell^i(t)\right\| \\
&\quad \geq \delta - 2\alpha\frac{\delta}{4} \\
&\quad \geq \delta/2.
\end{aligned}$$

This implies $[w^\star(t), w^i(t)] \subset \mathcal{G}_{\delta/2}^\kappa$, as desired. ■

Proof of Proposition 3 By definition $\varepsilon_{t+1}^{\star}$ equals $\sum_{j=1}^{M} \mathbb{1}_{\{t \in T^j\}} \phi^j(t) \varepsilon_{t+1}^j$, for all $t \geq 0$.

On the one hand, since the real number $\phi^j(t)$ belongs to the interval $[\eta, 1]$ (by Lemma 8) $\varepsilon_{t+1}^{\star}$ is bounded from above by $\frac{MK_2}{t \vee 1}$ using the right-hand side inequality of Assumption 8.

On the other hand, $\varepsilon_{t+1}^{\star}$ is bounded from below by the nonnegative real number $\eta \frac{K_1}{t \vee 1}$ using the left-hand side inequality of Assumption 8. Note also that as Assumption 9 holds, this real number is a positive one. Therefore, it follows that

$$\varepsilon_t^{\star} \xrightarrow[t \to \infty]{} 0$$

and

$$\sum_{t=0}^{\infty} \varepsilon_t^{\star} = \infty.$$

■

Proof of Proposition 4 *Consistency of* $\sum_{t=0}^{\infty} \Delta M_t^{(1)}$. Let δ be a positive real number and let $t \geq t_\delta^2$, where t_δ^2 is given by Lemma 19. We may write

$$\begin{aligned}
&\mathbb{1}_{A_\delta^t} \sum_{j=1}^{M} \mathbb{1}_{\{t \in T^j\}} \phi^j(t) \varepsilon_{t+1}^j \left\| h\left(w^\star(t)\right) - h\left(w^j(t)\right) \right\| \\
&\leq \mathbb{1}_{\left\{[w^\star(t), w^j(t)] \subset \mathcal{G}_{\delta/2}^{\kappa}\right\}} \sum_{j=1}^{M} \phi^j(t) \varepsilon_{t+1}^j \left\| \nabla C\left(w^\star(t)\right) - \nabla C\left(w^j(t)\right) \right\| \\
&\qquad \text{(using statement 2. of Lemma 13 and the fact that } \nabla C = h \text{ on } \mathcal{D}_*^{\kappa}) \\
&\leq \mathbb{1}_{\left\{[w^\star(t), w^j(t)] \subset \mathcal{G}_{\delta/2}^{\kappa}\right\}} P_{\delta/2} \sum_{j=1}^{M} \varepsilon_{t+1}^j \left\| w^\star(t) - w^j(t) \right\| \\
&\qquad \text{(by Lemma 2)} \\
&\leq \sqrt{\kappa}\, \mathrm{diam}(\mathcal{G}) A K_2^2 P_{\delta/2} M^2 \frac{\theta_t}{t} \\
&\qquad \text{(by Lemma 11).}
\end{aligned}$$

Thus, since $\sum_{t=0}^{\infty} \frac{\theta_t}{t} < \infty$, the series

$$\sum_{t=0}^{\infty} \mathbb{1}_{A_\delta^t} \sum_{j=1}^{M} \mathbb{1}_{\{t \in T^j\}} \phi^j(t) \varepsilon_{t+1}^j \left\| h\left(w^\star(t)\right) - h\left(w^j(t)\right) \right\|$$

is almost surely convergent. Under Assumption 11, we have

$$\mathbb{P}\left\{ \bigcup_{\delta > 0} \bigcap_{t \geq 0} A_\delta^t \right\} = 1.$$

It follows that the series $\sum_{t=0}^{\infty} \Delta M_t^{(1)}$ converges almost surely in $\left(\mathbb{R}^d\right)^{\kappa}$.

Consistency of $\sum_{t=0}^{\infty} \Delta M_t^{(2)}$. The sequence of random variables $M_t^{(2)}$ defined, for all $t \geq 0$, by

$$\begin{aligned} M_t^{(2)} &\triangleq \sum_{\tau=0}^{t} \Delta M_\tau^{(2)} \\ &= \sum_{\tau=0}^{t} \sum_{j=1}^{M} \mathbb{1}_{\{\tau \in T^j\}} \varepsilon_{\tau+1}^j \phi^j(\tau) \left(h\left(w^j(\tau)\right) - H\left(\mathbf{z}_{\tau+1}^j, w^j(\tau)\right)\right). \end{aligned}$$

is a vector valued martingale with respect to the filtration $\{\mathcal{F}_t\}_{t=0}^{\infty}$. It turns out that this martingale has square integrable increments. Precisely,

$$\sum_{t=0}^{\infty} \mathbb{E}\left\{ \left\| M_{t+1}^{(2)} - M_t^{(2)} \right\|^2 \;\middle|\; \mathcal{F}_t \right\} = \sum_{t=1}^{\infty} \mathbb{E}\left\{ \left\| \Delta M_t^{(2)} \right\|^2 \;\middle|\; \mathcal{F}_t \right\} < \infty.$$

Indeed, for all $j \in \{1, \ldots, M\}$ and $t \geq 1$,

$$\begin{aligned} &\sum_{\tau=1}^{t} \mathbb{E}\left\{ \left\| \mathbb{1}_{\{\tau \in T^j\}} \varepsilon_{\tau+1}^j \left(h\left(w^j(\tau)\right) - H\left(\mathbf{z}_{\tau+1}^j(\tau), w^j(\tau)\right)\right) \right\|^2 \mid \mathcal{F}_\tau \right\} \\ &\quad \leq \sum_{\tau=1}^{t} \left(\varepsilon_{\tau+1}^j\right)^2 \mathbb{E}\left\{ \left\| h\left(w^j(\tau)\right) - H\left(\mathbf{z}_{\tau+1}^j(\tau), w^j(\tau)\right) \right\|^2 \mid \mathcal{F}_\tau \right\} \\ &\quad \leq 2 \sum_{\tau=1}^{t} \left(\varepsilon_{\tau+1}^j\right)^2 \mathbb{E}\left\{ \left\| h\left(w^j(\tau)\right) \right\|^2 + \left\| H\left(\mathbf{z}_{\tau+1}^j(\tau), w^j(\tau)\right) \right\|^2 \mid \mathcal{F}_\tau \right\} \\ &\quad \leq 4\kappa \operatorname{diam}(\mathcal{G})^2 \sum_{\tau=1}^{t} \left(\varepsilon_{\tau+1}^j\right)^2 \\ &\qquad \text{(using Assumption 10)} \\ &\quad \leq 4\kappa \operatorname{diam}(\mathcal{G})^2 K_2^2 \sum_{\tau=1}^{t} \frac{1}{\tau^2}. \end{aligned}$$

We conclude that the series $\sum_{t \geq 1} \Delta M_t^{(2)}$ is almost surely convergent. ■

Proof of Proposition 5 Denote by $\langle x, y \rangle$ the canonical inner product of two vectors $x, y \in \mathbb{R}^d$ and also, with a slight abuse of notation, the canonical inner product of two vectors $x, y \in \left(\mathbb{R}^d\right)^\kappa$. Let δ be a positive real number. Take any $t \geq \max\left\{t_\delta^1, t_\delta^2\right\}$, where t_δ^1 and t_δ^2 are defined as in Lemma 13. One has,

$$\mathbb{1}_{A_\delta^{t+1}} C\left(w^\star(t+1)\right) \leq \mathbb{1}_{A_\delta^t} C\left(w^\star(t+1)\right).$$
$$\text{(by definition } A_\delta^{t+1} \subset A_\delta^t\text{)}$$

Consequently,

$$
\begin{aligned}
&\mathbb{1}_{A_\delta^{t+1}} C\left(w^\star(t+1)\right) \\
&\quad \leq \mathbb{1}_{A_\delta^t} C\left(w^\star(t)\right) + \mathbb{1}_{A_\delta^t} \langle \nabla C(w^\star(t)), w^\star(t+1) - w^\star(t) \rangle \\
&\qquad + \mathbb{1}_{\left\{[w^\star(t), w^\star(t+1)] \subset \mathcal{G}_{\delta/2}^\kappa\right\}} \\
&\qquad\quad \times \left[\sup_{z \in [w^\star(t), w^\star(t+1)]} \{ \| \nabla C(z) - \nabla C(w^\star(t)) \| \} \, \| w^\star(t+1) - w^\star(t) \| \right] \\
&\quad \leq \mathbb{1}_{A_\delta^t} C\left(w^\star(t)\right) + \mathbb{1}_{A_\delta^t} \langle \nabla C(w^\star(t)), w^\star(t+1) - w^\star(t) \rangle \\
&\qquad + P_{\delta/2} \, \| w^\star(t+1) - w^\star(t) \|^2 \\
&\qquad\quad \text{(using Lemma 2.)}
\end{aligned}
$$

The first inequality above holds since the bounded increment formula above is valid by statement 1 of Lemma 13. Let us now bound separately the right hand side members of the second inequality.

Firstly, the next inequality holds by Inequality (20) provided in the proof of Lemma 13,

$$
P_{\delta/2} \, \| w^\star(t+1) - w^\star(t) \|^2 \leq \kappa P_{\delta/2} \left(\frac{K_2 M \operatorname{diam}(\mathcal{G})}{t} \right)^2 .
$$

Secondly,

$$
\begin{aligned}
&\mathbb{1}_{A_\delta^t} \langle \nabla C(w^\star(t)), w^\star(t+1) - w^\star(t) \rangle \\
&\quad = \mathbb{1}_{A_\delta^t} \langle \nabla C(w^\star(t)), \sum_{j=1}^{M} \phi^j(t) s^j(t) \rangle \\
&\qquad\quad \text{(by Equation (10))} \\
&\quad = \mathbb{1}_{A_\delta^t} \sum_{j=1}^{M} \langle \nabla C(w^j(t)), \phi^j(t) s^j(t) \rangle \\
&\qquad + \mathbb{1}_{A_\delta^t} \sum_{j=1}^{M} \langle \nabla C(w^\star(t)) - \nabla C(w^j(t)), \phi^j(t) s^j(t) \rangle .
\end{aligned}
$$

Thus,

$$\begin{aligned}
&\mathbb{1}_{A_\delta^t}\langle \nabla C(w^\star(t)), w^\star(t+1)-w^\star(t)\rangle \\
&\quad \le \mathbb{1}_{A_\delta^t}\sum_{j=1}^{M}\langle \nabla C(w^j(t)), \phi^j(t)s^j(t)\rangle \\
&\qquad + \mathbb{1}_{A_\delta^t}\sum_{j=1}^{M}\left|\langle \nabla C(w^\star(t))-\nabla C(w^j(t)), \phi^j(t)s^j(t)\rangle\right| \\
&\quad \le \mathbb{1}_{A_\delta^t}\sum_{j=1}^{M}\langle \nabla C(w^j(t)), \phi^j(t)s^j(t)\rangle \\
&\qquad + \sum_{j=1}^{M}\mathbb{1}_{A_\delta^t}\left\|\nabla C(w^\star(t))-\nabla C(w^j(t))\right\|\left\|\phi^j(t)s^j(t)\right\| \\
&\qquad\quad \text{(using Cauchy-Schwarz inequality)}.
\end{aligned}$$

Therefore,

$$\begin{aligned}
&\mathbb{1}_{A_\delta^t}\langle \nabla C(w^\star(t)), w^\star(t+1)-w^\star(t)\rangle \\
&\quad \le \mathbb{1}_{A_\delta^t}\sum_{j=1}^{M}\langle \nabla C(w^j(t)), \phi^j(t)s^j(t)\rangle \\
&\qquad + \sum_{j=1}^{M}\mathbb{1}_{\left\{[w^\star(t),w^j(t)]\subset \mathcal{G}_{\delta/2}^{\kappa}\right\}}\left\|\nabla C(w^\star(t))-\nabla C(w^j(t))\right\|\left\|\phi^j(t)s^j(t)\right\| \\
&\qquad\quad \text{(by statement 2 of Lemma 13)} \\
&\quad \le \mathbb{1}_{A_\delta^t}\sum_{j=1}^{M}\langle \nabla C(w^j(t)), \phi^j(t)s^j(t)\rangle \\
&\qquad + P_{\delta/2}\sum_{j=1}^{M}\left\|w^\star(t)-w^j(t)\right\|\left\|\phi^j(t)s^j(t)\right\| \\
&\qquad\quad \text{(using Lemma 2)}
\end{aligned}$$

$$\begin{aligned}
&\mathbb{1}_{A_\delta^t}\langle \nabla C(w^\star(t)), w^\star(t+1)-w^\star(t)\rangle \\
&\quad \le \mathbb{1}_{A_\delta^t}\sum_{j=1}^{M}\langle \nabla C(w^j(t)), \phi^j(t)s^j(t)\rangle \\
&\qquad + P_{\delta/2}AK_2^2\kappa M^2\,\mathrm{diam}(\mathcal{G})^2\frac{\theta_t}{t} \\
&\qquad\quad \text{(using Lemma 11 and the upper bound (20))}.
\end{aligned}$$

Finally,

$$\begin{aligned}
&\mathbb{1}_{A_\delta^{t+1}} C\left(w^\star(t+1)\right) \\
&\quad \le \mathbb{1}_{A_\delta^t} C\left(w^\star(t)\right) + \mathbb{1}_{A_\delta^t} \sum_{j=1}^{M} \langle \nabla C(w^j(t)), \phi^j(t) s^j(t) \rangle \\
&\qquad + P_{\delta/2} A K_2^2 \kappa M^2 \operatorname{diam}(\mathcal{G})^2 \frac{\theta_t}{t} \\
&\qquad + \kappa P_{\delta/2} \left(\frac{K_2 M \operatorname{diam}(\mathcal{G})}{t} \right)^2 . \qquad (21)
\end{aligned}$$

Set

$$\Omega_\delta^1 \triangleq P_{\delta/2} A K_2^2 \kappa M^2 \operatorname{diam}(\mathcal{G})^2$$

and

$$\Omega_\delta^2 \triangleq \kappa P_{\delta/2} \left(K_2 M \operatorname{diam}(\mathcal{G}) \right)^2 .$$

■

In the sequel, we shall need the following lemma.

Lemma 14 *For all $t \ge \max\left\{t_\delta^1, t_\delta^2\right\}$, the quantity W_t below is a nonnegative supermartingale with respect to the filtration $\{\mathcal{F}_t\}_{t=0}^\infty$:*

$$\begin{aligned}
W_t \triangleq \mathbb{1}_{A_\delta^t} C\left(w^\star(t)\right) + \eta K_1 \sum_{\tau=0}^{t-1} \mathbb{1}_{A_\delta^\tau} \frac{1}{\tau} \sum_{j=1}^{M} \mathbb{1}_{\{\tau \in T^j\}} \left\| \nabla C\left(w^j(\tau)\right) \right\|^2 \\
+ \Omega_\delta^1 \sum_{\tau=t}^{\infty} \frac{\theta(\tau)}{\tau} + \Omega_\delta^2 \sum_{\tau=t}^{\infty} \frac{1}{\tau^2}, \quad t \ge 1.
\end{aligned}$$

Proof of Lemma 14 Indeed, using the upper bound provided by Equation (21),

$$\begin{aligned}
&\mathbb{E}\left\{\mathbb{1}_{A_\delta^{t+1}} C\left(w^\star(t+1)\right) \;\middle|\; \mathcal{F}_t\right\} \\
&\quad \leq \mathbb{1}_{A_\delta^t} C\left(w^\star(t)\right) + \mathbb{1}_{A_\delta^t} \sum_{j=1}^{M} \mathbb{E}\left\{\langle \nabla C(w^j(t)), \phi^j(t) s^j(t)\rangle \;\middle|\; \mathcal{F}_t\right\} \\
&\qquad + \Omega_\delta^1 \frac{1}{t}\theta_t + \Omega_\delta^2 \frac{1}{t^2} \\
&\quad = \mathbb{1}_{A_\delta^t} C\left(w^\star(t)\right) \\
&\qquad + \mathbb{1}_{A_\delta^t} \sum_{j=1}^{M} \left\langle \nabla C(w^j(t)), \mathbb{E}\left\{-\mathbb{1}_{\{t\in T^j\}} \phi^j(t) \varepsilon_{t+1}^j H(\mathbf{z}_{t+1}^j, w^j(t))\rangle \;\middle|\; \mathcal{F}_t\right\}\right\rangle \\
&\qquad + \Omega_\delta^1 \frac{\theta_t}{t} + \Omega_\delta^2 \frac{1}{t^2} \\
&\quad = \mathbb{1}_{A_\delta^t} C\left(w^\star(t)\right) \\
&\qquad - \mathbb{1}_{A_\delta^t} \sum_{j=1}^{M} \mathbb{1}_{\{t\in T^j\}} \phi^j(t) \varepsilon_{t+1}^j \left\|\nabla C(w^j(t))\right\|^2 + \Omega_\delta^1 \frac{\theta_t}{t} + \Omega_\delta^2 \frac{1}{t^2} \\
&\quad \leq \mathbb{1}_{A_\delta^t} C\left(w^\star(t)\right) \\
&\qquad - \frac{\eta K_1}{t} \mathbb{1}_{A_\delta^t} \sum_{j=1}^{M} \mathbb{1}_{\{t\in T^j\}} \left\|\nabla C(w^j(t))\right\|^2 + \Omega_\delta^1 \frac{\theta_t}{t} + \Omega_\delta^2 \frac{1}{t^2}.
\end{aligned}$$

In the last inequality we used the fact that $\phi^j(t) \geq \eta$ (Lemma 8) and $\varepsilon_{t+1}^j \geq \frac{K_1}{t}$ (Assumption 8).

It is straightforward to verify that, we have $W_t - \mathbb{E}\{W_{t+1}|\mathcal{F}_t\} \geq 0$ which prove the desired result. ■

Proof of Proposition 5 (continued) Since $\{W_t\}_{t=1}^\infty$ is a nonnegative supermartingale (by Lemma 14), W_t converges almost surely as $t \to \infty$ (see for instance Durrett 1990). Then, as $\sum_{\tau=t}^\infty \frac{\theta(\tau)}{\tau} \xrightarrow[t\to\infty]{} 0$ and $\sum_{\tau=t}^\infty \frac{1}{\tau^2} \xrightarrow[t\to\infty]{} 0$, we have

$$\mathbb{1}_{A_\delta^t} C(w^\star(t)) \xrightarrow[t\to\infty]{} C_\infty, \quad \text{a.s.}, \tag{22}$$

where $C_\infty \in [0,\infty)$ and, because the origin of the expression is increasing in t, the following series converges

$$\sum_{\tau=0}^{\infty} \mathbb{1}_{A_\delta^\tau} \frac{1}{\tau \vee 1} \sum_{j=1}^{M} \mathbb{1}_{\{\tau\in T^j\}} \left\|\nabla C\left(w^j(\tau)\right)\right\|^2 < \infty, \quad \text{a.s.} \tag{23}$$

Proof of statement 1. Assumption 11 means that

$$\mathbb{P}\left\{\bigcup_{\delta>0}\bigcap_{t\geq 0} A_\delta^t\right\} = 1.$$

Statement 1 follows easily from the convergence (22).

Proof of statement 2. The required convergence (19) is proven as follows. We have

$$\begin{aligned}
&\sum_{\tau=0}^{t} \varepsilon_{\tau+1}^{\star} \mathbb{1}_{A_\delta^\tau} \left\|\nabla C\left(w^\star(\tau)\right)\right\|^2 \\
&\quad \leq \sum_{\tau=0}^{t} \sum_{j=1}^{M} \phi^j(\tau) \mathbb{1}_{\{\tau \in T^j\}} \mathbb{1}_{A_\delta^\tau} \varepsilon_{\tau+1}^{j} \left\|\nabla C\left(w^\star(\tau)\right)\right\|^2 \\
&\qquad \text{(using Equality (13))} \\
&\quad \leq 2K_2 \sum_{\tau=0}^{t} \mathbb{1}_{A_\delta^\tau} \frac{1}{\tau \vee 1} \sum_{j=1}^{M} \mathbb{1}_{\{\tau \in T^j\}} \left\|\nabla C\left(w^j(\tau)\right)\right\|^2 \\
&\qquad \text{(using Assumption 9)} \\
&\quad + 2K_2 \sum_{\tau=0}^{t} \mathbb{1}_{\left\{[w^\star(\tau), w^j(\tau)] \subset \mathcal{G}_{\delta/2}^\kappa\right\}} \frac{1}{\tau \vee 1} \sum_{j=1}^{M} \left\|\nabla C\left(w^j(\tau)\right) - \nabla C\left(w^\star(\tau)\right)\right\|^2 \\
&\qquad \text{(using Assumption 9 and statement 2 of Lemma 13.)}
\end{aligned}$$

Thus,

$$\begin{aligned}
&\sum_{\tau=0}^{t} \varepsilon_{\tau+1}^{\star} \mathbb{1}_{A_\delta^\tau} \left\|\nabla C\left(w^\star(\tau)\right)\right\|^2 \\
&\quad \leq 2K_2 \sum_{\tau=0}^{t} \mathbb{1}_{A_\delta^\tau} \frac{1}{\tau \vee 1} \sum_{j=1}^{M} \mathbb{1}_{\{\tau \in T^j\}} \left\|\nabla C\left(w^j(\tau)\right)\right\|^2 \\
&\qquad + 2K_2 P_{\delta/2}^2 \sum_{\tau=0}^{t} \mathbb{1}_{\left\{[w^\star(\tau), w^j(\tau)] \subset \mathcal{G}_{\delta/2}^\kappa\right\}} \frac{1}{\tau \vee 1} \sum_{j=1}^{M} \left\|w^j(\tau) - w^\star(\tau)\right\|^2 \\
&\qquad \text{(by Lemma 2).}
\end{aligned}$$

Thus,

$$\begin{aligned}
&\sum_{\tau=0}^{t} \varepsilon_{\tau+1}^{\star} \mathbb{1}_{A_\delta^\tau} \left\|\nabla C\left(w^\star(\tau)\right)\right\|^2 \\
&\quad \leq 2K_2 \sum_{\tau=0}^{t} \mathbb{1}_{A_\delta^\tau} \frac{1}{\tau \vee 1} \sum_{j=1}^{M} \mathbb{1}_{\{\tau \in T^j\}} \left\|\nabla C\left(w^j(\tau)\right)\right\|^2 \\
&\qquad + 2P_{\delta/2}^2 K_2^3 \kappa M^3 A^2 \operatorname{diam}(\mathcal{G})^2 \sum_{\tau=1}^{t} \frac{1}{\tau \vee 1} \theta_\tau^2 \\
&\qquad \text{(by Lemma 11).}
\end{aligned}$$

Finally, using the convergence (23), one has

$$\sum_{\tau=0}^{\infty} \varepsilon_{\tau+1}^{\star} \mathbb{1}_{A_\delta^\tau} \left\|\nabla C\left(w^\star(\tau)\right)\right\|^2 < \infty, \quad \text{a.s.},$$

and the conclusion follows from the fact that Assumption 11 implies

$$\mathbb{P}\left\{\bigcup_{\delta>0} \bigcap_{t\geq 0} A_\delta^t\right\} = 1.$$

■

Proof of Theorem 12 The proof consists in verifying the assumptions of Theorem 10 with the function $\widehat{G}$ defined by Equation (5).

It has been outlined that Assumption 10 implies that $w^\star(t)$ lie in the compact set $\mathcal{G}^\kappa$, almost surely, for all $t \geq 0$. Consequently, in the definition of $\widehat{G}(w^\star)$ the lim inf symbol can be omitted. For all $\mathbf{z} \in \mathcal{G}$ and all $t \geq 0$, we have $\|H(\mathbf{z}, w^\star(t))\| \leq \sqrt{\kappa}\,\mathrm{diam}\,(\mathcal{G})$, almost surely, whereas $\{h(w^\star(t))\}_{t=0}^{\infty}$ satisfies

$$h(w^\star(t)) = \mathbb{E}\left\{H\left(\mathbf{z}, w^\star(t)\right) \mid \mathcal{F}_t\right\}, \quad t \geq 0, \text{a.s.}$$

Thus, the sequences $\{w^\star(t)\}_{t=0}^{\infty}$ and $\{h(w^\star(t))\}_{t=0}^{\infty}$ are bounded almost surely.

Proposition 3, respectively Proposition 4, respectively Proposition 5 show that the first assumption, respectively the third assumption, respectively the fourth assumption of Theorem 10 hold. This concludes the proof of the theorem. ■

Justification of the statements (17). Recall that the definition of θ is provided in Lemma 11. Let us remark that it is sufficient to analyse the behavior in t of the quantity $\sum_{\tau=1}^{t-1} \rho^{t-\tau}/\tau$. Let $\varepsilon > 0$ then for all $t \geq \lfloor 1/\varepsilon \rfloor + 1$, we have

$$\begin{aligned}
&\sum_{\tau=1}^{t-1} \frac{\rho^{t-\tau}}{\tau} \\
&\quad = \sum_{\tau=1}^{\lfloor 1/\varepsilon \rfloor} \frac{\rho^{t-\tau}}{\tau} + \sum_{\tau=\lfloor 1/\varepsilon \rfloor+1}^{t-1} \frac{\rho^{t-\tau}}{\tau} \\
&\quad \leq \sum_{\tau=1}^{\lfloor 1/\varepsilon \rfloor} \rho^{t-\tau} + \varepsilon \sum_{\tau=\lfloor 1/\varepsilon \rfloor+1}^{t-1} \rho^{t-\tau} \\
&\quad \leq \frac{\rho^{t-\lfloor 1/\varepsilon \rfloor}}{1-\rho} + \frac{\varepsilon}{1-\rho} \\
&\qquad \text{(using the fact that } \rho \in (0,1)\text{)}.
\end{aligned}$$

Consequently, for t sufficiently large we have

$$\sum_{\tau=1}^{t-1} \frac{\rho^{t-\tau}}{\tau} \leq \frac{2\varepsilon}{1-\rho}$$

which proves the first claim.
The second claim follows the same technique by letting "$\varepsilon = 1/\sqrt{t}$".
Thus, for $t \geq 1$ we have

$$\theta_t \leq \frac{\rho^{t-\lfloor \sqrt{t} \rfloor - 1}}{1-\rho} + \frac{1/\sqrt{t}}{1-\rho}.$$

Finally, for $T \geq 1$, it holds

$$\sum_{t=1}^{T} \sum_{\tau=1}^{t-1} \frac{\rho^{t-\tau}}{\tau} \leq \frac{1}{1-\rho} \left(\sum_{t=1}^{T} \rho^{n-\lfloor \sqrt{n} \rfloor - 1} + \sum_{t=1}^{T} \frac{1}{n^{3/2}} \right).$$

The two partial sums in the above parenthesis have finite limits which prove the second statement.

References

E. A. Abaya and G. L. Wise. Convergence of vector quantizers with applications to optimal quantization. *SIAM Journal on Applied Mathematics*, 44(1):183–189, 1984.

A. Antos. Improved minimax bounds on the test and training distortion of empirically designed vector quantizers. *IEEE Transactions on Information Theory*, 51(11):4022–4032, 2005.

A. Antos, L. Györfi, and A. György. Individual convergence rates in empirical vector quantizer design. *IEEE Transactions on Information Theory*, 51(11):4013–4022, 2005.

P. L. Bartlett, T. Linder, and G. Lugosi. The minimax distortion redundancy in empirical quantizer design. *IEEE Transactions on Information Theory*, 44(5):1802–1813, 1998.

A. Benveniste, M. Métivier, and P. Priouret. *Adaptive Algorithms and Stochastic Approximations*. Springer-Verlag, 1990.

D. P. Bertsekas and J. N. Tsitsiklis. *Parallel and Distributed Computation: Numerical Methods*. Prentice-Hall, Inc., 1989.

G. Biau, L. Devroye, and G. Lugosi. On the performance of clustering in hilbert spaces. *IEEE Transactions on Information Theory*, 54(2):781–790, 2008.

V. D. Blondel, J. M. Hendrickx, A. Olshevsky, and J. N. Tsitsiklis. Convergence in multiagent coordination, consensus, and flocking. In *Decision and Control, 2005 and 2005 European Control Conference*., 2005.

L. Bottou. Stochastic gradient learning in neural networks. In *Proceedings of Neuro-Nîmes 91*, 1991.

L. Bottou. Online algorithms and stochastic approximations. In *Online Learning and Neural Networks*. Cambridge University Press, 1998.

L. Bottou and Y. Bengio. Convergence properties of the *k*-means algorithm. In *Advances in Neural Information Processing Systems*. MIT Press, 1995.

L. Bottou and Y. LeCun. Large scale online learning. In *Advances in Neural Information Processing Systems 16*. MIT Press, 2004.

F. Bullo, J. Cortés, and S. Martínez. *Distributed Control of Robotic Networks*. Princeton University Press, 2009.

G. Choquet. *Topology*. Academic Press, 1966.

P. A. Chou. The distortion of vector quantizers trained on n vectors decreases to the optimum at $o_p(1/n)$. In *In Proceedings of IEEE International Symposium on Information Theory*, 1994.

J. W. Durham, R. Carli, P. Frasca, and F. Bullo. Discrete partitioning and coverage control with gossip communication. In *ASME Conference Proceedings*. ASME, 2009.

R. Durrett. *Probability: Theory and Examples*. Duxbury Press, 1990.

J. C. Fort and G. Pagès. On the a.s. convergence of the kohonen algorithm with a general neighborhood function. *The Annals of Applied Probability*, 5(4):1177–1216, 1995.

J. C. Fort and G. Pagès. Convergence of stochastic algorithms: from the kushner-clark theorem to the lyapounov functional method. *Advances in Applied Probability*, 28(4):1072–1094, 1996.

P. Frasca, R. Carli, and F. Bullo. Multiagent coverage algorithms with gossip communication: Control systems on the space of partitions. In *American Control Conference*, 2009.

A. Gersho and R. M. Gray. *Vector Quantization and Signal Compression*. Kluwer, 1992.

S. Graf and H. Luschgy. *Foundations of Quantization for Probability Distributions*. Springer-Verlag, 2000.

M. Inaba, N. Katoh, and H. Imai. Applications of weighted voronoi diagrams and randomization to variance-based *k*-clustering: (extended abstract). In *SCG '94: Proceedings of the Tenth Annual Symposium on Computational Geometry*, 1994.

D. Jungnickel. *Graphs, Networks and Algorithms*. Springer-Verlag, 1999.

T. Kohonen. Analysis of a simple self-organizing process. *Biological Cybernetics*, 44(2):135–140, 1982.

H. J. Kushner and D. S. Clark. *Stochastic Approximation for Constrained and Unconstrained Systems*. Springer-Verlag, 1978.

T. Linder. On the training distortion of vector quantizers. *IEEE Transactions on Information Theory*, 46(4):1617–1623, 2000.

T. Linder. Learning-theoretic methods in vector quantization. In *Lecture Notes for the Advanced School on the Principles of Nonparametric Learning*, 2001.

T. Linder, G. Lugosi, and K. Zeger. Rates of convergence in the source coding theorem, in empirical quantizer design, and in universal lossy source coding. *IEEE Transactions on Information Theory*, 40(6):1728–1740, 1994.

S. Lloyd. Least squares quantization in pcm. *IEEE Transactions on Information Theory*, 28(2): 129–137, 2003.

J. B. MacQueen. Some methods of classification and analysis of multivariate observations. In *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability*, 1967.

R. McDonald, K. Hall, and G. Mann. Distributed training strategies for the structured perceptron. In *HLT 2010: Human Language Technologies*, 2010.

N. Murata. A statistical study of on-line learning. In *Online Learning and Neural Networks*, 1998.

G. Pagès. A space vector quantization for numerical integration. *Journal of Applied and Computational Mathematics*, 89(1):1–38, 1997.

D. Pollard. Stong consistency of *k*-means clustering. *The Annals of Statistics*, 9(1):135–140, 1981.

D. Pollard. Quantization and the method of k-means. *IEEE Transactions on Information Theory*, 28(2):199–205, 1982a.

D. Pollard. A central limit theorem for k-means clustering. *The Annals of Probability*, 28(4):199–205, 1982b.

H. Robbins and S. Monro. A stochastic approximation method. *The Annals of Mathematical Statistics*, 22(3):400–407, 1951.

M. J. Sabin and R. M. Gray. Global convergence and empirical consistency of the generalized lloyd algorithm. *IEEE Transactions on Information Theory*, 32(2):148–155, 1986.

J. Tsitsiklis. *Problems in Decentralized Decision Making and Computation*. PhD thesis, Department of EECS, MIT, Cambridge, USA, 1984.

J. Tsitsiklis, D. Bertsekas, and M. Athans. Distributed asynchronous deterministic and stochastic gradient optimization algorithms. *IEEE Transactions on Automatic Control*, 31(9):803–812, 1986.

M. Zinkevich, A. Smola, and J. Langford. Slow learners are fast. In *Advances in Neural Information Processing Systems 22*. Curran Associates, 2009.