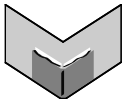
The Journal of Machine Learning Research
Volume 13
Print-Archive Edition

Pages 1263–2502



Microtome Publishing
Brookline, Massachusetts
www.mtome.com

The Journal of Machine Learning Research
Volume 13
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 2012.

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 13, 2012

Journal of Machine Learning Research 13 (2012) 1263-1291 Submitted 7/10; Revised 7/11; Published 5/12

Minimax Manifold Estimation

Christopher R. Genovese GENOVESE@STAT.CMU.EDU
Department of Statistics
Carnegie Mellon University
5000 Forbes Ave.
Pittsburgh, PA 15213, USA

Marco Perone-Pacifico MARCO.PERONEPACIFICO@UNIROMA1.IT
Dipartimento di Scienze Statistiche
Sapienza University of Rome
Piazzale Aldo Moro 5 – 00185 Roma, Italy

Isabella Verdinelli[*] ISABELLA@STAT.CMU.EDU
Larry Wasserman[†] LARRY@STAT.CMU.EDU
Department of Statistics
Carnegie Mellon University
5000 Forbes Ave.
Pittsburgh, PA 15213, USA

Editor: Ulrike von Luxburg

Abstract

We find the minimax rate of convergence in Hausdorff distance for estimating a manifold M of dimension d embedded in $\mathbb{R}^D$ given a noisy sample from the manifold. Under certain conditions, we show that the optimal rate of convergence is $n^{-2/(2+d)}$. Thus, the minimax rate depends only on the dimension of the manifold, not on the dimension of the space in which M is embedded.

Keywords: manifold learning, minimax estimation

1. Introduction

We consider the problem of estimating a manifold M given noisy observations near the manifold. The observed data are a random sample $Y_1,\ldots,Y_n$ where $Y_i \in \mathbb{R}^D$. The model for the data is

$$Y_i = \xi_i + Z_i$$

where $\xi_1,\ldots,\xi_n$ are unobserved variables drawn from a distribution supported on a manifold M with dimension $d < D$. The noise variables $Z_1,\ldots,Z_n$ are drawn from a distribution F. Our main assumption is that M is a compact, d-dimensional, smooth Riemannian submanifold in $\mathbb{R}^D$; the precise conditions on M are given in Section 2.1.

A manifold M and a distribution for (ξ, Z) induce a distribution $Q \equiv Q_M$ for Y. In Section 2.2, we define a class of such distributions

$$\mathcal{Q} = \left\{ Q_M : \; M \in \mathcal{M} \right\}$$

∗. Also in the Department of Statistical Sciences, Sapienza University of Rome, Italy.
†. Also in the Machine Learning Department, Carnegie Mellon University.

where $\mathcal{M}$ is a set of manifolds. Given two sets A and B, the Hausdorff distance between A and B is

$$H(A,B) = \inf\Big\{\varepsilon : \ A \subset B \oplus \varepsilon \quad \text{and} \quad B \subset A \oplus \varepsilon\Big\}$$

where

$$A \oplus \varepsilon = \bigcup_{x \in A} B_D(x,\varepsilon)$$

and $B_D(x,\varepsilon)$ is an open ball in $\mathbb{R}^D$ centered at x with radius ε. We are interested in the minimax risk

$$R_n(\mathcal{Q}) = \inf_{\widehat{M}} \sup_{Q \in \mathcal{Q}} \mathbb{E}_Q[H(\widehat{M},M)]$$

where the infimum is over all estimators $\widehat{M}$. By an estimator $\widehat{M}$ we mean a measurable function of $Y_1,\ldots,Y_n$ taking values in the set of all manifolds. Our first main result is the following minimax lower bound which is proved in Section 3.

Theorem 1 *Under assumptions (A1)-(A4) given in Section 2, there is a constant $C_1 > 0$ such that, for all large n,*

$$\inf_{\widehat{M}} \sup_{Q \in \mathcal{Q}} \mathbb{E}_Q\Big[H(\widehat{M},M)\Big] \geq C_1 \left(\frac{1}{n}\right)^{\frac{2}{2+d}}$$

where the infimum is over all estimators $\widehat{M}$.

Thus, no method of estimating M can have an expected Hausdorff distance smaller than the stated bound. Note that the rate depends on d but not on D even though the support of the distribution Q for Y has dimension D. Our second result is the following upper bound which is proved in Section 4.2.

Theorem 2 *Under assumptions (A1)-(A4) given in Section 2, there exists an estimator $\widehat{M}$ such that, for all large n,*

$$\sup_{Q \in \mathcal{Q}} \mathbb{E}_Q\Big[H(\widehat{M},M)\Big] \leq C_2 \left(\frac{\log n}{n}\right)^{\frac{2}{2+d}}$$

for some $C_2 > 0$.

Thus the rate is tight, up to logarithmic factors. The estimator in Theorem 2 is of theoretical interest because it establishes that the lower bound is tight. But, the estimator constructed in the proof of that theorem is not practical and so in Section 5, we construct a very simple estimator $\widehat{M}$ such that

$$\sup_{Q \in \mathcal{Q}} \mathbb{E}_Q\Big[H(\widehat{M},M)\Big] \leq \left(\frac{C\log n}{n}\right)^{1/D}.$$

This is slower than the minimax rate, but the estimator is computationally very simple and requires no knowledge of d or the smoothness of M.

1.1 Related Work

There is a vast literature on manifold estimation. Much of the literature deals with using manifolds for the purpose of dimension reduction. See, for example, Baraniuk and Wakin (2007) and references therein. We are interested instead in actually estimating the manifold itself. There is a large literature on this problem in the field of computational geometry; see, for example, Dey (2006), Dey and Goswami (2004), Chazal and Lieutier (2008) Cheng and Dey (2005) and Boissonnat and Ghosh (2010). However, very few papers allow for noise in the statistical sense, by which we mean observations drawn randomly from a distribution. In the literature on computational geometry, observations are called noisy if they depart from the underlying manifold in a very specific way: the observations have to be close to the manifold but not too close to each other. This notion of noise is quite different from random sampling from a distribution. An exception is Niyogi et al. (2008) who constructed the following estimator. Let $I = \{i : \ \widehat{p}(Y_i) > \lambda\}$ where $\widehat{p}$ is a density estimator. They define $\widehat{M} = \bigcup_{i\in I} B_D(Y_i, \varepsilon)$ and they show that if λ and ε are chosen properly, then $\widehat{M}$ is homologous to M. (This means that M and $\widehat{M}$ share certain topological properties.) However, the result does not guarantee closeness in Hausdorff distance. Note that $\bigcup_{i=1}^n B_D(Y_i, \varepsilon)$ is precisely the Devroye-Wise estimator for the support of a distribution (Devroye and Wise, 1980).

1.2 Notation

Given a set S, we denote its boundary by ∂S. We let $B_D(x,r)$ denote a D-dimensional open ball centered at x with radius r. If A is a set and x is a point then we write $d(x,A) = \inf_{y\in A} ||x-y||$ where $||\cdot||$ is the Euclidean norm. Let

$$A \circ B = (A \cap B^c) \bigcup (A^c \cap B)$$

denote symmetric set difference between sets A and B.

The uniform measure on a manifold M is denoted by μ_M. Lebesgue measure on $\mathbb{R}^k$ is denoted by ν_k. In case $k = D$, we sometimes write V instead of ν_D; in other words $V(A)$ is simply the volume of A. Any integral of the form $\int f$ is understood to be the integral with respect to Lebesgue measure on $\mathbb{R}^D$. If P and Q are two probability measures on $\mathbb{R}^D$ with densities p and q then the *Hellinger distance* between P and Q is

$$h(P,Q) \equiv h(p,q) = \sqrt{\int (\sqrt{p} - \sqrt{q})^2} = \sqrt{2\left(1 - \int \sqrt{pq}\right)}$$

where the integrals are with respect to ν_D. Recall that

$$\ell_1(p,q) \leq h(p,q) \leq \sqrt{\ell_1(p,q)} \tag{1}$$

where $\ell_1(p,q) = \int |p-q|$. Let $p(x) \wedge q(x) = \min\{p(x), q(x)\}$. The *affinity* between P and Q is

$$||P \wedge Q|| = \int p \wedge q = 1 - \frac{1}{2}\int |p-q|.$$

Let P^n denote the n-fold product measure based on n independent observations from P. In the appendix Section 7.1 we show that

$$||P^n \wedge Q^n|| \geq \frac{1}{2}\left(1 - \frac{1}{2}\int |p-q|\right)^{2n}. \tag{2}$$

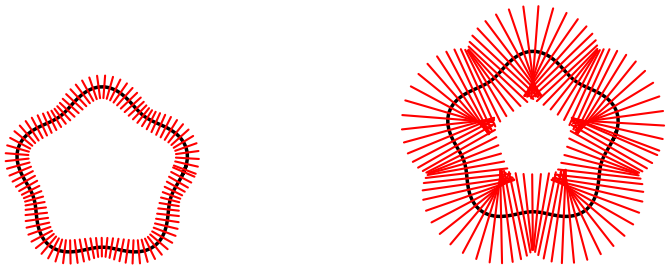


Figure 1: The condition number $\Delta(M)$ of a manifold is the largest number κ such that the normals to the manifold do not cross as long as they are not extended beyond κ. The plot on the left shows a one-dimensional manifold (a curve) and some normals of length $r < \kappa$. The plot on the right shows the same manifold and some normals of length $r > \kappa$.

We write $X_n = O_P(a_n)$ to mean that, for every $\varepsilon > 0$ there exists $C > 0$ such that $\mathbb{P}(||X_n||/a_n > C) \leq \varepsilon$ for all large n. Throughout, we use symbols like $C, C_0, C_1, c, c_0, c_1 \ldots$ to denote generic positive constants whose value may be different in different expressions.

2. Model Assumptions

In this section we describe all the assumptions on the manifold and on the underlying distributions.

2.1 Manifold Conditions

We shall be concerned with d-dimensional compact Riemannian submanifolds without boundary embedded in $\mathbb{R}^D$ with $d < D$. (Informally, this means that M looks like $\mathbb{R}^d$ in a small neighborhood around any point in M.) We assume that M is contained in some compact set $\mathcal{K} \subset \mathbb{R}^D$.

At each $u \in M$ let $T_u M$ denote the tangent space to M and let $T_u^{\perp} M$ be the normal space. We can regard $T_u M$ as a d-dimensional hyperplane in $\mathbb{R}^D$ and we can regard $T_u^{\perp} M$ as the $D-d$ dimensional hyperplane perpendicular to $T_u M$. Define the *fiber of size a at u* to be $L_a(u) \equiv L_a(u, M) = T_u^{\perp} M \bigcap B_D(u, a)$.

Let $\Delta(M)$ be the largest r such that each point in $M \oplus r$ has a unique projection onto M. The quantity $\Delta(M)$ will be small if either M highly curved or if M is close to being self-intersecting. Let $\mathcal{M} \equiv \mathcal{M}(\kappa)$ denote all d-dimensional manifolds embedded in $\mathcal{K}$ such that $\Delta(M) \geq \kappa$. Throughout this paper, κ is a fixed positive constant. The quantity $\Delta(M)$ has been rediscovered many times. It is called the *condition number* in Niyogi et al. (2006), the *thickness* in Gonzalez and Maddocks (1999) and the *reach* in Federer (1959).

An equivalent definition of $\Delta(M)$ is the following: $\Delta(M)$ is the largest number r such that the fibers $L_r(u)$ never intersect. See Figure 1. Note that if M is a sphere then $\Delta(M)$ is just the radius of the sphere and if M is a linear space then $\Delta(M) = \infty$. Also, if $\sigma < \Delta(M)$ then $M \oplus \sigma$ is the disjoint union of its fibers:

$$M \oplus \sigma = \bigcup_{u \in M} L_\sigma(u). \tag{3}$$

Define $\mathsf{tube}(M, a) = \bigcup_{u \in M} L_a(u)$. Thus, if $\sigma < \Delta(M)$ then $M \oplus \sigma = \mathsf{tube}(M, \sigma)$.

Let $p, q \in M$. The angle between two tangent spaces T_p and T_q is defined to be

$$\mathsf{angle}(T_p, T_q) = \cos^{-1}\left(\min_{u \in T_p} \max_{v \in T_q} |\langle u - p, v - q \rangle|\right)$$

where $\langle u, v \rangle$ is the usual inner product in $\mathbb{R}^D$. Let $d_M(p,q)$ denote the geodesic distance between $p, q \in M$.

We now summarize some useful results from Niyogi et al. (2006).

Lemma 3 *Let $M \subset \mathcal{K}$ be a manifold and suppose that $\Delta(M) = \kappa > 0$. Let $p, q \in M$.*

1. *Let γ be a geodesic connecting p and q with unit speed parameterization. Then the curvature of γ is bounded above by $1/\kappa$.*
2. $\cos(\mathsf{angle}(T_p, T_q)) > 1 - d_M(p,q)/\kappa$. *Thus,* $\mathsf{angle}(T_p, T_q) \leq \sqrt{2 d_M(p,q)/\kappa} + o(\sqrt{d_M(p,q)/\kappa})$.
3. *If $a = ||p - q|| \leq \kappa/2$ then $d_M(p,q) \leq \kappa - \kappa\sqrt{1 - (2a)/\kappa} = a + o(a)$.*
4. *If $a = ||p - q|| \leq \kappa/2$ then $a \geq d_M(p,q) - (d_M(p,q))^2/(2\kappa)$.*
5. *If $||q - p|| > \varepsilon$ and $v \in B_D(q, \varepsilon) \cap T_p^{\perp} M \cap B_D(p, \kappa)$ then $||v - p|| < \varepsilon^2/\kappa$.*
6. *Fix any $\delta > 0$. There exists points $x_1, \ldots, x_N \in M$ such that $M \subset \bigcup_{j=1}^{N} B_D(x_j, \delta)$ and such that $N \leq (c/\delta)^d$.*

For further information about manifolds, see Lee (2002).

2.2 Distributional Assumptions

The distribution of Y is induced by the distribution of ξ and Z. We will assume that ξ is drawn uniformly on the manifold. Then we assume that Z is drawn uniformly on the normal to M. More precisely, given ξ, we draw Z uniformly on $L_\sigma(\xi)$. In other words, the noise is perpendicular to the manifold. The result is that, if $\sigma < \kappa$, then the distribution $Q = Q_M$ of Y has support equal to $M \oplus \sigma$.

The distributional assumption on ξ is not critical. Any smooth density bounded away from 0 on the manifold will lead to similar results. However, the assumption on the noise Z is critical. We have chosen the simplest noise distribution here. (Perpendicular noise is also assumed in Niyogi et al., 2008.) In current work, we are deriving the rates for more complicated noise distributions. The rates are quite different and the proofs are more complex. Those results will be reported elsewhere.

The set of distributions we consider is as follows. Let κ and σ be fixed positive numbers such that $0 < \sigma < \kappa$. Let

$$\mathcal{Q} \equiv \mathcal{Q}(\kappa, \sigma) = \Big\{ Q_M : \ M \in \mathcal{M}(\kappa) \Big\}.$$

For any $M \in \mathcal{M}(\kappa)$ consider the corresponding distribution Q_M, supported on $S_M = M \oplus \sigma$. Let q_M be the density of Q_M with respect to Lebesgue measure. We now show that q_M is bounded above and below by a uniform density.

Recall that the essential supremum and essential infimum of q_M are defined by

$$\operatorname*{ess\,sup}_{y \in A} q_M = \inf\Big\{ a \in \mathbb{R} : \ \nu_D(\{y : \ q_M(y) > a\} \cap A) = 0 \Big\}$$

and

$$\operatorname*{ess\,inf}_{y\in A} q_M = \sup\Big\{a\in\mathbb{R}:\ \nu_D(\{y:\ q_M(y)<a\}\cap A)=0\Big\}.$$

Also recall that, by the Lebesgue density theorem, $q_M(y)=\lim_{\varepsilon\to 0} Q_M(B_D(y,\varepsilon))/V(B_D(y,\varepsilon))$ for almost all y. Let U_M be the uniform distribution on $M\oplus\sigma$ and let $u_M = 1/V(M\oplus\sigma)$ be the density of U_M. Note that, for $A\subset M\oplus\sigma$, $U_M(A)=V(A)/V(M\oplus\sigma)$.

Lemma 4 *There exist constants $0<C_*\le C^*<\infty$, depending only on κ and d, such that*

$$C_* \le \inf_{M\in\mathcal{M}} \operatorname*{ess\,inf}_{y\in S_M} \frac{q_M(y)}{u_M(y)} \le \sup_{M\in\mathcal{M}} \operatorname*{ess\,sup}_{y\in S_M} \frac{q_M(y)}{u_M(y)} \le C^*.$$

Proof Choose any $M\in\mathcal{M}(\kappa)$. Let x by any point in the interior of S_M. Let $B=B_D(x,\varepsilon)$ where $\varepsilon>0$ is small enough so that $B\subset S_M = M\oplus\sigma$. Let y be the projection of x onto M. We want to upper and lower bound $Q(B)/V(B)$. Then we will take the limit as $\varepsilon\to 0$. Consider the two spheres of radius κ tangent to M at y in the direction of the line between x and y. (See Figure 2.) Note that $Q(B)$ is maximized by taking M to be equal to the upper sphere and $Q(B)$ is minimized by taking M to be equal to the lower sphere. Let us consider first the case where M is equal to the upper sphere. Let

$$U=\Big\{u\in M:\ L_\sigma(u)\cap B\neq\emptyset\Big\}$$

be the projection of B onto M. By simple geometry, $U=M\cap B_D(y,r\varepsilon)$ where

$$\Big(1+\frac{\sigma}{\kappa}\Big)^{-1}\le r\le\Big(1+\frac{\sigma}{\kappa}\Big).$$

Let Vol denote d-dimensional volume on M. Then $\mathsf{Vol}(B_D(y,r\varepsilon)\cap M)\le c_1 r^d\varepsilon^d\omega_d$ where ω_d is the volume of a unit d-ball and c_1 depends only on κ and d. To see this, note that because M is a manifold and $\Lambda(M)\ge\kappa$, it follows that near y, M may be locally parameterized as a smooth function $f=(f_1,\ldots,f_{D-d})$ over $B\cap T_yM$. The surface area of the graph of f over $B\cap T_yM$ is bounded by $\int_{B_D(y,r\varepsilon)\cap T_yM}\sqrt{1+\|\nabla f_i\|^2}$, which is bounded by a constant c_1 uniformly over $\mathcal{M}$. Hence, $\mathsf{Vol}(B_D(y,r\varepsilon)\cap M)\le c_1\mathsf{Vol}(B_D(y,r\varepsilon)\cap T_yM)=c_1r^d\varepsilon^d\omega_d$.

Let Λ_M be the uniform distribution on M and let Γ_u denote the uniform measure on $L_\sigma(u)$. Note that, for $u\in U$, $L_\sigma(u)\cap B$ is a $(D-d)$-ball whose radius is at most ε. Hence,

$$\Gamma_u(L_\sigma(u)\cap B)\le\frac{\varepsilon^{D-d}\omega_{D-d}}{\sigma^{D-d}\omega_{D-d}}=\Big(\frac{\varepsilon}{\sigma}\Big)^{D-d}.$$

Thus,

$$\begin{aligned}
Q_M(B) &= \int_M \Gamma_u(B\cap L_\sigma(u))d\Lambda_M(u)=\int_U\Gamma_u(B\cap L_\sigma(u))d\Lambda_M(u)\\
&\le \Big(\frac{\varepsilon}{\sigma}\Big)^{D-d}\Lambda(U)=\Big(\frac{\varepsilon}{\sigma}\Big)^{D-d}\frac{\mathsf{Vol}(B_D(y,r)\cap M)}{\mathsf{Vol}(M)}\\
&\le \Big(\frac{\varepsilon}{\sigma}\Big)^{D-d}\frac{\varepsilon^d r^d\omega_d}{\mathsf{Vol}(M)}\le\Big(\frac{\varepsilon}{\sigma}\Big)^{D-d}\frac{\varepsilon^d(1+\sigma/\kappa)^d\omega_d}{\mathsf{Vol}(M)}.
\end{aligned}$$

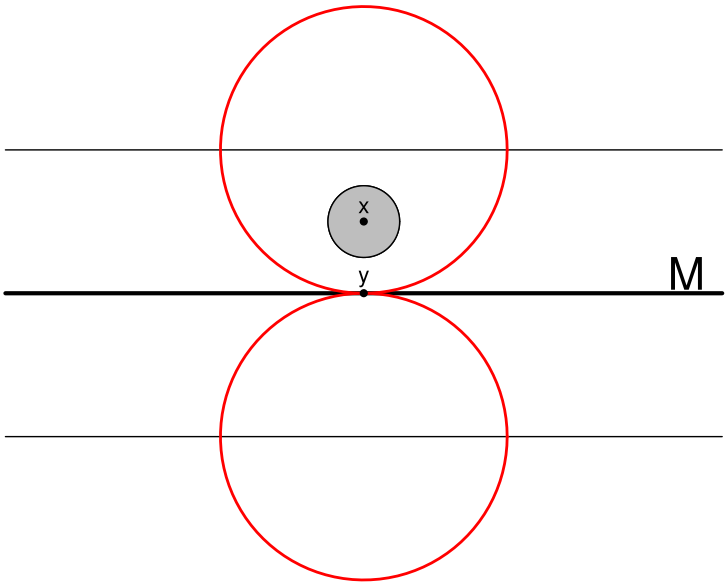


Figure 2: Figure for proof of Lemma 4. x is a point in the support $M \oplus \sigma$. y is the projection of x onto M. The two spheres are tangent to M at y and have radius κ.

Now, $U_M(B) = V(B)/V(M \oplus \sigma) = \varepsilon^D \omega_D / (\sigma^{D-d} \mathsf{Vol}(M))$. Hence,

$$\frac{Q_M(B)}{U_M(B)} \leq \left(1 + \frac{\sigma}{\kappa}\right)^d \omega_d.$$

Taking limits as $\varepsilon \to 0$ we have that $q_M(y) \leq C^* u_M(y)$ for almost all y.

The proof of the lower bound is similar to the upper bound except for the following changes: let U_0 denote all $u \in U$ such that the radius of $B \cap L_\sigma(u)$ is at least $\varepsilon/2$. Then $\Lambda(U_0) \geq \Lambda(U)(1 - O(\varepsilon))$ and the projection of U_0 onto M is again of the form $B_D(y, r\varepsilon) \cap M$. By Lemma 5.3 of Niyogi et al. (2006),

$$\mathsf{Vol}(B_D(y,r) \cap M) \geq \left(1 - \frac{r^2\varepsilon^2}{4\kappa^2}\right)^{d/2} r^d \varepsilon^d \omega_d$$

and the latter is larger than $2^{-d/2} r^d \varepsilon^d \omega_d$ for all small ε. Also, $\Gamma_u(L_\sigma(u) \cap B) \geq (\varepsilon/(2\sigma))^{D-d}$ for all $u \in U_0$. ■

Of course, an immediate consequence of the above lemma is that, for every $M \in \mathcal{M}(\kappa)$ and every measurable set A, $C_* U_M(A) \leq Q_M(A) \leq C^* U_M(A)$. We conclude this section by recording all the assumptions in Theorems 1 and 2:

(A1) The manifold M is d-dimensional and is contained in a compact set $\mathcal{K} \subset \mathbb{R}^D$ with $d < D$.

(A2) The manifold M satisfies $\Delta(M) \geq \kappa > 0$.

(A3) The observed data $Y_1, \ldots, Y_n$ are iid observations with $Y_i = X_i + \xi_i$. Here, $\xi_1, \ldots, \xi_n$ are drawn uniformly on M. X_i given ξ_i is drawn uniformly on $L_\sigma(\xi_i) = T_{\xi_i}^{\perp} \bigcap B_D(\xi_i, \sigma)$.

(A4) The noise level σ satisfies $0 < \sigma < \kappa$.
Remark: As noted by a referee, the assumptions are very specific and the results do depend critically on the assumptions especially the assumption that d is known.
Remark: A referee has pointed out that another reasonable model is to assume that the Y_i have a uniform distribution on the tube of size σ around the manifold. To the best of our knowledge, this does not correspond to our model except in the special case where $\Delta(M) = \infty$. However, all the results of our paper still apply in this case as long as $\sigma < \kappa$.

3. Minimax Lower Bound

In this section we derive a lower bound on the minimax rate of convergence for this problem. We will make use of the following result due to LeCam (1973). The following version is from Lemma 1 of Yu (1997).

Lemma 5 (Le Cam 1973) *Let $\mathcal{Q}$ be a set of distributions. Let $\theta(Q)$ take values in a metric space with metric ρ. Let $Q_0, Q_1 \in \mathcal{Q}$ be any pair of distributions in $\mathcal{Q}$. Let $Y_1, \ldots, Y_n$ be drawn iid from some $Q \in \mathcal{Q}$ and denote the corresponding product measure by Q^n. Let $\widehat{\theta}(Y_1, \ldots, Y_n)$ be any estimator. Then*

$$\sup_{Q \in \mathcal{Q}} \mathbb{E}_{Q^n}\Big[\rho\big(\widehat{\theta}(Y_1, \ldots, Y_n), \theta(Q)\big)\Big] \geq \rho\big(\theta(Q_0), \theta(Q_1)\big)\ ||Q_0^n \wedge Q_1^n||.$$

To get a useful bound from Le Cam's lemma, we need to construct an appropriate pair Q_0 and Q_1. This is the topic of the next subsection.

3.1 A Geometric Construction

In this section, we construct a pair of manifolds $M_0, M_1 \in \mathcal{M}(\kappa)$ and corresponding distributions Q_0, Q_1 for use in Le Cam's lemma. An informal description is as follows. Roughly speaking, M_0 and M_1 minimize the Hellinger distance $h(Q_0, Q_1)$ subject to their Hausdorff distance $H(M_0, M_1)$ being equal to a given value γ.

Let

$$M_0 = \Big\{(u_1, \ldots, u_d, 0, \ldots, 0) :\ -1 \leq u_j \leq 1,\ 1 \leq j \leq d\Big\}$$

be a d-dimensional hyperplane in $\mathbb{R}^D$. Hence $\Delta(M_0) = \infty$. Place a hypersphere of radius κ below M_0. Push the sphere upwards into M_0 causing a bump of height γ at the origin. This creates a new manifold M_0' such that $H(M_0, M_0') = \gamma$. However, M_0' is not smooth. We will roll a sphere of radius κ around M_0' to get a smooth manifold M_1 as in Figure 3. We re-iterate that this is only an informal description and the reader should see Section 7.2 for the formal details.

Theorem 6 *Let γ be a small positive number. Let M_0 and M_1 be as defined in Section 7.2. Let Q_i be the corresponding distributions on $M_i \oplus \sigma$ for $i = 0, 1$. Then:*

1. $\Delta(M_i) \geq \kappa$, $i = 0, 1$.
2. $H(M_0, M_1) = \gamma$.
3. $\int |q_0 - q_1| = O(\gamma^{(d+2)/2})$.

Proof See Section 7.2. ■

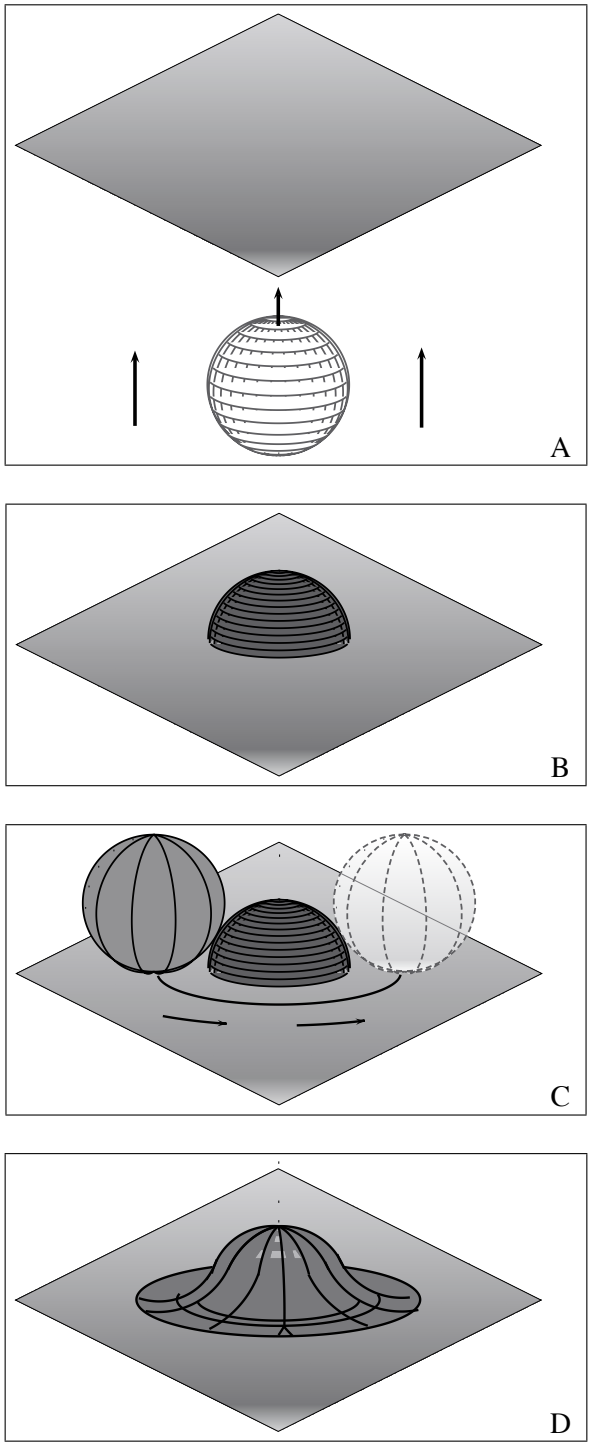


Figure 3: A sphere of radius κ is pushed upwards into the plane M_0 (panel A). The resulting manifold M_0' is not smooth (panel B). A sphere is then rolled around the manifold (panel C) to produce a smooth manifold M_1 (panel D).

3.2 Proof of the Lower Bound

Now we are in a position to prove the first theorem. Let us first restate the theorem.

Theorem 1. *Under assumptions (A1)-(A4), there is a constant $C > 0$ such that, for all large n,*

$$\inf_{\widehat{M}} \sup_{Q \in \mathcal{Q}} \mathbb{E}_Q \left[H(\widehat{M}, M) \right] \geq C n^{-\frac{2}{2+d}}$$

where the infimum is over all estimators $\widehat{M}$.

Proof of Theorem 1. Let M_0 and M_1 be as defined in Section 3.1. Let Q_i be the uniform distribution on $M_i \oplus \sigma$, $i = 0, 1$. Let q_i be the density of Q_i with respect to Lebesgue measure ν_D, $i = 0, 1$. Then, from Theorem 6, $H(M_0, M_1) = \gamma$ and $\int |q_0 - q_1| = O(\gamma^{(d+2)/2})$. Le Cam's lemma then gives, for any $\widehat{M}$,

$$\sup_{Q \in \mathcal{Q}} \mathbb{E}_{Q^n}[H(M, \widehat{M})] \geq H(M_0, M_1) \, ||Q_0^n \wedge Q_1^n|| \geq \frac{\gamma}{2}(1 - c\gamma^{(d+2)/2})^{2n}$$

where we used Equation (2). Setting $\gamma = n^{-2/(d+2)}$ yields the result. ■

4. Upper bound

To establish the upper bound, we will construct an estimator that achieves the appropriate rate. The estimator is intended only for the theoretical purpose of establishing the rate. (A simpler but non-optimal method is discussed in Section 5.) Recall that $\mathcal{M} = \mathcal{M}(\kappa)$ is the set of all d-dimensional submanifolds M contained in $\mathcal{K}$ such that $\Delta(M) \geq \kappa > 0$. Before proceeding, we need to discuss sieve maximum likelihood.

4.1 Sieve Maximum Likelihood

Let $\mathcal{P}$ be any set of distributions such that each $P \in \mathcal{P}$ has a density p with respect to Lebesgue measure ν_D. Recall that h denotes Hellinger distance. A set of pairs of functions $\mathcal{B} = \{(\ell_1, u_1), \ldots, (\ell_N, u_N)\}$ is an ε-Hellinger bracketing for $\mathcal{P}$ if, (i) for each $p \in \mathcal{P}$ there is a $(\ell, u) \in \mathcal{B}$ such that $\ell(y) \leq p(y) \leq u(y)$ for all y and (ii) $h(\ell, u) \leq \varepsilon$. The logarithm of the size of the smallest ε-bracketing is called the *bracketing entropy* and is denoted by $\mathcal{H}_{[]}(\varepsilon, \mathcal{P}, h)$.

We will make use of the following result which is Example 4 of Shen and Wong (1995).

Theorem 7 (Shen and Wong, 1995) *Let ε_n solve the equation $\mathcal{H}_{[]}(\varepsilon_n, \mathcal{P}, h) = n\varepsilon_n^2$. Let $(\ell_1, u_1), \ldots, (\ell_N, u_N)$ be an ε_n bracketing where $N = \mathcal{H}_{[]}(\varepsilon_n, \mathcal{P}, h)$. Define the set of densities $S_n^* = \{p_1^*, \ldots, p_N^*\}$ where $p_t^* = u_t / \int u_t$. Let $\widehat{p}^*$ maximize the likelihood $\prod_{i=1}^n p_t^*(Y_i)$ over the set S_n^*. Then*

$$\sup_{P \in \mathcal{P}} P^n \left(\{ h(p, \widehat{p}^*) \geq \varepsilon_n \} \right) \leq c_1 e^{-c_2 n \varepsilon_n^2}.$$

The sequence $\{S_n^*\}$ in Theorem 7 is called a *sieve* and the estimator $\widehat{p}^*$ is called a *sieve-maximum likelihood estimator*. The estimator $\widehat{p}^*$ need not be in $\mathcal{P}$. We will actually need an estimator that is contained in $\mathcal{P}$. We may construct one as follows. Let $\widehat{p}^*$ be the sieve mle corresponding to S_n^*. Then $\widehat{p}^* = p_t^*$ for some t. Let $(\widehat{\ell}, \widehat{u}) \equiv (\ell_t, u_t)$ be the corresponding bracket.

Lemma 8 *Assume the conditions in Theorem 7. Let $\widehat{p}$ be any density in $\mathcal{P}$ such that $\widehat{\ell} \leq \widehat{p} \leq \widehat{u}$. If $\varepsilon_n \leq 1$ then*

$$\sup_{P \in \mathcal{P}} P^n \left(\{ h(p, \widehat{p}) \geq c\varepsilon_n \} \right) \leq c_1 e^{-c_2 n \varepsilon_n^2}.$$

Proof By the triangle inequality, $h(p,\widehat{p}) \leq h(p,\widehat{p}^*) + h(\widehat{p},\widehat{p}^*) = h(p,\widehat{p}^*) + h(\widehat{p},u_t/\int u_t)$ where $\widehat{p}^* = u_t/\int u_t$ for some t. From Theorem 7, $h(p,\widehat{p}^*) \leq \varepsilon_n$ with high probability. Thus we need to show that $h(\widehat{p},u_t/\int u_t) \leq C\varepsilon_n$. It suffices to show that, in general, $h(p,u/\int u) \leq C\,h(\ell,u)$ whenever $\ell \leq p \leq u$.

Let (ℓ,u) be a bracket and let $\delta^2 = h^2(\ell,u) \leq 1$. Let $\ell \leq p \leq u$. We claim that $h^2(p,u/\int u) \leq 4\delta^2$. (Taking $\delta = \varepsilon_n$ then proves the result.) Let $c^2 = \int u$. Then $1 \leq c^2 = \int u = \int p + \int(u-p) = 1 + \int(u-p) = 1 + \ell_1(u,p) \leq 1 + 2h(u,\ell) = 1 + 2\delta$. Now,

$$\begin{aligned} h^2\left(p, \frac{u}{\int u}\right) &= \int(\sqrt{u}/c - \sqrt{p})^2 = \frac{1}{c^2}\int(\sqrt{u} - c\sqrt{p})^2 \leq \int(\sqrt{u} - c\sqrt{p})^2 \\ &= \int((\sqrt{u}-\sqrt{p}) + (c-1)\sqrt{p})^2 \leq 2\int(\sqrt{u}-\sqrt{p})^2 + 2(c-1)^2 \\ &\leq 2\delta^2 + 2(\sqrt{1+2\delta}-1)^2 \leq 2\delta^2 + 2\delta^2 = 4\delta^2 \end{aligned}$$

where the last inequality used the fact that $\delta \leq 1$. ■

In light of the above result, we define modified maximum likelihood sieve estimator $\widehat{p}$ to be any $p \in \mathcal{P}$ such that $\widehat{\ell} \leq \widehat{p} \leq \widehat{u}$. For simplicity, in the rest of the paper, we refer to the modified sieve estimator $\widehat{p}$, simply as the maximum likelihood estimator (mle).

4.2 Outline of Proof

We are now ready to find an estimator $\widehat{M}$ that converges at the optimal rate (up to logarithmic terms.) Our strategy for estimating M has the following steps:

Step 1. We split the data into two halves.

Step 2. Let $\widetilde{Q}$ be the maximum likelihood estimator using the first half of the data. Define $\widetilde{M}$ to be the corresponding manifold. We call $\widetilde{M}$, the pilot estimator. We show that $\widetilde{M}$ is a consistent estimator of M that converges at a sub-optimal rate $a_n = n^{-\frac{2}{D(d+2)}}$. To show this we:

- *a*. Compute the Hellinger bracketing entropy of $\mathcal{Q}$. (Theorem 9, Lemmas 10 and 11).
- *b*. Establish the rate of convergence of the mle in Hellinger distance, using the bracketing entropy and Theorem 7.
- *c*. Relate the Hausdorff distance to the Hellinger distance and hence establish the rate of convergence a_n of the mle in Hausdorff distance. (Lemma 13).
- *d*. Conclude that the true manifold is contained, with high probability, in $\mathcal{M}_n = \{M \in \mathcal{M}(\kappa) :\ H(M,\widetilde{M}) \leq a_n\}$ (Lemma 14). Hence, we can now restrict attention to $\mathcal{M}_n$.

Step 3. To improve the pilot estimator, we need to control the relationship between Hellinger and Hausdorff distance and thus need to work over small sets on which the manifold cannot vary too greatly. Hence, we cover the pilot estimator with long, thin slabs $R_1,\ldots,R_N$. We do this by first covering $\widetilde{M}$ with spheres $\beth_1,\ldots,\beth_N$ of radius $\delta_n = O((\log n/n)^{1/(2+d)})$. We define a slab R_j to be the union of fibers of size $b = \sigma + a_n$ within one of the spheres: $R_j = \cup_{x\in\beth_j} L_b(x,\widetilde{M})$. We then show that:

- *a*. The set of fibers on $\widetilde{M}$ cover each $M \in \mathcal{M}_n$ in a nice way. In particular, if $M \in \mathcal{M}_n$ then each fiber from $\widetilde{M}$ is nearly normal to M. (Lemma 15).

b. As M cuts through a slab, it stays nearly parallel to $\widetilde{M}$. Roughly speaking, M behaves like a smooth, nearly linear function within each slab. (Lemma 16).

Step 4. Using the second half of the data, we apply maximum likelihood within each slab. This defines estimators $\widehat{M}_j$, for $1 \le j \le N$. We show that:

a. The entropy of the set of distributions within a slab is very small. (Lemma 18).

b. Because the entropy is small, the maximum likelihood estimator within a slab converges fairly quickly in Hellinger distance. The rate is $\varepsilon_n = (\log n/n)^{1/(2+d)}$. (Lemma 19).

c. Within a slab, there is a tight relationship between Hellinger distance and Hausdorff distance. Specifically, $H(M_1, M_2) \le c\, h^2(Q_1, Q_2)$. (Lemma 20).

d. Steps (4b) and (4c) imply that $H(M \cap R_j, \widehat{M}_j) = O_P(\varepsilon_n^2) = O_P((\log n/n)^{2/(d+2)})$.

Step 5. Finally we define $\widehat{M} = \bigcup_{j=1}^{N} \widehat{M}_j$ and show that $\widehat{M}$ converges at the optimal rate because each $\widehat{M}_j$ does within its own slab.

The reason for getting a preliminary estimator and then covering the estimator with thin slabs is that, within a slab, there is a tight relationship between Hellinger distance and Hausdorff distance. This is not true globally but only in thin slabs. Maximum likelihood is optimal with respect to Hellinger distance. Within a slab, this allows us to get optimal rates in Hausdorff distance.

4.3 Step 1: Data Splitting

For simplicity assume the sample size is even and denote it by $2n$. We split the data into two halves which we denote by $X = (X_1, \ldots, X_n)$ and $Y = (Y_1, \ldots, Y_n)$.

4.4 Step 2: Pilot Estimator

Let $\widetilde{q}$ be the maximum likelihood estimator over $\mathcal{Q}$. Let $\widetilde{M}$ be the corresponding manifold. To study the properties of $\widetilde{M}$ requires two steps: computing the bracketing entropy of $\mathcal{Q}$ and relating $H(M, \widetilde{M})$ to $h(q, \widetilde{q})$. The former allows us to apply Theorem 7 to bound $h(q, \widetilde{q})$, and the latter allows us to control the Hausdorff distance.

4.5 Step 2a: Computing the Entropy of $\mathcal{Q}$

To compute the entropy of $\mathcal{Q}$ we start by constructing a finite net of manifolds to cover $\mathcal{M}(\kappa)$. A finite set of d-manifolds $\mathbb{M}_\gamma = \{M_1, \ldots, M_N\}$ is a γ-net (or a γ-cover) if, for each $M \in \mathcal{M}$ there exists $M_j \in \mathbb{M}_\gamma$ such that $H(M, M_j) \le \gamma$. Let $N(\gamma) = N(\gamma, \mathcal{M}, H)$ be the size of the smallest covering set, called the (Hausdorff) covering number of $\mathcal{M}$.

Theorem 9 *The Hausdorff covering number of $\mathcal{M}$ satisfies the following:*

$$N(\gamma) \equiv N(\gamma, \mathcal{M}, H) \le c_1\, \kappa_2(\kappa, d, D) \exp\left(\kappa_3(\kappa, d, D)\, \gamma^{-d/2}\right) \equiv c \exp\left(c' \gamma^{-d/2}\right)$$

where $\kappa_2(\kappa, d, D) = \binom{D}{d}^{(c_2/\kappa)^D}$ *and* $\kappa_3(\kappa, d, D) = 2^{d/2}(D-d)(c_2/\kappa)^D$, *for a constant* c_2 *that depends only on* κ *and* d.

Proof Recall that the manifolds in $\mathcal{M}$ all lie within $\mathcal{K}$. Consider any hypercube containing $\mathcal{K}$. Divide this cube into a grid of $J = (2c/\kappa)^D$ sub-cubes $\{C_1, \ldots, C_J\}$ of side length κ/c, where $c \geq 4$ is a positive constant chosen to be sufficiently large. Our strategy is to show that within each of these cubes, the manifold is the graph of a smooth function. We then only need count the number of such smooth functions.

In thinking about the manifold as (locally) the graph of a smooth function, it helps to be able to translate easily between the natural coordinates in $\mathcal{K}$ and the domain-range coordinates of the function. To that end, within each subcube C_j for $j \in \{1, \ldots, J\}$, we define $K = \binom{D}{d}$ coordinate frames, F_{jk} for $k \in \{1, \ldots, K\}$, in which d out of D coordinates are labeled as "domain" and the remaining $D-d$ coordinates are labeled as "range."

Each frame is associated with a relabeling of the coordinates so that the d "domain" coordinates are listed first and $D-d$ "range" coordinates last. That is, F_{jk} is defined by a one-to-one correspondence between $x \in C_j$ and $(u,v) \in \pi_{jk}(x)$ where $u \in \mathbb{R}^d$ and $v \in \mathbb{R}^{D-d}$ and $\pi_{jk}(x_1, \ldots, x_D) = (x_{i_1}, \ldots, x_{i_d}, x_{j_1}, \ldots, x_{j_{D-d}})$ for domain coordinate indices $i_1 < \ldots < i_d$ and range coordinate indices $j_1 < \ldots < j_{D-d}$.

We define $\text{domain}(F_{jk}) = \{u \in \mathbb{R}^d : \exists v \in \mathbb{R}^{D-d} \text{ such that } (u,v) \in F_{jk}\}$, and let $\mathcal{G}_{jk}$ denote the class of functions defined on $\text{domain}(F_{jk})$ whose second derivative (i.e., second fundamental form) is bounded above by a constant $C(\kappa)$ that depends only on κ. To say that a set $R \subset C_j$ is the graph of a function on a d-dimensional subset of the coordinates in C_j is equivalent to saying that for some frame F_{jk} and some set $A \subset \text{domain}(F_{jk})$, $R = \pi_{jk}^{-1}\{(u, f(u)) : u \in A\}$.

We will prove the theorem by establishing the following claims.

Claim 1. Let $M \in \mathcal{M}$ and C_j be a subcube that intersects M. Then: (i) for at least one $k \in \{1, \ldots, K\}$, the set $M \cap C_j$ is the graph of a function (i.e., single-valued mapping) defined on a set $\mathcal{A} \subset \text{domain}(F_{jk})$, of the form $(u_1, \ldots, u_d) \mapsto \pi_{jk}^{-1}((u, f(u)))$ for some function f on $\mathcal{A}$, and (ii) this function lies in $\mathcal{G}_{jk}$.

Claim 2. $\mathcal{M}$ is in one-to-one correspondence with a subset of $\mathcal{G} = \prod_{j=1}^{J} \bigcup_{k=1}^{K} \mathcal{G}_{jk}$.

Claim 3. The L^∞ covering number of $\mathcal{G}$ satisfies

$$N(\gamma, \mathcal{G}, L^\infty) \leq c_1 \binom{D}{d}^{(2c/\kappa)^D} \exp\left((D-d)(2c/\kappa)^D \gamma^{-d/2}\right).$$

Claim 4. There is a one-to-one correspondence between an $\gamma/2$ L^∞-cover of $\mathcal{G}$ and an γ Hausdorff-cover of $\mathcal{M}$.

Taken together, the claims imply that

$$N(\gamma, \mathcal{M}, H) \leq c_1 \binom{D}{d}^{(2c/\kappa)^D} \exp((D-d)(2c/\kappa)^D 2^{d/2} \gamma^{-d/2}).$$

Taking $c_2 = 2c$ proves the theorem.

Proof of Claim 1. We begin by showing that (i) implies (ii). By part 1 of Lemma 3, each $M \in \mathcal{M}$ has curvature (second fundamental form) bounded above by $1/\kappa$. This implies that the function identified in (i) has uniformly bounded second derivative and thus lies in the corresponding $\mathcal{G}_{jk}$.

We prove (i) by contradiction. Suppose that there is an $M \in \mathcal{M}$ such that for every j with $M \cap C_j \neq \emptyset$, the set $M \cap C_j$ is not the graph of a single-valued mapping for any of the K coordinate frames.

Fix $j \in \{1,\ldots,J\}$. Then in each domain(F_{jk}), there is a point u such that $C_j \cap \pi_{jk}^{-1}(u \times \mathbb{R}^{D-d})$ intersects M in at least two points, call them a_k and b_k. By construction $\|a_k - b_k\| \leq \sqrt{D-d} \cdot \kappa/c$, and hence by choosing c large enough (making the cubes small), part 3 of Lemma 3 tells us that $d_M(a_k, b_k) \leq 2\sqrt{D-d}\kappa/c$. Then we argue as follows:

1. By parts 2 and 3 of Lemma 3 and the fact that C_j has diameter $\sqrt{D}\kappa/c$ and

$$\max_{p,q \in C_j \cap M} \cos(\mathsf{angle}(T_pM, T_qM)) \geq 1 - \frac{2\sqrt{D}}{c}.$$

   For large enough c, the maximum angle between tangent vectors can be made smaller than $\pi/3$.

2. By part 2 of Lemma 3, any point z along a geodesic between a_k and b_k,

$$\cos(\mathsf{angle}(T_{a_k}M, T_zM)) \geq 1 - \frac{2\sqrt{D-d}}{c}.$$

   It follows that there is a point in $C_j \cap M$ and a tangent vector v_k at that point such that $\mathsf{angle}(v_k, b_k - a_k) = O(1/\sqrt{c})$.

3. We have for each of $K = \binom{D}{d}$ coordinate frames and associated tangent vectors $v_1, \ldots, v_K$ that are each nearly orthogonal to at least d of the others. Consequently, there are $\geq d+1$ nearly orthogonal tangent vectors of M within C_j. This contradicts point 1 and proves the claim.

Proof of Claim 2. We construct the correspondence as follows. For each cube C_j, let k_j^* be the smallest k such that $M \cap C_j$ is the graph of a function $\phi_{jk} \in \mathcal{G}_{jk}$ as in Claim 1. Map M to $\varphi = (\phi_{1k_1^*}, \ldots, \phi_{Jk_J^*})$, and let $\mathcal{F} \subset \mathcal{G}$ be the image of this map. If $M \neq M' \in \mathcal{M}$, then the corresponding φ and φ' must be distinct. If not, then $M \cap C_j = M' \cap C_j$ for all j, contradicting $M \neq M'$. The correspondence from $\mathcal{M}$ to $\mathcal{F}$ is thus a one-to-one correspondence.

Proof of Claim 3. From the results in Birman and Solomjak (1967), the set of functions defined on a pre-compact d-dimensional set that take values in a fixed dimension space $\mathbb{R}^m$ with uniformly bounded second derivative has L^∞ covering number bounded above by $c_1 e^{m(1/\gamma)^{d/2}}$ for some c_1. Part 1 of Lemma 3 shows that each $M \in \mathcal{M}$ has curvature (second fundamental form) bounded above by $1/\kappa$, so each $\mathcal{G}_{jk}$ satisfies Birman and Solomjak's conditions. Hence, $N(\gamma, \mathcal{G}_{jk}, L^\infty) \leq c_1 e^{(D-d)(1/\gamma)^{d/2}}$. Because all the $\mathcal{G}_{jk}$'s are disjoint, simple counting arguments show that $N(\gamma, \mathcal{G}, L^\infty) = \left(\binom{D}{d} N(\gamma, \mathcal{G}_{jk}, L^\infty)\right)^J$, where J is the number of cubes defined above. The claim follows. (Note that the functions in Claim 1 are defined on a subset of domain(F_{jk}). But because all such functions have an extension in $\mathcal{G}_{jk}$, a covering of $\mathcal{G}_{jk}$ also covers these functions defined on restricted domains.)

Proof of Claim 4. First, note that if two functions are less than γ distant in L^∞, their graphs are less than γ distant in Hausdorff distance, and vice versa. This implies that a γ L^∞-cover of a set of functions corresponds directly to an γ Hausdorff-cover of the set of the functions' graphs. Hence, in the argument that follows, we can work with functions or graphs interchangeably.

For $k \in \{1,\ldots,K\}$, let $\mathcal{G}_{jk}^\gamma$ be a minimal L^∞ cover of $\mathcal{G}_{jk}$ by $\gamma/2$ balls; specifically, we assume that $\mathcal{G}_{jk}^\gamma$ is the set of centers of these balls. For each $g_{jk} \in \mathcal{G}_{jk}^\gamma$, define $f_{jk}(u) = \pi_{jk}^{-1}(u, g_{jk}(u))$. For every j, choose one such f_{jk}, and define a set $M' = \bigcup_j (C_j \cap \text{range}(f_{jk_j}))$, which is a union of manifolds with boundary that have curvature bounded by $1/\kappa$. That is, such an M' is piecewise

smooth (smooth within each cube) but may fail to satisfy $\Delta(M') \geq \kappa$ globally. Let $\mathcal{A}$ be the collection of M' constructed this way. There are $N(\gamma/2, \mathcal{G}, L^\infty)$ elements in this collection.

By construction and Claim 2, for each $M \in \mathcal{M}$, there exists an $M' \in \mathcal{A}$ such that $H(M, M') \leq \gamma/2$. In other words, the set of $\gamma/2$ Hausdorff balls around the manifolds in $\mathcal{A}$ covers $\mathcal{M}$ but the elements of $\mathcal{A}$ are not themselves necessarily in $\mathcal{M}$. Let $B_H(A, \gamma/2)$ denote the set of all d-manifolds $M \in \mathcal{M}$ such that $H(A, M) \leq \gamma/2$. Let

$$\mathcal{A}_0 = \Big\{ A \in \mathcal{A} : \; B_H(A, \gamma/2) \cap \mathcal{M} \neq \emptyset \Big\}.$$

For each $A \in \mathcal{A}_0$, choose some $\widetilde{A} \in B_H(A, \gamma/2) \cap \mathcal{M}$. By the triangle inequality, the set $\{\widetilde{A} : \; A \in \mathcal{A}_0\}$ forms an γ Hausdorff-net for $\mathcal{M}$. This proves the claim. ■

We are almost ready to compute the entropy. We will need the following lemma.

Lemma 10 *Let $0 < \gamma < \kappa - \sigma$. There exists a constant $K > 0$ (depending only on $\mathcal{K}, \kappa$ and σ) such that, for any $M_1, M_2 \in \mathcal{M}(\kappa)$, $H(M_1, M_2) \leq \gamma$ implies that $|V(M_1 \oplus \sigma) - V(M_2 \oplus \sigma)| \leq K\gamma$. Also, for any $M \in \mathcal{M}(\kappa)$, $|V(M \oplus (\sigma + \gamma)) - V(M \oplus \sigma)| \leq K\gamma$.*

Proof Let $S_j = M_j \oplus \sigma$, $j = 1, 2$. Then, using (3),

$$S_2 \subset M_1 \oplus (\sigma + \gamma) = \bigcup_{u \in M_1} L_{\sigma+\gamma}(u).$$

Hence, uniformly over $\mathcal{M}$,

$$V(S_2) \leq \int_{M_1} \nu_{D-d}(L_{\sigma+\gamma}(u)) d\mu_{M_1} \leq \int_{M_1} \nu_{D-d}(L_\sigma(u)) d\mu_{M_1} + K\gamma = V(S_1) + K\gamma$$

since $\nu_{D-d}(B(u, \sigma + \gamma)) \leq \nu_{D-d}(B(u, \sigma)) + K\gamma$ for some $K > 0$ not depending on M_1 or M_2. By a symmetric argument, $V(S_1) \leq V(S_2) + K\gamma$. Hence, $|V(M_1 \oplus \sigma) - V(M_2 \oplus \sigma)| \leq K\gamma$. The second statement is proved in a similar way. ■

Now we construct a Hellinger bracketing. Let $\gamma = \varepsilon^2$. Let $\mathbb{M}_\gamma = \{M_1, \ldots, M_N\}$ be a γ-Hausdorff net of manifolds. Thus, by Theorem 9, $N = N(\varepsilon^2, \mathcal{M}, H) \leq c_1 e^{c_2 (1/\varepsilon)^d}$. Let ω denote the volume of a sphere of radius σ. Let q_j be the density corresponding to M_j. Define

$$u_j(y) = \left(q_j(y) + \frac{2\varepsilon^2}{V(M_j \oplus (\sigma + \varepsilon^2))} \right) I(y \in M_j \oplus (\sigma + \varepsilon^2))$$

and

$$\ell_j(y) = \left(q_j(y) - \frac{2\varepsilon^2}{V(M_j \oplus (\sigma - \varepsilon^2))} \right) I(y \in M_j \oplus (\sigma - \varepsilon^2)).$$

Let $\mathcal{B} = \{(\ell_1, u_1), \ldots, (\ell_N, u_N)\}$.

Lemma 11 *$\mathcal{B}$ is an ε-Hellinger bracketing of Q. Hence, $\mathcal{H}_{[]}(\varepsilon, Q, h) \leq C(1/\varepsilon)^d$.*

Proof Let $M \in \mathcal{M}(\kappa)$ and let $Q = Q_M$ be the corresponding distribution. Let q be the density of Q. Q is supported on $S = M \oplus \sigma$. There exists $M_j \in \mathbb{M}_\gamma$ such that $H(M, M_j) \leq \varepsilon^2$. Let y be in S. Then there is a $x \in M$ such that $||y-x|| \leq \sigma$. There is a $x' \in M_j$ such that $||x-x'|| \leq \varepsilon^2$. Hence, $d(y, M_j) \leq \sigma + \varepsilon^2$ and thus y is in the support of u_j. Now, for $y \in S$, $u_j(y) - q(y) = 2\varepsilon^2 / V(M_j \oplus (\sigma + \varepsilon^2)) \geq 0$. Hence, $q(y) \leq u_j(y)$. By a similar argument, $\ell_j(y) \leq q(y)$. Thus $\mathcal{B}$ is a bracketing. Now

$$\ell_1(\ell_j, u_j) \quad = \quad \int u_j - \int \ell_j = \left(1 + \frac{2K\varepsilon^2}{\omega}\right) - \left(1 - \frac{2K\varepsilon^2}{\omega}\right) = \frac{4K\varepsilon^2}{\omega}.$$

Finally, by (1), $h(u_j, \ell_j) \leq \sqrt{\ell_1(\ell_j, u_j)} = C\varepsilon$. Thus $\mathcal{B}$ is a $C\varepsilon$-Hellinger bracketing. ■

4.6 Step 2b. Hellinger Rate

Lemma 12 *Let $\widetilde{Q}$ be the mle. Then*

$$\sup_{Q \in \mathcal{Q}} Q^n \left(\left\{ h(Q, \widetilde{Q}) > C_0 n^{-\frac{1}{d+2}} \right\} \right) \leq \exp\left\{ -C n^{\frac{d}{2+d}} \right\}.$$

Proof We have shown (Lemma 11) that $\mathcal{H}_{[]}(\varepsilon, \mathcal{Q}, h) \leq C(1/\varepsilon)^d$. Solving the equation $H_{[]}(\varepsilon_n, \mathcal{Q}, h) = n\varepsilon_n^2$ from Theorem 7 we get $\varepsilon_n = (1/n)^{1/(d+2)}$. From Lemma 8, for all Q

$$Q^n \left(\left\{ h(Q, \widetilde{Q}) > C_0 n^{-\frac{1}{d+2}} \right\} \right) \leq c_1 e^{-c_2 n \varepsilon_n^2} = \exp\left\{ -C n^{\frac{d}{2+d}} \right\}.$$

■

4.7 Step 2c. Relating Hellinger Distance and Hausdorff Distance

Lemma 13 *Let $c = (\kappa - \sigma)\sqrt{\pi} C_* / (2\Gamma(D/2+1))$. If $M_1, M_2 \in \mathcal{M}(\kappa)$ and $h(Q_1, Q_2) < c$ then*

$$H(M_2, M_2) \leq \left[\frac{2}{\sqrt{\pi}} \left(\frac{\Gamma(D/2+1)}{C_*} \right)^{1/D} \right] h^{\frac{1}{D}}(Q_1, Q_2)$$

Proof Let $b = H(M_1, M_2)$ and $\gamma = \min\{\kappa - \sigma, b\}$. Let S_1, S_2 be the supports of Q_1 and Q_2. Because $H(M_1, M_2) = b$, we can find points $x \in M_1$ and $y \in M_2$ such that $\|y - x\| = b$. Note that $T_x M_1$ and $T_y M_2$. are parallel, otherwise we could move x or y and increase $\|y - x\|$. It follows that the line segment $[x, y]$ is along a common normal vector of the two manifolds and we can write $y = x \pm bu$ for some $u \in L_\sigma(u, M)$. Without loss of generality, assume that $y = x + bu$. Let $x' = x + \sigma u$ and $y' = y + \sigma u$. Hence, $x' \in \partial S_1$, $y' \in \partial S_2$ and $||x' - y'|| = b$. Note that ∂S_1 and ∂S_2 are themselves smooth D-manifolds with $\Delta(\partial S_i) \geq \kappa - \sigma > 0$.

We now make the following three claims:

1. $y' \in S_2 - S_1$.
2. $(x', y'] \subset S_2 - S_1$
3. interior $B\left(\frac{x'+y'}{2}, \frac{\gamma}{2}\right) \subset S_2 - S_1$

First, note that y' differs from y along a fiber of M_2 by exactly σ, therefore $[x',y'] \subset S_2$. Second, because $x' \in \partial S_1$, there is a neighborhood of x' in $[x',y']$ that is not contained in S_1. Hence, if there is a point in $S_1 \cap [x',y']$ there must be a point $z' \in \partial S_1 \cap [x',y']$, with $z' \neq x'$. This implies the existence of two distinct points whose fibers of length less than $\kappa - \sigma$ cross, which contradicts the fact that $\Delta(\partial S_1) \geq \kappa - \sigma$. Claims 1 and 2 follows.

Let $B = B\left(\frac{x'+y'}{2}, \frac{\gamma}{2}\right)$. By construction, B is tangent to ∂S_1 at x' and tangent to ∂S_2 at y', and B contains $[x',y']$. The ball has radius $\gamma/2 = (1/2)\min\{\kappa - \sigma, b\} < \kappa - \sigma$. Because B intersects $S_2 - S_1$, the interior of B cannot intersect either ∂S_1 or ∂S_2. Claim 3 follows by a similar argument as in the proof of Claim 2. (In particular, if there were a point in the interior of B that is either in S_1 or outside S_2, a line segment from $(x'+y')/2$ to that point would have to intersect the corresponding boundary, which cannot happen.)

Now $V(B) = (\gamma/2)^D \pi^{D/2}/\Gamma(D/2+1)$. So

$$\begin{aligned} h(Q_1,Q_2) &\geq \ell_1(Q_1,Q_2) = \int |q_1 - q_2| \geq \int_{S_1 \cap S_2^c} |q_1 - q_2| \\ &= \int_{S_1 \cap S_2^c} q_1 = Q_1(S_1 \cap S_2^c) \geq C_* V(S_1 \cap S_2^c) = C_* (\gamma/2)^D \pi^{D/2}/\Gamma(D/2+1). \end{aligned}$$

Hence,

$$\gamma = \min\{\kappa - \sigma, b\} \leq \left[\frac{2}{\sqrt{\pi}} \left(\frac{\Gamma(D/2+1)}{C_*}\right)^{1/D}\right] h^{1/D}(Q_1,Q_2).$$

If $\kappa - \sigma \leq b$ this implies that $h(Q_1,Q_2) > c$ which contradicts the assumption that $h(Q_1,Q_2) < c$. Therefore, $\gamma = b$ and the conclusion follows. ■

4.8 Step 2d. Computing The Hausdorff Rate of the Pilot

Lemma 14 *Let* $a_n = \left(\frac{C_0}{n}\right)^{\frac{2}{D(d+2)}}$. *For all large n,*

$$\sup_{Q \in \mathcal{Q}} Q^n \left(\{H(M,\widetilde{M}) > a_n\}\right) \leq \exp\left\{-Cn^{\frac{d}{2+d}}\right\}.$$

Proof Follows by combining Lemma 12 and Lemma 13. ■

We conclude that, with high probability, the true manifold M is contained in the set $\mathcal{M}_n = \left\{M \in \mathcal{M}(\kappa) : H(\widetilde{M},M) \leq a_n\right\}$.

4.9 Step 3: Cover With Slabs

Now we cover the pilot estimator $\widetilde{M}$ with (possibly overlapping) slabs. Let $\delta_n = \left(\frac{C\log n}{n}\right)^{\frac{1}{2+d}}$. It follows from part 6 of Lemma 3 that there exists a collection of points $F = \{x_1,\ldots,x_N\} \subset \widetilde{M}$, such that $N = (c\delta_n)^{-d} = (Cn/\log n)^{d/(2+d)}$ and such that $\widetilde{M} \subset \bigcup_{j=1}^N B_D(x_j, c\delta)$.

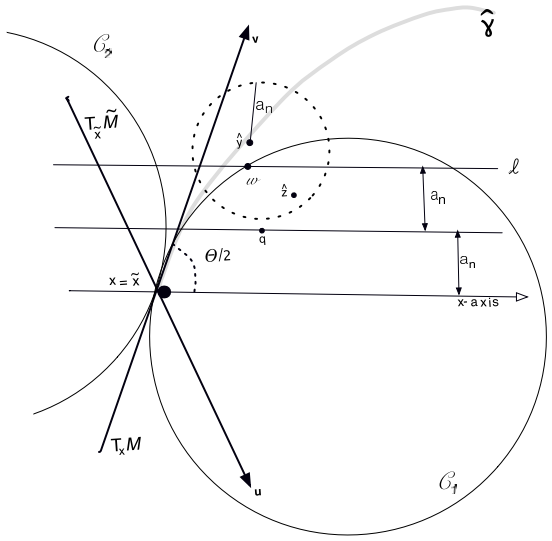


Figure 4: Figure for the proof of part 1 of Lemma 15.

4.10 Step 3a. The Fibers of $\widetilde{M}$ Cover M Nicely

Lemma 15 *Let $b = \sigma + a_n$. For $\widetilde{x} \in \widetilde{M}$, let $L_b(\widetilde{x}) = T_{\widetilde{x}}^{\perp}\widetilde{M} \cap B_D(\widetilde{x}, b)$ be a fiber at $\widetilde{x}$ of size b. Let $M \in \mathcal{M}_n$. Then:*

1. *If $\widetilde{x} \in \widetilde{M}$ and $x \in M$ are such that $\|x - \widetilde{x}\| \leq a_n$, then* $\mathsf{angle}(T_x M, T_{\widetilde{x}}\widetilde{M}) < \pi/4$.
2. *$L_b(\widetilde{x}) \cap M \neq \emptyset$.*
3. *If $x \in L_b(\widetilde{x}) \cap M$, then $\|x - \widetilde{x}\| \leq 2a_n$.*
4. *For any $\widetilde{x} \in \widetilde{M}$, $\#\{L_b(\widetilde{x}) \cap M\} = 1$.*
5. *We have $M \subset \bigcup_{\widetilde{x} \in \widetilde{M}} L_b(\widetilde{x})$.*

Proof *1.* Let x and $\widetilde{x}$ be as given in the statement of the lemma and let $\theta = \mathsf{angle}(T_x M, T_{\widetilde{x}}\widetilde{M})$. Suppose that $\theta \geq \pi/4$. There exists unit vectors $u \in T_{\widetilde{x}}\widetilde{M}$ and $v \in T_x M$ such that $\mathsf{angle}(u, v) = \theta$. Without loss of generality, we can assume that $x = \widetilde{x}$. (The extension to the case $x \neq \widetilde{x}$ is straightforward.)

Consider the plane defined by u and v as in Figure 4. We assume, without loss of generality, that $(u+v)/2$ generates the x-axis in this plane and that v lies above the x-axis and u lies below the x axis. Let ℓ denote the horizontal line, parallel to the x-axis and lying $2a_n$ units above the horizontal axis. Hence, u and v each make an angle greater than $\pi/8$ with respect to the x-axis.

Consider the two circles $\mathcal{C}_1$ and $\mathcal{C}_2$ tangent to M at x with radius κ where $\mathcal{C}_1$ lies below v and $\mathcal{C}_2$ lies above v. Let w be the point at which $\mathcal{C}_1$ intersects ℓ. The arclength of $\mathcal{C}_1$ from x to w is Ca_n for some $C > 1$. Let γ be the geodesic on M through x with gradient v. The projection $\widehat{\gamma}$ of γ into the plane must fall between $\mathcal{C}_1$ and $\mathcal{C}_2$. Let $y = \gamma(Ca_n)$ and $\widehat{y}$ be the projection of y into the plane.

Now $||y - \widetilde{x}|| \geq ||\widehat{y} - \widetilde{x}|| \geq ||w - \widetilde{x}|| \geq 2a_n > a_n$. There exists $\widetilde{z} \in \widetilde{M}$ such that $||\widetilde{z} - y|| \leq a_n$. Hence, $||\widehat{z} - \widehat{y}|| \leq a_n$ where $\widehat{z}$ is the projection of $\widetilde{z}$ into the plane. Let q be the point on the plane

with coordinates $(a_n\sqrt{C^2-1}, a_n)$. Thus, $||q-\widetilde{x}|| = C\,a_n$. Note that $\mathsf{angle}(\widehat{z}-\widetilde{x}, u)$ is larger than the angle between $q-\widetilde{x}$ and the x-axis which is $\arctan\left(\frac{1}{\sqrt{C^2-1}}\right) \equiv \alpha > 0$. Hence,

$$\mathsf{angle}(\widetilde{z}-\widetilde{x}, u) \geq \mathsf{angle}(\widehat{z}-\widetilde{x}, u) \geq \alpha.$$

Let $\widetilde{\gamma}$ be a geodesic on $\widetilde{M}$, parameterized by arclength connecting $\widetilde{x}$ and $\widetilde{z}$. Thus $\widetilde{\gamma}(0) = \widetilde{x}$ and $\widetilde{\gamma}(T) = \widetilde{z}$ for some T. There exists some $0 \leq t \leq T$ such that $\gamma'(t) \propto \widetilde{z}-\widetilde{x}$. So

$$\mathsf{angle}(\gamma'(t), \gamma'(0)) = \alpha > 0.$$

However, $||\widetilde{z}-\widetilde{x}|| \leq (C+1)\,a_n$ which implies, by part 2 of Lemma 3, that $\mathsf{angle}(\gamma'(t), \gamma'(0)) = O(\sqrt{a_n}) < \alpha$ which is a contradiction.

2. For any $\widetilde{x} \in \widetilde{M}$, the closest point $x \in M$ must satisfy $\|x-\widetilde{x}\| \leq a_n$. Let y be the projection of x onto $T_{\widetilde{x}}\widetilde{M}$. Let $U = T_{\widetilde{x}}\widetilde{M} \cap B_d(y, a_n)$. Let $\text{Cyl} = \bigcup_{u\in U} B_D(u, 3a_n) \cap \left(T_{\widetilde{x}}\widetilde{M}\right)^{\perp}$. Cyl is a small hyper-cylinder containing y and $\widetilde{x}$, with the former in the center. M cannot intersect the top or bottom faces of the cylinder. Otherwise, we can find a point $p \in M$ such that $\mathsf{angle}(T_{\widetilde{x}}\widetilde{M}, T_pM) > \arctan(1) = \pi/4$ contradicting *1*. Thus, any path through x on M must intersect the sides of Cyl. Hence, $L_b(\widetilde{x}) \cap M \neq \emptyset$.

3. Let $x \in M \cap L_b(\widetilde{x})$. Suppose that $||x-\widetilde{x}|| > 2a_n$. There exists $q \in \widetilde{M}$ such that $||q-x|| \leq a_n$. Note that $||q-\widetilde{x}|| > a_n$. Now we apply part 5 Lemma 3 with $p = \widetilde{x}$ and $v = x$. This implies that $||v-p|| = ||x-\widetilde{x}|| < a_n^2/\kappa$ which contradicts the assumption that $||x-\widetilde{x}|| > 2a_n$.

4. Suppose that more than one point of M were in $L_b(\widetilde{x})$. Pick two and call them x_1 and x_2. By *3*, $\|x_i-\widetilde{x}\| \leq 2a_n$. It follows that $\|x_1-x_2\| \leq 4a_n$ and thus they are $O(a_n)$ close in geodesic distance by part 3 of Lemma 3. Hence, there is a geodesic on M connecting x_1 and x_2 that is contained strictly within the Ca_n ball. Because x_2-x_1 lies in $L_b(\widetilde{x})$ and is consequently orthogonal to $T_{\widetilde{x}}\widetilde{M}$, there must exist a point on the geodesic whose angle with $T_{\widetilde{x}}\widetilde{M}$ equals $\pi/2$, contradicting part *1*.

5. Because $H(\widetilde{M}, M) \leq a_n$, we have that $M \subset \mathsf{tube}(\widetilde{M}, a_n)$. Because $a_n < \kappa$, the fibers $L_b(\widetilde{x})$ partition $\mathsf{tube}(\widetilde{M}, a_n)$. Hence, each $x \in M$ must lie on one (and only one) $L_b(\widetilde{x})$. ■

4.11 Step 3b. Construct Slabs that Cover M Nicely

Let $\beth_j = B_D(x_j, \delta_n) \cap \widetilde{M}$. Define the slab

$$R_j = \bigcup_{x\in\beth_j} L_b(x, \widetilde{M}).$$

Lemma 16 *The collection of slabs $R_1, \ldots, R_N$ has the following properties. Let $M \in \mathcal{M}_n$.*

1. $M \subset \bigcup_{j=1}^N R_j$.
2. *$M \cap R_j$ is function-like over R_j. That is, there exists a function $g_j : \beth_j \to \mathbb{R}^{D-d}$ such that $M \cap R_j = \{g_j(x) : x \in \beth_j\}$.*
3. *For each $x \in \beth_j$, $L_b(x) \cap M \neq \emptyset$.*
4. *There exists a linear function $\ell_j : \beth_j \to \mathbb{R}^{D-d}$ such that $\sup_{x\in\beth_j} ||g_j(x) - \ell_j(x)|| \leq C\delta_n^2$.*

5. $\sup_{M\in\mathcal{M}_n} \mathsf{diam}(M\cap R_j) \le C\delta_n$.

Thus the slabs cover M and M cuts across R_j is a function-like way. Moreover, $M\cap R_j$ is nearly linear.

Proof The first three claims follow immediately from Lemma 15. In particular, g_j in claim *2* is defined by $g_j(x) = \{M\cap L_b(x)\}$. Now we show *4*. We can write $g_j(x) = g_j(x_j) + (x-x_j)^T\nabla g + \frac{1}{2}(x-x_j)^T\mathsf{Hess}\,(x-x_j)$ where Hess is the Hessian matrix of g_j evaluated at some point between x and x_j. By part 1 of Lemma 3, the largest eigenvalue of Hess is bounded above by $1/\kappa$. Since $||x-x_j|| \le c\delta_n^2$, the claim follows. Part *5* follows easily. ■

4.12 Step 4: Local Conditional Likelihood

Recall that $\mathcal{M}_n = \{M\in\mathcal{M}(\kappa):\ H(\widetilde{M},M)\le a_n\}$. Let

$$\mathcal{Q}_n = \{Q_M:\ M\in\mathcal{M}_n\}.$$

Consider a slab R_j. For each $Q\in\mathcal{Q}_n$ define $Q_j \equiv Q(\cdot|R_j)$ by $Q_j(A) = Q(A\cap R_j)/Q(R_j)$. Note that Q_j is supported over $\mathsf{tube}(M,\sigma)\cap R_j$. Let $\mathcal{Q}_{n,j} = \{Q_j:\ Q\in\mathcal{Q}_n\}$. Before we proceed we need to establish the following.

Lemma 17 *Let* $I_j(M) = \mathsf{tube}(M,\sigma)\cap R_j$. *Then there exists* $c_0>0$ *such that*

$$\inf_{M\in\mathcal{M}_n} V(I_j(M)) \ge c_0\delta_n^d.$$

Proof By Lemma 16, $M\cap R_j$ lies in a slab of size a_n orthogonal to $\rceil_j$. Because the angle between the two manifolds on this set must be no more than $\pi/4$ and because $a_n>\delta_n$, the manifold M cannot intersect both the "top" and "bottom" surfaces of the slab. Hence, for large enough $C>0$, $\mathcal{I}_j = \bigcup_{x\in\rceil_j} B_D(x,\sigma/C)\subset I_j$. By construction, $V(I_j)\ge V(\mathcal{I}_j)\ge c\delta_n^d$. ■

4.13 Step 4a. The Entropy of $\mathcal{Q}_{n,j}$

Lemma 18 $\mathcal{H}_{[]}(\varepsilon,\mathcal{Q}_{n,j},h)\le c_1\log(c_2/\varepsilon)$.

Proof We begin by creating a γ Hausdorff net for $\mathcal{Q}_{n,j}$. To do this, we will parameterize the support of these distributions. Each $Q\in\mathcal{Q}_{n,j}$ has support in the collection $\mathcal{S}_{n,j} = \{(M\oplus\sigma)\cap R_j:\ M\in\mathcal{M}_n\}$. We will construct a γ-Hausdorff net for $\mathcal{S}_{n,j}$.

Let $\widetilde{x}\in\widetilde{M}$ be the center of $\rceil_j$. Let $y_1,\ldots,y_r$ be a $c_1\gamma$-net of $L_b(\widetilde{x})$, and let $\theta_1<\theta_2<\cdots<\theta_s<\pi/2-\eta$ for a small, fixed $\eta>0$ where $\theta_j-\theta_{j-1}\le c_2\gamma$. Note that $r=O(\gamma^{-(D-d)})$ and $s=O(1/\gamma)$. For every pair y_i and θ_j, let M_{ij} be a $M\in\mathcal{M}_n$ that crosses through y_i with $\mathsf{angle}(T_{y_i}M, T_{\widetilde{x}}\widetilde{M}) = \theta_j$. These manifolds comprise a collection of size $O((1/\gamma)^{D-d-1})$ which we will denote by $\mathsf{Net}(\gamma)$.

Let $M\in\mathcal{M}_n$. Let y be the point where M crosses $L_b(\widetilde{x})$. Let y_i be the closest point in the net to y and let θ_j be the closest angle in the net to $\mathsf{angle}(T_yM, T_{\widetilde{x}}\widetilde{M})$. Because the angle between M and M_{ij} is strictly less than $\pi/4$ (part 1 of Lemma 15) and the slab R_j has radius δ_n, it follows that $H(M,M_{ij})\le C_1\gamma+\delta_nC_2\gamma\le C\gamma$. Hence, $\mathsf{Net}(\gamma)$ is a γ-Hausdorff net.

Now consider $\mathsf{Net}(\gamma)$ with $\gamma = \varepsilon^2$. For each $M_{ij} \in \mathsf{Net}(\gamma)$ let q_{ij} be the corresponding density and define u_{ij} and ℓ_{ij} by

$$u_{ij}(y) = \left(q_{ij}(y) + \frac{C\varepsilon^2}{V(M_{ij} \oplus (\sigma + \varepsilon^2))} \right) I(y \in M_{ij} \oplus (\sigma + \varepsilon^2))$$

and

$$\ell_{ij}(y) = \left(q_{ij}(y) - \frac{C\varepsilon^2}{V(M_{ij} \oplus (\sigma - \varepsilon^2))} \right) I(y \in M_j \oplus (\sigma - \varepsilon^2)).$$

Let $\mathcal{B} = \{(\ell_{ij}, u_{ij})\}$.

Let $M \in \mathcal{M}_n$ and let M_{ij} be the element of the net closest to M. It follows easily that $u_{ij} \geq q_M \geq \ell_{ij}$. Thus $\mathcal{B}$ is a bracketing. Now,

$$\int u_{ij} - \ell_{ij} = 1 + C\varepsilon^2 - (1 - C\varepsilon^2) = 2C\varepsilon^2.$$

Hence, $h(u_{ij}, \ell_{ij}) \leq \sqrt{\int |u_{ij} - \ell_{ij}|} = \sqrt{2C}\varepsilon$. Hence, $\mathcal{B}$ is an $\sqrt{2C} - \varepsilon$-bracketing. So,

$$\mathcal{H}_{[]}(\varepsilon, \mathcal{Q}_{n,j}, h) \leq (D - d - 1) \log(c/\varepsilon),$$

which proves the lemma. ■

4.14 Step 4b. Hellinger Rate of the Conditional MLE

Let $\widehat{q}$ be the mle over $\mathcal{Q}_{n,j}$ using the Y_i's in R_j. Let $\widehat{M}$ be the manifold corresponding to $\widehat{q}$ and let $\widehat{M}_j = \widehat{M} \cap R_j$.

Lemma 19 *For all Q, all $A > 0$ and all large n,*

$$Q^n \left(\left\{ h(Q, \widehat{Q}) > \left(\frac{C_0 \log n}{n} \right)^{\frac{1}{2+d}} \right\} \right) \leq n^{-A}.$$

Proof Let N_j be the number of observations from the second half of the data that are in R_j. Let $\mu_j = \mathbb{E}(N_j)$ and define $m_n = n^{\frac{2}{2+d}}$. First, we claim that $N_j \geq \mu_j/2 = O(m_n)$ for all j, except on a set of probability $e^{-cn^{2/(2+d)}}$. Let $\pi_j = Q(R_j)$. By Lemma 17 and Lemma 4, $\pi_j \geq c\delta_n^d$ for some $c > 0$. Hence, $\mu_j \geq m_n$. Note that $\sigma^2 \equiv \mathsf{Var}(N_j)/n = \pi_j(1 - \pi_j) \leq \pi_j$. Let $t = \mu_j/2$. By Bernstein's inequality,

$$\mathbb{P}(N_j \leq \mu_j/2) = \mathbb{P}(N_j - \mu_j \leq -\mu_j/2) \leq \exp\left\{ -\frac{t^2}{2n\sigma^2 + 2t/3} \right\} \leq \exp\left\{ -cn^{2/(2+d)} \right\}.$$

Hence, by the union bound,

$$\mathbb{P}(N_j \leq \mu_j/2 \text{ for some } j) \leq \frac{1}{N} \exp\left\{ -cn^{2/(2+d)} \right\} \leq \exp\left\{ -c'n^{2/(2+d)} \right\}$$

since there are $N = O(1/\delta_n)$ slabs. Thus we can assume that there are at least order m_n observations in each R_j.

Since $\mathcal{H}_{[]}(\varepsilon, Q_{B,j}, h) \leq \log(C(1/\varepsilon))$, solving the equation $\mathcal{H}_{[]}(\varepsilon, Q_{B,j}, h) = m_n\varepsilon^2$ we get $\varepsilon_m \geq \sqrt{C\log m_n/m_n} = (\log n/n)^{2/(2(2+d))} = \delta_n$. From Lemma 8, we have, for all $Q \in \mathcal{Q}_{n,j}$,

$$Q^n\left(\left\{h(Q,\widehat{Q}) > \delta_n\right\}\right) = Q^n\left(\left\{h(Q,\widehat{Q}) > \varepsilon_m\right\}\right) \leq c_1 e^{-c_2 m_n \varepsilon_m^2} \leq n^{-A}.$$

■

4.15 Step 4c. Relating Hausdorff Distance to Hellinger Distance Within a Slab

Lemma 20 *For each $M_1, M_2 \in \mathcal{M}_n$, $H(M_1 \cap R_j, M_2 \cap R_j) \leq C\, h^2(Q_{j1}, Q_{j2})$.*

Proof Let g_1 and g_2 be defined as in Lemma 16. There exists $x \in \rotatebox{180}{L}_j$ such that $g_1(x) \in M_1, g_2(x) \in M_2$ and $||g_1(x) - g_2(x)|| = \gamma$. We claim there exists $\rotatebox{180}{L}' \subset \rotatebox{180}{L}_j$ such that $\inf_{x\in\rotatebox{180}{L}'} ||g_1(x) - g_2(x)|| \geq \gamma/2$ and such that $V(\rotatebox{180}{L}') \geq c\delta_n^d$. This follows since g_1 and g_2 are smooth, they both lie in a slab of size a_n around $\rotatebox{180}{L}_j$ and the angle between the tangent of $g_j(x)$ and $\rotatebox{180}{L}_j$ is bounded by $\pi/4$.

Create a modified manifold M_2' such that M_2' differs from M_1 over $\rotatebox{180}{L}'$ by a $\gamma/2$ shift orthogonal to $\rotatebox{180}{L}_j$ and such that M_2' is otherwise equal to M_1. It follows that $\ell_1(M_1, M_2) \geq \ell_1(M_1, M_2')$ and $h(Q_1, Q_2) \geq h(Q_1, Q_2')$.

Every point in the support of the conditioned distributions can be written as an ordered pair (x, y) where $x \in \rotatebox{180}{L}_j$ and y lies in a d' ball of radius σ. M_2' is shifted a distance of $\gamma/2$ in the direction orthogonal to $\rotatebox{180}{L}_j$. As a result, the ℓ_1 distance between M_1 and M_2' equals the integral over C' of the volume difference between two d' balls of the same radius that are shifted by $\gamma/2$ relative to each other. This volume $\delta_n^d\gamma$. Hence, $V(M_1 \cap \rotatebox{180}{L}_j) \circ (M_2 \cap \rotatebox{180}{L}_j) \geq \gamma\delta_n^d$. Let $A = \{x \in \rotatebox{180}{L}_j :\ q_1 > 0, q_2 = 0\}$, $B = \{x \in \rotatebox{180}{L}_j :\ q_1 > 0, q_2 > 0\}, C = \{x \in \rotatebox{180}{L}_j :\ q_1 = 0, q_2 > 0\}$. At least one of A or B has volume at least $\gamma\delta_n^d/2$. Without loss of generality, assume that it is A. Then

$$\begin{aligned} h^2(q_1, q_2) &= \int(\sqrt{q_1} - \sqrt{q_2})^2 \geq \int_A (\sqrt{q_1} - \sqrt{q_2})^2 = \int_A q_1 \\ &\geq \frac{C_* c\delta_n^d\gamma}{\delta_n^d} = cC_*\gamma = cC_* H(M_1, M_2). \end{aligned}$$

■

4.16 Step 4d. The Hausdorff Rate

Lemma 21 *For any $A > 0$ there exists C_0 such that*

$$Q^n\left(\left\{H(M \cap R_j, \widehat{M}_j) > \left(\frac{C_0 \log n}{n}\right)^{\frac{2}{2+d}}\right\}\right) \leq \frac{1}{n^A}.$$

Proof This follows by combining Lemma 20 and Lemma 19. ■

4.17 Step 5: Final Estimator

Now we can combine the estimators from the difference slabs. Let $\widehat{M} = \bigcup_{j=1}^{N} \widehat{M}_j$. Recall that the number of slabs is $N = (c\delta_n)^{-d} = (Cn/\log n)^{d/(2+d)}$.

Proof of Theorem 2. Choose an $A > 2/(2+d)$. We have:

$$\begin{aligned} Q^n\left(\left\{H(\widehat{M},M) > \left(\frac{C_0 \log n}{n}\right)^{\frac{2}{2+d}}\right\}\right) &\leq \sum_j Q^n\left(\left\{H(\widehat{M}_j, M\cap R_j) > \left(\frac{C_0\log n}{n}\right)^{\frac{2}{2+d}}\right\}\right) \\ &\leq \frac{N}{n^A} \\ &= \left(\frac{n}{C\log n}\right)^{\frac{1}{2+d}} \times \frac{1}{n^A} \leq \frac{c}{n^A}. \end{aligned}$$

Let $r_n = \left(\frac{C_0\log n}{n}\right)^{2/(2+d)}$. Since M and $\widehat{M}$ are contained in a compact set, $H(\widehat{M},M)$ is uniformly bounded above by a constant K_0. Hence,

$$\begin{aligned} \mathbb{E}_Q H(\widehat{M},M) &= \mathbb{E}_Q[H(\widehat{M},M)I(H(\widehat{M},M) > r_n)] + \mathbb{E}_Q[H(\widehat{M},M)I(H(\widehat{M},M) \leq r_n)] \\ &\leq K_0\, Q^n(H(\widehat{M},M) > r_n) + r_n \\ &\leq \frac{c}{n^A} + r_n = O\left(\left(\frac{\log n}{n}\right)^{2/(2+d)}\right). \end{aligned}$$

■

5. A Simple, Consistent Estimator

Here we give a practical, consistent estimator, one that does not converge at the optimal rate. It is a generalization of the estimator in Genovese et al. (2010) and is similar to the estimator in Niyogi et al. (2006). Let

$$\widehat{S} = \bigcup_{i=1}^{n} B_D(Y_i, \varepsilon)$$

and define $\widehat{\partial S} = \partial(\widehat{S})$, $\widehat{\sigma} = \max_{y\in\widehat{S}} d(y, \widehat{\partial S})$ and

$$\widehat{M} = \left\{y \in \widehat{S} :\ d(y,\widehat{\partial S}) \geq \widehat{\sigma} - 2\varepsilon\right\}.$$

Lemma 22 *Let $\varepsilon_n = C(\log n/n)^{1/D}$ in the estimator $\widehat{M}$. Then*

$$H(M,\widehat{M}) = O\left(\frac{\log n}{n}\right)^{1/D}$$

almost surely for all large n.

Before proving the lemma we need a few definitions. Following Cuevas and Rodríguez-Casal (2004), we say that a set S is (χ,λ)-*standard* if there exist positive numbers χ and λ such that

$$\nu_D(B_D(y,\varepsilon)\cap S) \geq \chi\, \nu_D(B(y,\varepsilon)) \quad \text{for all } y\in S,\ 0<\varepsilon\leq\lambda.$$

We say that S is *partly expandable* if there exist $r>0$ and $R\geq 1$ such that $H(\partial S, \partial(S\oplus\varepsilon)) \leq R\varepsilon$ for all $0\leq\varepsilon<r$. A standard set has no sharp peaks while a partly expandable set has not deep inlets.

Lemma 23 *If $\sigma < \Delta(M)$ then $S = M \oplus \sigma$ is standard with $\chi = 2^{-D}$ and $\lambda = \sigma$ and partly expandable with $r = \Delta(M) - \sigma$ and $R = 1$.*

Proof Let $\chi = 2^{-D}$. Let y be a point in S and let $\Lambda(y) \leq \sigma$ be its distance from the boundary ∂S. If $\Lambda(y) \geq \varepsilon$ then $B_D(y,\varepsilon) \cap S = B_D(y,\varepsilon)$ so that $\nu_D(B_D(y,\varepsilon) \cap S) = \nu_D(B_D(y,\varepsilon)) \geq \chi \nu_D(B_D(y,\varepsilon))$.

Suppose that $\Lambda(y) < \varepsilon$. Let v be a point on the manifold closest to y and let y^* be the point on the segment joining y to v such that $||y - y^*|| = \varepsilon/2$. The ball $A = B_D(y^*, \varepsilon/2)$ is contained in both $B_D(y,\varepsilon)$ and S. Hence, $\nu_D(B_D(y,\varepsilon) \cap S) \geq \nu_D(A) \geq \chi \nu_D(B_D(y,\varepsilon))$. This is true for all $\varepsilon \leq \sigma$, hence S is (χ,λ)-standard for $\chi = 1/2^D$ and $\lambda = \sigma$.

Now we show that S is partly expandable. By Proposition 1 in Cuevas and Rodríguez-Casal (2004) it suffices to show that a ball of radius r rolls freely outside S for some r, meaning that, for each $y \in \partial S$, there is an a such that $y \in B(a,r) \subset \overline{S^c}$, where S^c is the complement of S. Let O_y be the ball of radius $\Delta - \sigma$ tangent to y such that $O_y \subset S^c$. Such a ball exists by virtue of the fact that $\sigma < \Delta(M)$. ■

Theorem 24 (Cuevas and Rodríguez-Casal, 2004) *Let $Y_1, \ldots, Y_n$ be a random sample from a distribution with support S. Let S be compact, (λ,χ)-standard and partly expandable. Let*

$$\widehat{S} = \bigcup_{i=1}^{n} B(Y_i, \varepsilon_n)$$

and let $\widehat{\partial S}$ be the boundary of $\widehat{S}$. Let $\varepsilon_n = C(\log n/n)^{1/D}$ with $C > (2/(\chi\ \omega_D))^{1/D}$ where $\omega_D = V(B_D(0,1))$. Then, with probability one,

$$H(S,\widehat{S}) \leq C\left(\frac{\log n}{n}\right)^{1/D} \quad \text{and} \quad H(\partial S, \widehat{\partial S}) \leq C\left(\frac{\log n}{n}\right)^{1/D}$$

for all large n. Also, $S \subset \widehat{S}$ almost surely for all large n.

Proof of Lemma 22. Theorem 24 and Lemma 23 imply that $H(S,\widehat{S}) \leq C(\log n/n)^{1/D}$ and $H(\partial S, \widehat{\partial S}) \leq C(\log n/n)^{1/D}$. It follows that $\widehat{\sigma} \geq \sigma - \varepsilon$. First we show that $y \in \widehat{M}$ implies that $d(y,M) \leq 4\varepsilon$. Let $y \in \widehat{M}$. Then $d(y,\partial S) \geq d(y,\widehat{\partial S}) - \varepsilon \geq \widehat{\sigma} - 2\varepsilon - \varepsilon \geq \sigma - \varepsilon - 2\varepsilon - \varepsilon = \sigma - 4\varepsilon$. So $d(y,M) = \sigma - d(y,\partial S) \leq \sigma - \sigma + 4\varepsilon = 4\varepsilon$. Now we show that $M \subset \widehat{M}$. Suppose that $y \in M$. Then,

$$d(y,\widehat{\partial S}) \geq d(y,\partial S) - \varepsilon = \sigma - \varepsilon \geq \widehat{\sigma} - 2\varepsilon$$

so that $y \in \widehat{M}$. ■

6. Conclusion and Open Questions

We have established that the optimal rate for estimating a smooth manifold in Hausdorff distance is $n^{-\frac{2}{2+d}}$. We conclude with some comments and open questions.

1. We have assumed that the noise is perpendicular to the manifold. In current work we are deriving the minimax rate under the more general assumption that ε is drawn from a general,

spherically symmetric distribution. We also allow the distribution along the manifold to be any smooth density bounded away from 0. The rates are quite different and the methods for proving the rates are substantially more involved. Moreover, the rates depends on the behavior of the noise density near the boundary of its support. We will report on this elsewhere.

2. Perhaps the most important open question is to find a computationally tractable estimator that achieves the optimal rate. It is possible that combining the estimator in Section 5 with one of the estimators in the computational geometry literature (Dey, 2006) could work. However, it appears that some modification of such an estimator is needed. This is a difficult question which we hope to address in the future.

3. It is interesting to note that Niyogi et al. (2006) have a Gaussian noise distribution. While it is possible to infer the homology of M with Gaussian noise it is not possible to infer M itself with any accuracy. The reason is that manifold estimation is similar to (and in fact, more difficult than) nonparametric regression with measurement error. In that case, it is well known that the fastest possible rates under Gaussian noise are logarithmic. This highlights an important distinction between estimating the topological structure of M versus estimating M in Hausdorff distance.

4. The current results take $\Delta(M)$, d and σ as known (or at least bounded by known constants). In practice these must be estimated. We do not know whether there exist minimax estimators that are adaptive over $d, \Delta(M)$ and σ.

Acknowledgments

The authors thank Don Sheehy for helpful comments on an earlier draft of this paper. The authors also thank the reviewers for their comments and questions.

7. Appendix

This appendix contains proofs of some technical results used earlier in the paper.

7.1 Proof of Equation 2

We will use the following two results (see Section 2.4 of Tsybakov, 2008):

$$h^2(P^n, Q^n) = 2\left(1 - \left[1 - \frac{h^2(P,Q)}{2}\right]^n\right)$$

and

$$||P \wedge Q|| \geq \frac{1}{2}\left(1 - \frac{h^2(P,Q)}{2}\right)^2.$$

We have

$$\begin{aligned} ||P^n \wedge Q^n|| &\geq \frac{1}{2}\left(1 - \frac{h^2(P^n,Q^n)}{2}\right)^2 = \frac{1}{2}\left(1 - \frac{h^2(P,Q)}{2}\right)^{2n} \\ &\geq \frac{1}{2}\left(1 - \frac{\ell_1(P,Q)}{2}\right)^{2n} \end{aligned}$$

since $h^2(P,Q) \leq \ell_1(P,Q)$.

7.2 Proof of Theorem 6

We define two manifolds M_0 and M_1 with corresponding distributions Q_0 and Q_1 such that (i) $\Delta(M_i) \geq \kappa$ $i = 0,1$, (ii) $H(M_0,M_1) = \gamma$ and (iii) such that the volume of $S_0 \circ S_1$ is of order $\gamma^{\frac{d}{2}+1}$, where S_i is the support of Q_i.

We write a generic D-dimensional vector as $y = (u,v,z)$, with $u \in \mathbb{R}^d$, $v \in \mathbb{R}$, $z \in \mathbb{R}^{D-d-1}$. For each $u \in \mathbb{R}^d$ with $||u|| \leq 1$, define the disk in $\mathbb{R}^{d+1}$

$$D_0 = \left\{ (u,0) \in \mathbb{R}^{d+1} : \ u \in B_d(0,1) \right\}$$

and let

$$F_0 = \partial \left(\bigcup_{(u,v) \in D_0} B_{d+1}((u,v),\kappa) \right).$$

Now define the following d-dimensional manifold in $\mathbb{R}^D$

$$\begin{aligned} M_0 &= \left\{ (u,v,0_{D-d-1}) : \ (u,v) \in F_0 \right\} \\ &= \left\{ (u,a(u),0_{D-d-1}) : \ u \in B_d(0,1+\kappa) \right\} \cup \left\{ (u,-a(u),0_{D-d-1}) : \ u \in B_d(0,1+\kappa) \right\} \end{aligned}$$

where

$$a(u) = \begin{cases} \kappa & \text{if } ||u|| \leq 1 \\ \sqrt{\kappa^2 - (||u|| - 1)^2} & \text{if } 1 < \ ||u|| \leq 1+\kappa. \end{cases}$$

The manifold M_0 has no boundary and, by construction, $\Delta(M_0) \geq \kappa$.

Now define a second manifold that coincides with M_0 but has a small perturbation. Let $\gamma \in (0, 4\kappa)$ and define

$$M_1 = \left\{ (u,b(u),0_{D-d-1}) : \ u \in B_d(0,1+\kappa) \right\} \cup \left\{ (u,-a(u),0_{D-d-1}) : \ u \in B_d(0,1+\kappa) \right\}$$

where

$$b(u) = \begin{cases} \gamma + \sqrt{\kappa^2 - ||u||^2} & \text{if } ||u|| \leq \frac{1}{2}\sqrt{4\gamma\kappa - \gamma^2} \\ 2\kappa - \sqrt{\kappa^2 - (||u|| - \sqrt{4\gamma\kappa - \gamma^2})^2} & \text{if } \frac{1}{2}\sqrt{4\gamma\kappa - \gamma^2} < \ ||u|| \leq \sqrt{4\gamma\kappa - \gamma^2} \\ a(u) & \text{if } \sqrt{4\gamma\kappa - \gamma^2} < \ ||u|| \leq \sqrt{4\gamma\kappa - \gamma^2} + \kappa. \end{cases}$$

Note that $\Delta(M_1) \geq \kappa$ since the perturbation is obtained using portions of spheres of radius κ. In fact

- for $||u|| \leq \frac{1}{2}\sqrt{4\gamma\kappa - \gamma^2}$, $b(u)$ is the $d+1$-th coordinate of the "upper" portion of the $(d+1)$-dimensional sphere with radius κ centered at $(0,\cdots,0,\gamma)$, hence $b(u)$ satisfies

$$||u||^2 + (b(u) - \gamma)^2 = \kappa^2 \qquad \text{with } b(u) \geq \gamma;$$

- for $\frac{1}{2}\sqrt{4\gamma\kappa - \gamma^2} < \ ||u|| \leq \sqrt{4\gamma\kappa - \gamma^2}$, $b(u)$ is the $(d+1)$-th coordinate of the "lower" portion of the $(d+1)$-dimensional sphere with radius κ centered at $(u \cdot \sqrt{4\gamma\kappa - \gamma^2}/||u||, 2\kappa)$ (note that the center of the sphere differs according to the direction of u), hence $b(u)$ satisfies

$$\left|\left| u - \frac{u}{||u||}\sqrt{4\gamma\kappa - \gamma^2} \right|\right|^2 + (b(u) - 2\kappa)^2 = \kappa^2 \qquad \text{with } b(u) \leq 2\kappa.$$

To summarize, M_0 and M_1 are both manifolds with no boundary, $\Delta(M_0) \geq \kappa$ and $\Delta(M_1) \geq \kappa$. See Figure 5. Now

$$\begin{aligned} E_0 &= M_0 - M_1 = \left\{(u, a(u), 0_{D-d-1}) : \ u \in B_d(0, \sqrt{4\gamma\kappa - \gamma^2})\right\} \\ E_1 &= M_1 - M_0 = \left\{(u, b(u), 0_{D-d-1}) : \ u \in B_d(0, \sqrt{4\gamma\kappa - \gamma^2})\right\}. \end{aligned}$$

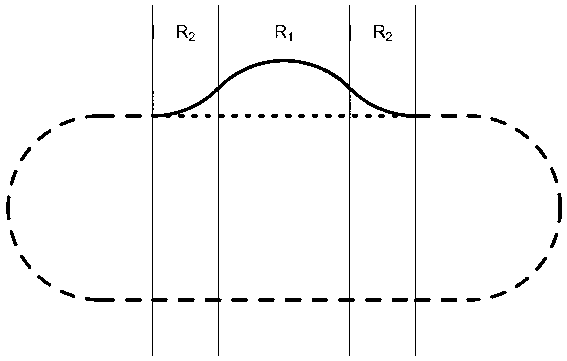


Figure 5: One section of manifolds M_0 and M_1. The common part is dashed, E_0 is dotted and E_1 solid. R_1 and R_2 denote the regions where the different definitions of the perturbation apply: R_1 is $||u|| \leq \frac{1}{2}\sqrt{4\gamma\kappa - \gamma^2}$ while R_2 denotes $\frac{1}{2}\sqrt{4\gamma\kappa - \gamma^2} < ||u|| \leq \sqrt{4\gamma\kappa - \gamma^2}$.

Note that for each point $y \in E_0$ there exists $y' \in E_1$ such that $||y - y'|| \leq |a(u) - b(u)| \leq \gamma$. Also, $y_0 = (0, a(0), 0) \in M_0$ has as its closest M_1 point $y_1 = (0, b(0), 0)$, so that $||y_0 - y_1|| = \gamma$. Hence $H(M_0, M_1) = H(E_0, E_1) = \gamma$.

To find an upper bound for $V(S_0 \circ S_1)$, we show that each $y = (u, v, z) \in S_1 - S_0$ satisfies the following conditions:

(i) $u \in B_d(0, \sqrt{4\gamma\kappa - \gamma^2})$;

(ii) $z \in B_{D-d-1}(0, \sigma)$;

(iii) $\kappa + \sigma - ||z|| < v \leq \kappa + \gamma + \sigma - ||z||$.

If $y = (u, v, z)$ belongs to S_1 and has $||u|| > \sqrt{4\gamma\kappa - \gamma^2}$, then there is a point of $M_0 \cap M_1$ within distance σ, hence $y \notin S_1 - S_0$. This proves (i). Before proving (ii) and (iii), note that if $u \in B_d(0, \sqrt{4\gamma\kappa - \gamma^2})$ then

$$\kappa = a(u) \leq b(u) \leq \kappa + \gamma.$$

Now, let $y' = (u', b(u'), 0) \in E_1$ be the point in S_1 closest to y. We have

$$d(y, S_1) = ||y - y'|| \leq ||u - u'|| + |v - b(u')| + ||z|| \leq \sigma.$$

This gives condition (ii) above $||z|| \leq \sigma$ and also

$$|v - b(u')| \leq \sigma - ||z||. \tag{4}$$

Since $b(u') \leq \kappa + \gamma$, we obtain

$$v \leq b(u') + \sigma - ||z|| \leq \kappa + \gamma + \sigma - ||z||$$

which is the right inequality in (iii). Finally,

$$\sigma < d(y, M_0) \leq ||y - (u, a(u), 0)|| \leq |v - a(u)| + ||z||$$

which implies either $v < a(u) - (\sigma - ||z||)$ or $v > a(u) + (\sigma - ||z||)$. The former inequality would imply

$$v < a(u) - (\sigma - ||z||) = \kappa - (\sigma - ||z||) \leq \inf_{u'} b(u') - (\sigma - ||z||)$$

so that $|v - b(u')| > \sigma - ||z||$ for all u', which is in contradiction with (4). Hence we have $v > a(u) + (\sigma - ||z||) = \kappa + (\sigma - ||z||)$ that is the left inequality in (iii).

As a consequence,

$$S_1 - S_0 \subset B_d(0, \sqrt{4\gamma\kappa - \gamma^2}) \times \left\{ (v, z) \in \mathbb{R}^{D-d} : \kappa - \gamma + \sigma - ||z|| < v \leq \kappa + \gamma + \sigma - ||z||, z \in B_{D-d-1}(0, \sigma) \right\}$$

and

$$V(S_0 - S_1) \leq C \cdot (\sqrt{4\gamma\kappa - \gamma^2})^d \cdot \gamma \cdot \sigma^{D-d-1}.$$

Hence, $V(S_0 - S_1) = O(\gamma^{\frac{d}{2}+1})$.

With similar arguments one can show that $V(S_1 - S_0) = O(\gamma^{\frac{d}{2}+1})$ so that

$$V(S_0 \circ S_1) = O(\gamma^{\frac{d}{2}+1}).$$

It then follows that $\int |q_0 - q_1| = O(\gamma^{(d+2)/2})$.

References

R. Baraniuk and M. Wakin. Random projections of smooth manifolds. *Foundations of Computational Mathematics*, 9:51–77, 2007.

M. Birman and M. Solomjak. Piecewise-polynomial approximation of functions of the classes w_p. *Mathematics of USSR Sbornik*, 73:295–317, 1967.

J. Boissonnat and A. Ghosh. Manifold reconstruction using tangential delaunay complexes. In *Proceedings of the 2010 Annual Symposium on Computational Geometry*, pages 324–333. ACM, 2010.

F. Chazal and A. Lieutier. Smooth manifold reconstruction from noisy and non-uniform approximation with guarantees. *Computational Geometry*, 40:156–170, 2008.

S. Cheng and T. Dey. Manifold reconstruction from point samples. In *Proceedings of the Sixteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1018–1027. SIAM, 2005.

A. Cuevas and A. Rodríguez-Casal. On boundary estimation. *Advances in Applied Probability*, 36 (2):340–354, 2004.

L. Devroye and G. Wise. Detection of abnormal behavior via nonparametric estimation of the support. *SIAM Journal on Applied Mathematics*, 38:480–488, 1980.

T. Dey. *Curve and Surface Reconstruction: Algorithms with Mathematical Analysis*. Cambridge University Press, 2006.

T. Dey and S. Goswami. Provable surface reconstruction from noisy samples. In *Proceedings of the Twentieth Annual Symposium on Computational Geometry*, pages 330–339. ACM, 2004.

H. Federer. Curvature measures. *Transactions of the American Statistical Society*, 93:418–491, 1959.

C. Genovese, M. Perone-Pacifico, I. Verdinelli, and L. Wasserman. Nonparametric filament estimation. *arXiv:1003.5536*, 2010.

O. Gonzalez and J. Maddocks. Global curvature, thickness, and the ideal shapes of knots. *Proceedings of the National Academy of Sciences*, 96(9):4769–4773, 1999.

L. LeCam. Convergence of estimates under dimensionality restrictions. *The Annals of Statistics*, pages 38–53, 1973.

J.M. Lee. *Introduction to Smooth Manifolds*. Springer, 2002.

P. Niyogi, S. Smale, and S. Weinberger. Finding the homology of submanifolds with high confidence from random samples. *Discrete and Computational Geometry*, 39:419–441, 2006.

P. Niyogi, S. Smale, and S. Weinberger. A topological view of unsupervised learning from noisy data. *Unpublished technical report, University of Chicago*, 2008.

X. Shen and W. Wong. Probability inequalities for likelihood ratios and convergence rates of sieve mles. *The Annals of Statistics*, 23:339–362, 1995.

A. Tsybakov. *Introduction to Nonparametric Estimation*. Springer, 2008.

B. Yu. Assouad, Fano, and Le Cam. In *Festschrift for Lucien Le Cam*. Springer, 1997.

Journal of Machine Learning Research 13 (2012) 1293-1332 Submitted 7/10; Revised 1/12; Published 5/12

Query Strategies for Evading Convex-Inducing Classifiers

Blaine Nelson BLAINE.NELSON@WSII.UNI-TUEBINGEN.DE
Wilhelm Schickard Institute for Computer Science
University of Tübingen
Sand 1
72076 Tübingen, Germany

Benjamin I. P. Rubinstein BEN.RUBINSTEIN@MICROSOFT.COM
Microsoft Research
1288 Pear Avenue
Mountain View, CA 94043, USA

Ling Huang LING.HUANG@INTEL.COM
ISTC on Secure Computing at UC Berkeley
711 Soda Hall
Berkeley, CA 94720-1776, USA

Anthony D. Joseph ADJ@CS.BERKELEY.EDU
Steven J. Lee STEVENJLEE@BERKELEY.EDU
Satish Rao SATISHR@CS.BERKELEY.EDU
J. D. Tygar TYGAR@CS.BERKELEY.EDU
Computer Science Division
University of California
Berkeley, CA 94705-1545, USA

Editor: Tong Zhang

Abstract

Classifiers are often used to detect miscreant activities. We study how an adversary can systematically query a classifier to elicit information that allows the attacker to evade detection while incurring a near-minimal cost of modifying their intended malfeasance. We generalize the theory of Lowd and Meek (2005) to the family of convex-inducing classifiers that partition their feature space into two sets, one of which is convex. We present query algorithms for this family that construct undetected instances of approximately minimal cost using only polynomially-many queries in the dimension of the space and in the level of approximation. Our results demonstrate that near-optimal evasion can be accomplished for this family without reverse engineering the classifier's decision boundary. We also consider general ℓ_p costs and show that near-optimal evasion on the family of convex-inducing classifiers is generally efficient for both positive and negative convexity for all levels of approximation if $p = 1$.

Keywords: query algorithms, evasion, reverse engineering, adversarial learning

1. Introduction

A number of systems and security engineers have proposed the use of machine learning to detect miscreant activities in a variety of applications; for example, spam, intrusion, virus, and fraud detection. However, all known detection techniques have blind spots: classes of miscreant activity

that fail to be detected. While learning allows the detection algorithm to adapt over time, real-world constraints on the learner typically allow an adversary to programmatically find vulnerabilities. We consider how an adversary can systematically discover blind spots by querying a fixed or learning-based detector to find a low cost (for some cost function) instance that the detector does not filter. As a motivating example, consider a spammer who wishes to minimally modify a spam message so it is not classified as a spam (here, cost is a measure of how much the spam must be modified). As a second example, consider the design of an exploit that must avoid intrusion detection systems (here, cost may be a measure of the exploit's severity). There are a variety of domain-specific mechanisms an adversary can use to observe the classifier's response to a query, or in other words, to query a membership oracle of the filter; for example, the spam filter of a public email system can be observed by creating a dummy account on that system and sending the queries to that account. By observing the responses of the detector, the adversary can search for a modification while using as few queries as possible.

The idealized theoretical problem of near-optimal evasion was first posed by Lowd and Meek (2005). We continue their investigation by generalizing their results to convex-inducing classifiers—classifiers that partition feature space into two sets, one of which is convex. The family of convex-inducing classifiers is a particularly natural set to examine, as it includes the family of linear classifiers studied by Lowd and Meek as well as anomaly detection classifiers using bounded PCA (Lakhina et al., 2004), anomaly detection algorithms that use hyper-sphere boundaries (Bishop, 2006), one-class classifiers that predict anomalies by thresholding the log-likelihood of a log-concave (or uni-modal) density function, and quadratic classifiers with a decision function of the form $\mathbf{x}^\top \mathbf{A}\mathbf{x} + \mathbf{b}^\top \mathbf{x} + c \geq 0$ if $\mathbf{A}$ is semidefinite (see Boyd and Vandenberghe, 2004, Chapter 3), to name a few. Furthermore, the family of convex-inducing classifiers also includes more complicated bodies such as the countable intersection of halfspaces, cones, or balls.

We also show that near-optimal evasion does not require reverse engineering the classifier's decision boundary, which is the approach taken by Lowd and Meek (2005) for evading linear classifiers in a continuous domain. Our algorithms for evading convex-inducing classifiers do not require fully estimating the classifier's boundary. Instead, we directly search for a minimal-cost evading instance. Since our algorithms require only polynomially-many queries, while reverse engineering the general convex case is hard (see Rademacher and Goyal, 2009), our algorithms witness a separation between the complexities of reverse engineering and evasion. In the special case of linear classifiers, our algorithms achieve better query complexity than the previously-published reverse-engineering technique. Finally, we also extend near-optimal evasion to general ℓ_p costs. For these costs, we show that our algorithms can also be used for near-optimal evasion, but are generally not efficient. However, in the cases when our algorithms are not efficient, we show that there is no efficient query-based algorithm.

A preliminary version of this paper was previously published as the report (Nelson et al., 2010b) extending our earlier work (Nelson et al., 2010a). This paper is organized as follows. We overview past work related to near-optimal evasion in the remainder of this section. In Section 2, we formalize the near-optimal evasion problem, and review Lowd and Meek's definitions and results. We present algorithms for evasion that are near-optimal under weighted ℓ_1 costs in Section 3, and we consider minimizing general ℓ_p costs in Section 4. We conclude the paper by discussing future directions for near-optimal evasion in Section 5.

1.1 Related Work

Lowd and Meek (2005) first explored near-optimal evasion and developed a method that reverse engineered linear classifiers in a continuous domain. Our approach generalizes that result and improves upon it in three significant ways.

- We consider a more general family of classifiers: the family of convex-inducing classifiers that partition the space of instances into two sets, one of which is convex. This family subsumes the family of linear classifiers considered by Lowd and Meek.
- Our approach does not fully estimate the classifier's decision boundary (which is generally hard; see Rademacher and Goyal 2009) or reverse-engineer the classifier's state. Instead, we directly search for an instance that the classifier labels as negative and is close to the desired attack instance (an evading instance of near-minimal cost). Lowd and Meek previously demonstrated a direct search technique for linear classifiers in Boolean spaces, but that technique is not applicable to the classifiers we consider.
- Even though our algorithms find solutions for a more general family of classifiers, our algorithms still use only polynomially-many queries in the dimension of the feature space and the accuracy of the desired approximation. Moreover, our K-STEP MULTILINESEARCH (Algorithm 3) solves the linear case with asymptotically fewer queries than the previously-published reverse-engineering technique.

Dalvi et al. (2004) use a game-theoretic approach to preemptively patch a cost-sensitive naive Bayes classifier's blind spots. They construct a modified classifier designed to detect optimally modified instances. Brückner and Scheffer (2009) and Kantarcioglu et al. (2009) have extended this setting to larger families of classifiers and developed techniques to solve for equilibrium strategies to their game. This prior research is complementary to query-based evasion; the near-optimal evasion problem studies how an adversary can use queries to find blind spots of a classifier that is unknown but queryable whereas their game-theoretic approaches assume the adversary knows the classifier and can optimize their evasion accordingly at each step of an iterated game.

A number of authors have studied evading sequence-based intrusion detector systems (IDSs) (Tan et al., 2002; Wagner and Soto, 2002). In exploring *mimicry attacks*, these authors demonstrated that real IDSs can be fooled by modifying exploits to mimic normal behaviors. These authors used offline analysis of the IDSs to construct their modifications; by contrast, our modifications are optimized by querying the classifier.

The field of active learning also studies a form of query-based optimization (Schohn and Cohn, 2000). As summarized by Settles (2009), the three primary approaches to active learning are membership query synthesis, stream-based selective sampling and pool-based sampling. Our work is most closely related to the membership query synthesis subfield introduced by Angluin (1988) in which the learner can request the label for any instance in feature space rather than for unlabeled instances drawn from a distribution. However, while active learning and near-optimal evasion are similar in their exploration of query strategies, the objectives for these two settings are quite different—evasion approaches search for low-cost negative instances within a factor $1+\varepsilon$ of an optimal cost whereas active learning algorithms seek to obtain hypotheses with low generalization error often in a PAC-setting (see Section 2.3 for a discussion on reverse-engineering approaches to evasion and active learning). It is interesting to note, nonetheless, that results in active learning settings (e.g., Dasgupta et al., 2009; Feldman, 2009) have also achieved polynomial query complexities

in specific cases. However, we focus solely on the evasion objective, and we leave the exploration of relationships between our results and those in active learning to future work.

Another class of related techniques that use query-based optimization are non-gradient global optimization methods often referred to as direct search. Simple examples of these techniques include bisection and golden-section search methods, for finding roots and extrema of univariate functions, and derivative approximation approaches such as the secant method and interpolation methods (e.g., Burden and Faires, 2000). Combinations of these approaches include Dekker's and Brent's algorithms (e.g., Brent, 1973), which exhibit superlinear convergence under certain conditions on the query function; that is, the number of queries is inversely quadratic in the desired error tolerance. However, while these approaches can be adapted to multiple dimensions, their query complexity grows exponentially with the dimension. Other approaches include the simplex method of Nelder and Mead (1965) and the DIRECT search algorithm introduced by Jones et al. (1993) (refer to Jones, 2001 and Kolda et al., 2003 for surveys of direct search methods), however, we are unaware of query bounds for these methods. While any direct search methods can be adapted for near-optimal evasion, these methods were designed to optimize an irregular function in a regular domain with few dimensions whereas the near-optimal evasion problem involves optimizing regular known functions (the cost function) over an unknown, possibly irregular, and high-dimensional domain (the points labeled as negative by the classifier). The methods we present specifically exploit the regular structure of ℓ_p costs and of the convex-inducing classifiers to achieve near-optimality with only polynomially-many queries.

2. Problem Setup

We begin by introducing our notation and assumptions. First, we assume that instances are represented in D-dimensional Euclidean *feature space*[1] $\mathcal{X} = \Re^D$ such as for some intrusion detection systems (e.g., Wang and Stolfo, 2004). Each component of an instance $\mathbf{x} \in \mathcal{X}$ is a *feature* which we denote as x_d. We use δ_d to denote each coordinate vector of the form $(0,\ldots,1,\ldots,0)$ with a 1 only at the d^{th} feature. We assume the feature space representation is known by the adversary and there are no restrictions on the adversary's queries; that is, any point $\mathbf{x}$ in feature space $\mathcal{X}$ can be queried by the adversary to learn the classifier's prediction at that point. These assumptions may not be true in every real-world setting (for instance, spam detectors are often defined with discrete features and designers often attempt to hide or randomize their feature set; for example, see Wang et al., 2006), but they allow us to investigate strategies taken by a worst-case adversary. We revisit these assumptions in Section 5.

We further assume the target classifier f belongs to a family of classifiers $\mathcal{F}$. Any *classifier* $f \in \mathcal{F}$ is a mapping $f : \mathcal{X} \to \mathcal{Y}$ from feature space $\mathcal{X}$ to its *response space* $\mathcal{Y}$. We assume the adversary's attack will be against a fixed f so the learning method and the training data used to select f are irrelevant. We assume the adversary does not know f but knows its family $\mathcal{F}$. We also restrict our attention to binary classifiers with $\mathcal{Y} = \{'-','+'\}$.

We assume $f \in \mathcal{F}$ is deterministic and so it partitions $\mathcal{X}$ into two sets—the positive class $\mathcal{X}_f^+ = \{\mathbf{x} \in \mathcal{X} \mid f(\mathbf{x}) = '+'\}$ and the negative class $\mathcal{X}_f^- = \{\mathbf{x} \in \mathcal{X} \mid f(\mathbf{x}) = '-'\}$. We take the negative set to be *normal* instances. We assume that the adversary is aware of at least one instance in each class,

1. Lowd and Meek also consider integer and Boolean-valued feature spaces and derive results for several classes of learners in these discrete-valued spaces.

$\mathbf{x}^- \in \mathcal{X}_f^-$ and $\mathbf{x}^A \in \mathcal{X}_f^+$, and can observe $f(\mathbf{x})$ for any $\mathbf{x}$ by issuing a *membership query* (see Section 5 for a more detailed discussion).

2.1 Adversarial Cost

We assume the adversary has a notion of utility over the feature space, which we quantify with a cost function $A : \mathcal{X} \to \Re^{0+}$ (the non-negative reals); for example, for a spammer, this could be a string edit distance on email messages. The adversary wishes to optimize A over the negative class, $\mathcal{X}_f^-$; for example, the spammer wants to send spam that will be classified as normal email ('$-$') rather than as spam ('$+$'). We assume this cost function is a distance to some target instance $\mathbf{x}^A \in \mathcal{X}_f^+$ that is most desirable to the adversary. We focus on the general class of weighted ℓ_p ($0 < p \leq \infty$) cost functions relative to the target $\mathbf{x}^A$ given by

$$A_p^{(\mathbf{c})}\left(\mathbf{x} - \mathbf{x}^A\right) = \left(\sum_{d=1}^{D} c_d \left|x_d - x_d^A\right|^p\right)^{1/p} , \tag{1}$$

where $0 < c_d < \infty$ is the relative cost the adversary associates with altering the d^{th} feature. When the relative costs are uniform, $c_d = 1$ for all d, we use the simplified notation A_p to refer to the cost function. Similarly, when referring to a generic weighted cost function with weights $\mathbf{c}$, we use the notation $A^{(\mathbf{c})}$. We also consider the cases when some features have $c_d = 0$ (adversary doesn't care about the d^{th} feature) or $c_d = \infty$ (adversary requires the d^{th} feature to match x_d^A). We use $\mathcal{B}^C(A;\mathbf{y}) = \{\mathbf{x} \in \mathcal{X} \mid A(\mathbf{x} - \mathbf{y}) \leq C\}$ to denote the cost ball (or sublevel set) centered at $\mathbf{y}$ with cost no more than the threshold C. For instance, $\mathcal{B}^C\left(A_1;\mathbf{x}^A\right)$ is the set of instances that do not exceed an ℓ_1 cost of C from the target $\mathbf{x}^A$. For convenience, we also use $\mathcal{B}^C(A) \triangleq \mathcal{B}^C\left(A;\mathbf{x}^A\right)$ to denote the C-cost-ball of A re-centered at the adversary's target, $\mathbf{x}^A$, since we focus on costs relative to this instance. Unless stated otherwise, we take "ℓ_1 cost" to mean a weighted ℓ_1 cost in the sequel.

Unfortunately, ℓ_p costs do not include many interesting costs such as string edit distances for spam and other real-world settings, such as the intrusion detection example from above where there may be no natural notion of distance between points. Nevertheless, the objective of this paper is not to provide practical evasion algorithms but rather to understand the theoretic capabilities of an adversary on the analytically tractable, albeit practically restrictive, family of ℓ_p costs. Weighted ℓ_1 costs are particularly appropriate for adversarial problems in which the adversary is interested in some features more than others and his cost is assessed based on the degree to which a feature is altered. Moreover, the ℓ_1-norm is a natural measure for a word-level edit distance for email spam, where larger weights model tokens that are more costly to remove (e.g., a payload URL). In Section 3, we focus on the weighted ℓ_1 costs studied by Lowd and Meek before exploring general ℓ_p costs in Section 4. In the latter case, our discussion focuses on uniform weights for ease of exposition, but the results also extend to the cost-sensitive case as presented for weighted ℓ_1 costs.

Lowd and Meek (2005) define *minimal adversarial cost (MAC)* of a classifier f to be

$$MAC(f,A) \triangleq \inf_{\mathbf{x} \in \mathcal{X}_f^-} \left[A\left(\mathbf{x} - \mathbf{x}^A\right)\right] ;$$

that is, the greatest lower bound on the cost obtained by any negative instance. They further define a data point to be an ε-approximate *instance of minimal adversarial cost (ε-IMAC)* if it is a negative instance with a cost no more than a factor $(1+\varepsilon)$ of the *MAC*; that is, every ε-*IMAC* is a member of

the set

$$\varepsilon\text{-}IMAC(f,A) \triangleq \left\{ \mathbf{x} \in \mathcal{X}_f^- \;\middle|\; A\left(\mathbf{x}-\mathbf{x}^A\right) \leq (1+\varepsilon)\cdot \mathrm{MAC}(f,A) \right\} \quad . \tag{2}$$

The adversary's goal is to find an ε-*IMAC* efficiently, while issuing as few queries as possible.

2.2 Search Terminology

The notion of near optimality introduced in Equation (2) is that of *multiplicative optimality*; that is, an ε-*IMAC* must have a cost within a factor of $(1+\varepsilon)$ of the *MAC*. However, the results of this paper can also be immediately adapted for *additive optimality* in which we seek instances with cost no more than $\eta > 0$ *greater* than the *MAC*. To differentiate between these notions of optimality, we will use the notation ε-$IMAC^{(*)}$ to refer to the set in Equation (2) and define an analogous set η-$IMAC^{(+)}$ for additive optimality as

$$\eta\text{-}IMAC^{(+)}(f,A) \triangleq \left\{ \mathbf{x} \in \mathcal{X}_f^- \;\middle|\; A\left(\mathbf{x}-\mathbf{x}^A\right) \leq \eta + \mathrm{MAC}(f,A) \right\} \quad . \tag{3}$$

We use the terms ε-$IMAC^{(*)}$ and η-$IMAC^{(+)}$ to refer both to the sets defined in Equation (2) and (3) as well as the members of these sets—the usage will be clear from the context.

Either notion of optimality allows us to efficiently use bounds on the *MAC* to find an ε-$IMAC^{(*)}$ or an η-$IMAC^{(+)}$. Suppose there is a negative instance, $\mathbf{x}^-$, with cost C^-, and there is a $C^+ > 0$ such that all instances with cost no more than C^+ are positive; that is, $C^+ \leq MAC(f,A) \leq C^-$. Then the negative instance $\mathbf{x}^-$ is ε-multiplicatively optimal if $C^-/C^+ \leq (1+\varepsilon)$ whereas it is η-additively optimal if $C^- - C^+ \leq \eta$. In the sequel, we will consider algorithms that can achieve either additive or multiplicative optimality via binary search. Namely, if the adversary can determine whether an intermediate cost establishes a new upper or lower bound on the *MAC*, then binary search strategies can iteratively reduce the t^{th} gap between any bounds C_t^- and C_t^+ with the fewest steps. We now provide common terminology for the binary search and in Section 3 we use convexity to establish a new bound at each iteration.

Lemma 1 *If an algorithm can provide bounds* $0 < C^+ \leq MAC(f,A) \leq C^-$*, then this algorithm has achieved* $(C^- - C^+)$*-additive optimality and* $(\frac{C^-}{C^+} - 1)$*-multiplicative optimality.*

In the t^{th} iteration of an additive binary search, the *additive gap* between the t^{th} bounds, C_t^- and C_t^+, is given by $G_t^{(+)} = C_t^- - C_t^+$ with $G_0^{(+)}$ defined accordingly by the initial bounds $C_0^- = C^-$ and $C_0^+ = C^+$. The search uses a proposal step of $C_t = (C_t^- + C_t^+)/2$, a stopping criterion of $G_t^{(+)} \leq \eta$ and achieves η-additive optimality in

$$L_\eta^{(+)} = \left\lceil \log_2 \left[\frac{G_0^{(+)}}{\eta} \right] \right\rceil$$

steps. In fact, binary search has the best worst-case query complexity for achieving η-additive optimality.

Binary search can also be used for multiplicative optimality by searching in exponential space. Assuming that $C^- \geq C^+ > 0$, we can rewrite our upper and lower bounds as $C^- = 2^a$ and $C^+ = 2^b$, and thus the multiplicative optimality condition becomes $a - b \leq \log_2(1+\varepsilon)$; that is, an additive optimality condition. Thus, binary search on the exponent achieves ε-multiplicative optimality and

does so with the best worst-case query complexity. The *multiplicative gap* of the t^{th} iteration is $G_t^{(*)} = C_t^- / C_t^+$ with $G_0^{(*)}$ defined accordingly by the initial bounds C_0^- and C_0^+. The t^{th} query is $C_t = \sqrt{C_t^- \cdot C_t^+}$, the stopping criterion is $G_t^{(*)} \leq 1+\varepsilon$ and the search achieves ε-multiplicative optimality in

$$L_\varepsilon^{(*)} = \left\lceil \log_2 \left\lceil \frac{\log_2 \left(G_0^{(*)} \right)}{\log_2 (1+\varepsilon)} \right\rceil \right\rceil \tag{4}$$

steps. Although both additive and multiplicative criteria are related, there are two differences between these notions of optimality.

First, multiplicative optimality only makes sense when C_0^+ is strictly positive whereas additive optimality can still be achieved if $C_0^+ = 0$. Taking $C_0^+ > 0$ is equivalent to assuming that $\mathbf{x}^A$ is in the interior of $\mathcal{X}_f^+$ (a requirement for our algorithms to achieve multiplicative optimality). Otherwise, when $\mathbf{x}^A$ is on the boundary of $\mathcal{X}_f^+$, there is no ε-*IMAC*$^{(*)}$ for any $\varepsilon > 0$ unless there is some point $\mathbf{x}^* \in \mathcal{X}_f^-$ with 0 cost. Practically though, the need for a lower bound is a minor hindrance—as we demonstrate in Section 3.1.3, there is an algorithm that can efficiently establish a lower bound C_0^+ for any ℓ_p cost if such a lower bound exists.

Second, the additive optimality criterion is not *scale invariant* (i.e., any instance $\mathbf{x}^\dagger$ that satisfies the optimality criterion for cost A also satisfies it for $A'(\mathbf{x}) = s \cdot A(\mathbf{x})$ for any $s > 0$) whereas multiplicative optimality is scale invariant. Additive optimality is, however, *shift invariant* (i.e., any instance $\mathbf{x}^\dagger$ that satisfies the optimality criterion for cost A also satisfies it for $A'(\mathbf{x}) = s + A(\mathbf{x})$ for any $s \geq 0$) whereas multiplicative optimality is not. Scale invariance is more salient in near-optimal evasion because if the cost function is also scale invariant (all proper norms are) then the optimality condition is invariant to a rescaling of the underlying feature space; for example, a change in units for all features. Thus, multiplicative optimality is a unitless notion of optimality whereas additive optimality is not.

The following result states that additive optimality's lack of scale invariance allows for the feature space to be arbitrarily rescaled until any fixed level of additive optimality can no longer be achieved; that is, the units of the cost determine whether a particular level of additive accuracy can be achieved. By contrast multiplicative costs are unitless.

Proposition 2 *Consider any hypothesis space $\mathcal{F}$, target instance $\mathbf{x}^A$ and cost function A. If there exists some $\varepsilon > 0$ such that no efficient query-based algorithm can find an ε-IMAC$^{(*)}$ for any $0 < \varepsilon \leq \varepsilon$, then there is no efficient query-based algorithm that can find an η-IMAC$^{(+)}$ for any $0 < \eta \leq \varepsilon \cdot MAC(f,A)$. In particular consider a sequence of classifiers f_n admitting unbounded MACs, and a sequence $\varepsilon_n > 0$ such that $1/\varepsilon_n = o(MAC(f_n,A))$. Then if no general algorithm can efficiently find an ε_n-IMAC$^{(*)}$ on each f_n then no general algorithm can efficiently find an η_n-IMAC$^{(+)}$ for $\eta_n \to \infty$.*

Proof Consider any classifier $f \in \mathcal{F}$ such that $MAC(f,A) > 0$. Suppose there exists some $\mathbf{x} \in \eta$-*IMAC*$^{(+)}$ for some $\eta > 0$. Let $\varepsilon = \eta / MAC(f,A)$ then by definition

$$A\left(\mathbf{x} - \mathbf{x}^A\right) \leq \eta + MAC(f,A) = (1+\varepsilon) MAC(f,A) \quad ,$$

implying that $\mathbf{x} \in \varepsilon$-*IMAC*$^{(*)}$. Then by the contrapositive, if no ε-*IMAC*$^{(*)}$ can be efficiently found for any $0 < \varepsilon \leq \varepsilon$, then no η-*IMAC*$^{(+)}$ can be efficiently found for any $0 < \eta \leq \varepsilon \cdot MAC(f,A)$. The last result is an immediate corollary. ■

The last statement is, in fact, applicable to many common settings. For instance, for any of the weighted ℓ_p costs (with $0 < p \leq \infty$ and $0 < c_d < \infty$ for all d) the family of linear classifiers and the family of hypersphere classifiers are both sufficiently diverse to yield such a sequence of classifiers that admit unbounded *MAC*s as required by the last statement. Thus, the family of convex-inducing classifiers can also yield such a sequence. Moreover, as we show in Section 4, there are indeed ℓ_p costs for which there exists $\varepsilon > 0$ such that no efficient query-based algorithm can find an ε-*IMAC*$^{(*)}$ for any $0 < \varepsilon \leq \varepsilon$. The consequence of this is that there is no general algorithm capable of achieving additive optimality for any fixed η with respect to the convex-inducing classifiers for these ℓ_p costs.

For the remainder of this paper, we will address ε-multiplicative optimality for an ε-*IMAC* (except where explicitly noted) and define $L_\varepsilon = L_\varepsilon^{(*)}$ and $G_t = G_t^{(*)}$. Nonetheless, our algorithms can be immediately adapted to additive optimality by simply changing the proposal step, stopping condition, and the definitions of $L_\varepsilon^{(*)}$ and G_t; the binary searches for additive and multiplicative optimality differ in their proposal steps and stopping criteria only. Finally, while we express query complexity in the sequel in terms of multiplicative L_ε, note that $L_\varepsilon^{(*)} = \Theta(\log \frac{1}{\varepsilon})$ and so in this way our query complexities can be rewritten to directly depend on ε.

2.3 Near-Optimal Evasion

Lowd and Meek (2005) introduce the concept of *adversarial classifier reverse engineering (ACRE) learnability* to quantify the difficulty of finding an ε-*IMAC* instance for a particular family of classifiers, $\mathcal{F}$, and a family of adversarial costs, $\mathcal{A}$. Using our notation, their definition of *ACRE* ε-learnable is

> A set of classifiers $\mathcal{F}$ is *ACRE* ε-learnable under a set of cost functions $\mathcal{A}$ if an algorithm exists such that for all $f \in \mathcal{F}$ and $A \in \mathcal{A}$, it can find an $\mathbf{x} \in \varepsilon$-*IMAC*$(f, A)$ using only polynomially-many membership queries in terms of the dimension D, the encoded size of f, and the encoded size of $\mathbf{x}^+$ and $\mathbf{x}^-$.

In this definition, Lowd and Meek use encoded size to refer to the length of the string of digits used to encode f, $\mathbf{x}^+$, and $\mathbf{x}^-$. In generalizing their result, we slightly alter their definition of query complexity. First, to quantify query complexity we use only the dimension, D, and the number of steps, L_ε, required by a univariate binary search to narrow the gap to within the desired accuracy. By including L_ε in our definition of query complexity, we do not require the encoded size of $\mathbf{x}^+$ and $\mathbf{x}^-$ since L_ε implicitly captures the size of the distance between these points as discussed above. Second, we assume the adversary only has two initial points $\mathbf{x}^- \in \mathcal{X}_f^-$ and $\mathbf{x}^A \in \mathcal{X}_f^+$ (the original setting used a third $\mathbf{x}^+ \in \mathcal{X}_f^+$): we restrict our setting to the case of $\mathbf{x}^+ = \mathbf{x}^A$, yielding simpler search procedures.[2] Finally, our algorithms do not reverse engineer the decision boundary, so "ACRE" would be a misnomer here. Instead we refer to the overall problem as *Near-Optimal Evasion* and replace *ACRE* ε-learnable with the following definition of ε-*IMAC* searchable.

> A family of classifiers $\mathcal{F}$ is ε-*IMAC searchable* under a family of cost functions $\mathcal{A}$ if for all $f \in \mathcal{F}$ and $A \in \mathcal{A}$, there is an algorithm that finds some $\mathbf{x} \in \varepsilon$-*IMAC*$(f, A)$

2. As is apparent in our algorithms, using $\mathbf{x}^+ = \mathbf{x}^A$ makes the attacker less covert since it is significantly easier to infer the attacker's intentions based on their queries. Covertness is not an explicit goal in ε-*IMAC* search, but it would be a requirement of many real-world attackers. However, since our goal is not to design real attacks but rather analyze the best possible attack so as to understand our classifier's vulnerabilities, covertness can be ignored.

using polynomially-many membership queries in D and L_ε. We will refer to such an algorithm as *efficient*.

Our definition does not include the encoded size of the classifier, f, because our approach to near-optimal evasion does not reverse engineer the classifier's parameters. Unlike Lowd and Meek's approach for continuous spaces, our algorithms construct queries to provably find an ε-*IMAC* without reverse engineering the classifier's decision boundary; that is, estimating the decision surface of f or estimating the parameters that specify it. Efficient query-based reverse engineering for $f \in \mathcal{F}$ is sufficient for minimizing A over the estimated negative space. However, generally reverse engineering is an expensive approach for near-optimal evasion, requiring query complexity that is exponential in the feature space dimension for general convex classes (Rademacher and Goyal, 2009), while finding an ε-*IMAC* need not be as we demonstrate in this paper.[3] In fact, the requirements for finding an ε-*IMAC* differ significantly from the objectives of reverse-engineering approaches such as active learning. Both approaches use queries to reduce the size of version space $\hat{\mathcal{F}} \subset \mathcal{F}$; that is, the set of classifiers consistent with the adversary's membership queries. However reverse-engineering approaches minimize the expected number of disagreements between members of $\hat{\mathcal{F}}$. To find an ε-*IMAC*, by contrast, we need only provide a single instance, $\mathbf{x}^\dagger \in \varepsilon\text{-}IMAC(f,A)$, for all $f \in \hat{\mathcal{F}}$, while leaving the classifier largely unspecified; that is, we need to show that

$$\bigcap_{f \in \hat{\mathcal{F}}} \varepsilon\text{-}IMAC(f,A) \neq \emptyset \ .$$

This objective allows the classifier to be unspecified in much of $\mathcal{X}$. We present algorithms for ε-*IMAC* search on a family of classifiers that generally cannot be efficiently reverse engineered—the queries we construct necessarily elicit an ε-*IMAC* only; the classifier itself will be underspecified in large regions of $\mathcal{X}$ so our techniques do not reverse engineer the classifier. Similarly, for linear classifiers in Boolean spaces, Lowd and Meek demonstrated an efficient algorithm for near-optimal evasion that does not reverse engineer the classifier—it too searches directly for an ε-*IMAC* and it shows that this family is 2-*IMAC* searchable for ℓ_1 costs with uniform feature weights, $\mathbf{c}$.

3. Evasion of Convex Classes for ℓ_1 Costs

We generalize ε-*IMAC* searchability to the family of *convex-inducing classifiers* $\mathcal{F}^{\text{convex}}$ that partition the feature space $\mathcal{X}$ into a positive and negative class, one of which is convex. The convex-inducing classifiers include the linear classifiers studied by Lowd and Meek (2005), anomaly detectors using bounded PCA (Lakhina et al., 2004) and using hyper-sphere boundaries (Bishop, 2006), one-class classifiers that predict anomalies by thresholding the log-likelihood of a log-concave (or uni-modal) density function, and quadratic classifiers with a decision function of the form $\mathbf{x}^\top \mathbf{A}\mathbf{x} + \mathbf{b}^\top \mathbf{x} + c \geq 0$ if $\mathbf{A}$ is semidefinite (see Boyd and Vandenberghe, 2004, Chapter 3). The convex-inducing classifiers also include bodies such as any intersections of a countable number of halfspaces, cones, or balls.

Restricting $\mathcal{F}$ to be the family of convex-inducing classifiers simplifies ε-*IMAC* search. In our approach to this problem, we divide $\mathcal{F}^{\text{convex}}$, the family of convex-inducing classifiers, into

3. Lowd and Meek (2005) also previously showed that the reverse-engineering technique of finding a feature's sign witness is NP-complete for linear classifiers with Boolean features but also that this family was nonetheless 2-*IMAC* searchable.

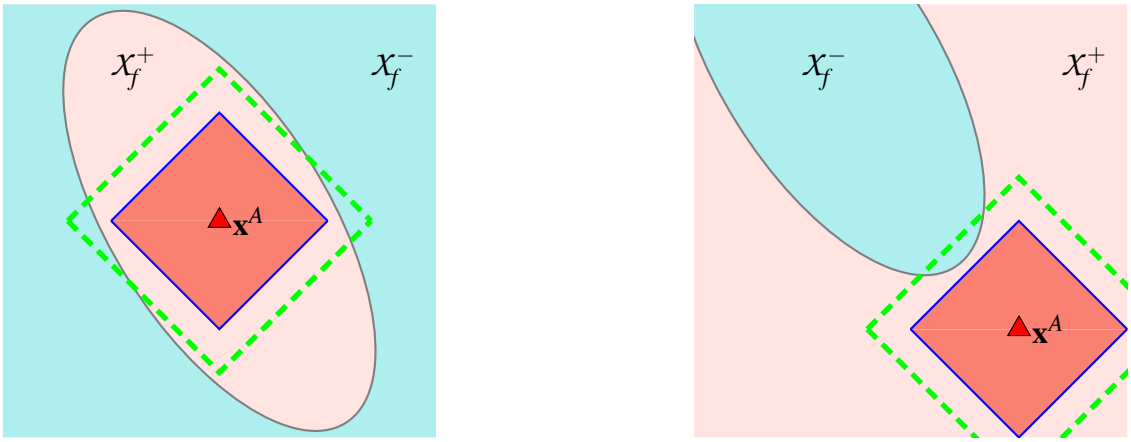


Figure 1: Geometry of convex sets and ℓ_1 balls. (a) If the positive set $\mathcal{X}_f^+$ is convex, finding an ℓ_1 ball contained within $\mathcal{X}_f^+$establishes a lower bound on the cost, otherwise at least one of the ℓ_1 ball's corners witnesses an upper bound. (b) If the negative set $\mathcal{X}_f^-$ is convex, we can establish upper and lower bounds on the cost by determining whether or not an ℓ_1 ball intersects with $\mathcal{X}_f^-$, but this intersection need not include any corner of the ball.

$\mathcal{F}^{\text{convex,'}-\text{'}}$ and $\mathcal{F}^{\text{convex,'}+\text{'}}$ corresponding to the classifiers that induce a convex set $\mathcal{X}_f^-$ or $\mathcal{X}_f^+$, respectively (of course, linear classifiers belong to both). When the negative class $\mathcal{X}_f^-$ is convex (i.e., $f \in \mathcal{F}^{\text{convex,'}-\text{'}}$), the problem reduces to minimizing a (convex) function A constrained to a convex set—if $\mathcal{X}_f^-$ were known to the adversary, then this would correspond to solving a convex program. When the positive class $\mathcal{X}_f^+$ is convex (i.e., $f \in \mathcal{F}^{\text{convex,'}+\text{'}}$), however, our task is to minimize the convex function A outside of a convex set; this is generally a hard problem (cf. Section 4.1.4 where we show that minimizing an ℓ_2 cost can require exponential query complexity). Nonetheless for certain cost functions A, it is easy to determine whether a particular cost ball $\mathcal{B}^C(A)$ is completely contained within a convex set. This leads to efficient approximation algorithms.

We construct efficient algorithms for *query-based* optimization of the (weighted) ℓ_1 cost $A_1^{(\mathbf{c})}$ of Equation (1) for the family of convex-inducing classifiers. There is an asymmetry to this problem depending on whether the positive or negative class is convex as illustrated in Figure 1. When the positive set is convex, determining whether the ℓ_1 ball $\mathcal{B}^C(A_1^{(\mathbf{c})})$ is a subset of $\mathcal{X}_f^+$ only requires querying the vertices of the ball as depicted in Figure 1(a). When the negative set is convex, determining whether $\mathcal{B}^C(A_1^{(\mathbf{c})}) \cap \mathcal{X}_f^- = \emptyset$ is non-trivial since the intersection need not occur at a vertex as depicted in Figure 1(b). We present an efficient algorithm for optimizing (weighted) ℓ_1 costs when $\mathcal{X}_f^+$ is convex and a polynomial randomized algorithm for optimizing any convex cost when $\mathcal{X}_f^-$ is convex. In both cases, we consider only convex sets with non-empty interiors. The algorithms we present achieve multiplicative optimality via the binary search strategies discussed in the previous section. In the sequel, we use Equation (4) to define L_ε and $C_0^- = A_I^{(\mathbf{c})}\left(\mathbf{x}^- - \mathbf{x}^A\right)$ as an initial upper bound on the *MAC*. We also assume there is some $C_0^+ > 0$ that lower bounds the *MAC*.

3.1 ε-*IMAC* Search for a Convex $\mathcal{X}_f^+$

Solving the ε-*IMAC* Search problem when $f \in \mathcal{F}^{\text{convex,'+'}}$ is hard in the general case of a convex cost A. Here we introduce algorithms for the ℓ_1 cost that solve the problem as a binary search. Namely, given initial costs C_0^+ and C_0^- that bound the *MAC*, our algorithm can efficiently determine whether $\mathcal{B}^{C_t}(A_1) \subset \mathcal{X}_f^+$ for any intermediate cost $C_t^+ < C_t < C_t^-$. If the ℓ_1 ball is contained in $\mathcal{X}_f^+$, then C_t becomes the new lower bound C_{t+1}^+. Otherwise C_t becomes the new upper bound C_{t+1}^-. Since our objective given in Equation (2) is to obtain multiplicative optimality, our steps will take the form $C_t = \sqrt{C_t^+ \cdot C_t^-}$. We now explain how we exploit the properties of the ℓ_1 ball and convexity of $\mathcal{X}_f^+$ to efficiently determine whether $\mathcal{B}^C(A_1)$ is a subset of $\mathcal{X}_f^+$ for any C. We also discuss practical aspects of our algorithm and extensions to other ℓ_p cost functions.

The existence of an efficient query algorithm relies on three facts: (1) $\mathbf{x}^A \in \mathcal{X}_f^+$; (2) every ℓ_1 cost C-ball centered at $\mathbf{x}^A$ intersects with $\mathcal{X}_f^-$ only if at least one of its vertices is in $\mathcal{X}_f^-$; and (3) C-balls of ℓ_1 costs only have $2 \cdot D$ vertices. The vertices of the ℓ_1 ball $\mathcal{B}^C(A_1)$ are axis-aligned instances differing from $\mathbf{x}^A$ in exactly one feature (e.g., the d^{th} feature) and can be expressed as

$$\mathbf{x}^A \pm \frac{C}{c_d} \cdot \delta_d \quad , \tag{5}$$

which belongs to the C-ball of our ℓ_1 cost (the coefficient $\frac{C}{c_d}$ normalizes for the weight c_d on the d^{th} feature). We now formalize the second fact as follows.

Lemma 3 *For all $C > 0$, if there exists some $\mathbf{x} \in \mathcal{X}_f^-$ that achieves a cost of $C = A_1^{(\mathbf{c})}\left(\mathbf{x} - \mathbf{x}^A\right)$, then there is some feature d such that a vertex of the form of Equation* (5) *is in $\mathcal{X}_f^-$ (and also achieves cost C by Equation 1).*

Proof Suppose not; then there is some $\mathbf{x} \in \mathcal{X}_f^-$ such that $A_1^{(\mathbf{c})}\left(\mathbf{x} - \mathbf{x}^A\right) = C$ and $\mathbf{x}$ has $M \geq 2$ features that differ from $\mathbf{x}^A$ (if $\mathbf{x}$ only differs in one feature it would be of the form of Equation 5). Let $\{d_1, \ldots, d_M\}$ be the differing features and let $b_{d_i} = \text{sign}\left(x_{d_i} - x_{d_i}^A\right)$ be the sign of the difference between $\mathbf{x}$ and $\mathbf{x}^A$ along the d_i-th feature. For each d_i, let $\mathbf{e}_{d_i} = \mathbf{x}^A + \frac{C}{c_{d_i}} \cdot b_{d_i} \cdot \delta_{d_i}$ be a vertex of the form of Equation (5) which has a cost C (from Equation 1). The M vertices $\mathbf{e}_{d_i}$ form an M-dimensional equi-cost simplex of cost C on which $\mathbf{x}$ lies; that is, $\mathbf{x} = \sum_{i=1}^M \alpha_i \cdot \mathbf{e}_{d_i}$ for some $0 \leq \alpha_i \leq 1$. If all $\mathbf{e}_{d_i} \in \mathcal{X}_f^+$, then the convexity of $\mathcal{X}_f^+$ implies that all points in their simplex are in $\mathcal{X}_f^+$ and so $\mathbf{x} \in \mathcal{X}_f^+$ which violates our premise. Thus, if any instance in $\mathcal{X}_f^-$ achieves cost C, there is always at least one vertex of the form Equation (5) in $\mathcal{X}_f^-$ that also achieves cost C. ■

As a consequence, if all such vertices of any C ball $\mathcal{B}^C(A_1)$ are positive, then all $\mathbf{x}$ with $A_1^{(\mathbf{c})}(\mathbf{x}) \leq C$ are positive thus establishing C as a lower bound on the *MAC*. Conversely, if any of the vertices of $\mathcal{B}^C(A_1)$ are negative, then C is an upper bound on *MAC*. Thus, by simultaneously querying all $2 \cdot D$ equi-cost vertices of $\mathcal{B}^C(A_1)$, we either establish C as a new lower or upper bound on the *MAC*. By performing a binary search on C we iteratively halve the multiplicative gap between our bounds until it is within a factor of $1 + \varepsilon$. This yields an ε-*IMAC* of the form of Equation (5).

A general form of this multiline search procedure is presented as Algorithm 1 and depicted in Figure 2. MULTILINESEARCH simultaneously searches along the directions in a set $\mathcal{W}$ of search

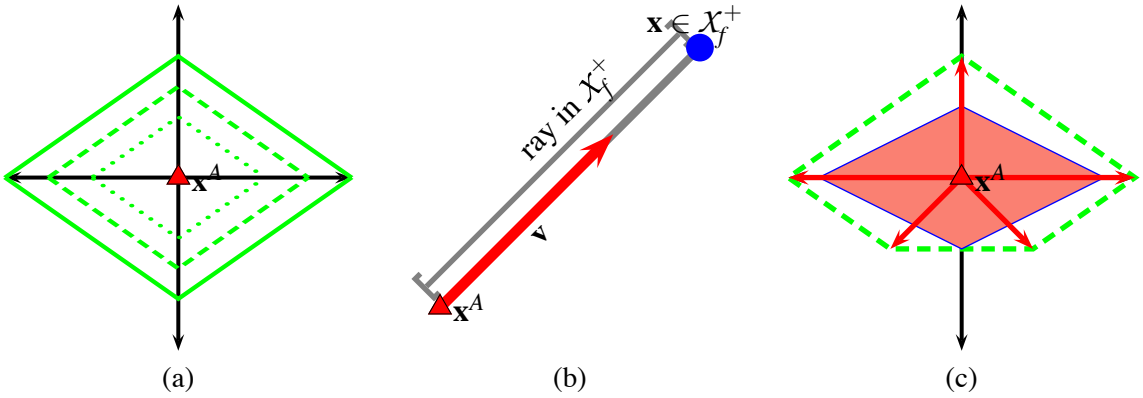


Figure 2: The geometry of search. (a) Weighted ℓ_1 balls are centered around the target $\mathbf{x}^A$ and have $2 \cdot D$ vertices; (b) Search directions in multi-line search radiate from $\mathbf{x}^A$ to probe specific costs; (c) In general, we leverage convexity of the cost function when searching to evade. By probing all search directions at a specific cost, the convex hull of the positive queries bounds the ℓ_1 cost ball contained within it.

directions that radiate from their origin at $\mathbf{x}^A$ and that are vectors of unit cost; that is, $A(\mathbf{w}) = 1$ for every $\mathbf{w} \in \mathcal{W}$. (We transform a given set of non-normalized search vectors $\{\mathbf{v}\}$ into unit search vectors by simply applying a normalization constant of $A(\mathbf{v})^{-1}$ to each vector.) At each step of MULTILINESEARCH, at most $|\mathcal{W}|$ queries are issued in order to construct a bounding shell (i.e., the convex hull of these queries will either form an upper or lower bound on the *MAC*) to determine whether $\mathcal{B}^C(A) \subset \mathcal{X}_f^+$. Once a negative instance is found at cost C, we cease further queries at cost C since a single negative instance is sufficient to establish a lower bound. We call this policy *lazy querying*—a practice that will lead to better bounds for a worst-case classifier. Further, when an upper bound is established for a cost C (a negative vertex is found), our algorithm prunes all directions that were positive at cost C. This pruning is sound; by convexity, these pruned directions are positive for all costs less than the new upper bound C on the *MAC*. Finally, by performing a binary search on the cost, MULTILINESEARCH finds an ε-*IMAC* with no more than $|\mathcal{W}| \cdot L_\varepsilon$ queries but at least $|\mathcal{W}| + L_\varepsilon$ queries. Thus, this algorithm is $O(|\mathcal{W}| \cdot L_\varepsilon)$.

It is worth noting that, in its present form, MULTILINESEARCH has two implicit assumptions. First, we assume all search directions radiate from a common origin, $\mathbf{x}^A$, and $A(\mathbf{0}) = 0$. Without this assumption, the ray-constrained cost function $A(s \cdot \mathbf{w})$ is still convex in $s \geq 0$ but not necessarily monotonic as required for binary search. Second, we assume the cost function A is a *positive homogeneous function* along any ray from $\mathbf{x}^A$; that is, $A(s \cdot \mathbf{w}) = |s| \cdot A(\mathbf{w})$. This assumption allows MULTILINESEARCH to scale its unit search vectors to achieve the same scaling of their cost. Although the algorithm could be adapted to eliminate these assumptions, the cost functions in Equation (1) satisfy both assumptions since they are norms centered at $\mathbf{x}^A$.

Algorithm 2 uses MULTILINESEARCH for ℓ_1 costs by taking $\mathcal{W}$ to be the vertices of the unit-cost ℓ_1 ball centered at $\mathbf{x}^A$. In this case, the search issues at most $2 \cdot D$ queries to determine whether $\mathcal{B}^C(A_1)$ is a subset of $\mathcal{X}_f^+$ and so Algorithm 2 is $O(L_\varepsilon \cdot D)$. However, MULTILINESEARCH does not rely on its directions being vertices of the ℓ_1 ball although those vertices are sufficient to span the ℓ_1 ball. Generally, MULTILINESEARCH is agnostic to the configuration of its search directions

Algorithm 1 MULTI-LINE SEARCH

MLS $\left(\mathcal{W}, \mathbf{x}^A, \mathbf{x}^-, C_0^+, C_0^-, \varepsilon\right)$
$\mathbf{x}^* \leftarrow \mathbf{x}^-$
$t \leftarrow 0$
while $C_t^- / C_t^+ > 1 + \varepsilon$ **do**
  $C_t \leftarrow \sqrt{C_t^+ \cdot C_t^-}$
  for all $\mathbf{e} \in \mathcal{W}$ **do**
    Query: $f_{\mathbf{e}}^t \leftarrow f\left(\mathbf{x}^A + C_t \cdot \mathbf{e}\right)$
    if $f_{\mathbf{e}}^t =$ '$-$' **then**
      $\mathbf{x}^* \leftarrow \mathbf{x}^A + C_t \cdot \mathbf{e}$
      Prune $\mathbf{i}$ from $\mathcal{W}$ if $f_{\mathbf{i}}^t =$ '$+$'
      break for-loop
    end if
  end for
  $C_{t+1}^+ \leftarrow C_t^+$ and $C_{t+1}^- \leftarrow C_t^-$
  if $\forall \mathbf{e} \in \mathcal{W}\ f_{\mathbf{e}}^t =$ '$+$' **then** $C_{t+1}^+ \leftarrow C_t$
  else $C_{t+1}^- \leftarrow C_t$
  $t \leftarrow t + 1$
end while
return: $\mathbf{x}^*$

Algorithm 2 CONVEX $\mathcal{X}_f^+$ SET SEARCH

ConvexSearch $\left(\mathbf{x}^A, \mathbf{x}^-, \mathbf{c}, \varepsilon, C^+\right)$
$D \leftarrow dim\left(\mathbf{x}^A\right)$
$C^- \leftarrow A^{(\mathbf{c})}\left(\mathbf{x}^- - \mathbf{x}^A\right)$
$\mathcal{W} \leftarrow \emptyset$
for $i = 1$ to D **do**
  $\mathbf{e}^i \leftarrow \frac{1}{c_i} \cdot \delta_i$
  $\mathcal{W} \leftarrow \mathcal{W} \cup \left\{\pm \mathbf{e}^i\right\}$
end for
return: *MLS* $\left(\mathcal{W}, \mathbf{x}^A, \mathbf{x}^-, C^+, C^-, \varepsilon\right)$

Figure 3: Algorithms for multi-line search. Algorithm 1 is a generic procedure for performing simultaneous binary searches along multiple search directions emanating from $\mathbf{x}^A$; each direction, $\mathbf{e} \in \mathcal{W}$, must be a unit-cost direction. Algorithm 2 uses this MULTILINESEARCH procedure to minimize weighted ℓ_1 costs when the positive class of a classifier is convex. For this procedure, every weight, c_i, must be on the range $(0, \infty)$ although extensions are discussed in Section 3.1.3.

and can be adapted for any set of directions that can provide a sufficiently tight bound on the cost using the convexity of $\mathcal{X}_f^+$(see Section 4.1.1 for the bounding requirements the search directions must satisfy). However, as we show in Section 4.1, the number of search directions required to adequately bound an ℓ_p cost ball for $p > 1$ can be exponential in D.

3.1.1 K-STEP MULTI-LINE SEARCH

Here we present a variant of the multi-line search algorithm that better exploits pruning to reduce the query complexity of Algorithm 1—we call this variant K-STEP MULTILINESEARCH. The MULTILINESEARCH algorithm consists of $2 \cdot |\mathcal{W}|$ simultaneous binary searches (a breadth-first strategy). This strategy prunes directions most effectively when the convex body is asymmetrically elongated relative to $\mathbf{x}^A$ but fails to prune for symmetrically rounded bodies. We could instead search each direction sequentially (a depth-first strategy) and still obtain a worst case of $O(L_\varepsilon \cdot D)$ queries. This strategy uses fewer queries to shrink the cost gap on symmetrically rounded bodies but is unable to do so for asymmetrically elongated bodies. We therefore propose an algorithm that mixes these strategies.

At each phase, the K-STEP MULTILINESEARCH (Algorithm 3) chooses a single direction $\mathbf{e}$ and queries it for K steps to generate candidate bounds B^- and B^+ on the *MAC*. The algorithm makes substantial progress towards reducing G_t without querying other directions (a depth-first strategy). It then iteratively queries all remaining directions at the candidate lower bound B^+ (a breadth-first strategy). Again we use lazy querying and stop as soon as a negative instance is found since B^+ is then no longer a viable lower bound. In this case, although the candidate bound is invalidated, we can still prune all directions that were positive at B^+. Thus, in every iteration, either the gap is substantially decreased or at least one search direction is pruned. We show that for $K = \lceil \sqrt{L_\varepsilon} \rceil$, the algorithm achieves a delicate balance between the usual breadth-first and depth-first approaches to attain a better worst-case complexity than either.

Theorem 4 *Algorithm 3 will find an* ε*-IMAC with at most* $O\left(L_\varepsilon + \sqrt{L_\varepsilon}|\mathcal{W}|\right)$ *queries when* $K = \lceil \sqrt{L_\varepsilon} \rceil$.

The proof of this theorem appears in Appendix A. As a consequence of Theorem 4, finding an ε-*IMAC* with Algorithm 3 for an ℓ_1 cost requires $O\left(L_\varepsilon + \sqrt{L_\varepsilon}D\right)$ queries. Further, Algorithm 2 can incorporate K-STEP MULTILINESEARCH directly by replacing its function calls to MULTILINESEARCH with K-STEP MULTILINESEARCH and using $K = \lceil \sqrt{L_\varepsilon} \rceil$.

3.1.2 LOWER BOUND

Here we find a lower bound on the number of queries required by any algorithm to find an ε-*IMAC* when $\mathcal{X}_f^+$ is convex for any convex cost function (e.g., Equation 1 for $p \geq 1$). Below we present a theorem that provides a lower bound for multiplicative optimality (for additive optimality, there is an analogous lower bound for any $0 < \eta < C_0^- - C_0^+$). Notably, since an ε-*IMAC* uses multiplicative optimality, we incorporate a bound $C_0^+ > 0$ on the *MAC* into our statement.

Theorem 5 *For any* $D > 0$, *any positive convex function* $A : \Re^D \to \Re^+$, *any initial bounds* $0 < C_0^+ < C_0^-$ *on the MAC, and* $0 < \varepsilon < \frac{C_0^-}{C_0^+} - 1$, *all algorithms must submit at least* $\max\{D, L_\varepsilon^{(*)}\}$ *membership queries in the worst case to be* ε*-multiplicatively optimal on* $\mathcal{F}^{\text{convex,'+'}}$.

The proof of this result is in Appendix B. In this theorem, we restrict ε to the interval $\left(0, \frac{C_0^-}{C_0^+} - 1\right)$ since, outside of this interval, the strategy is trivial. For $\varepsilon = 0$ no approximation algorithm terminates and for $\varepsilon \geq \frac{C_0^-}{C_0^+} - 1$, $\mathbf{x}^-$ is an ε-*IMAC*, so no queries are required.

Theorem 5 shows that ε-multiplicative optimality requires $\Omega(L_\varepsilon^{(*)} + D)$ queries. Thus, we see that our K-STEP MULTILINESEARCH algorithm (Algorithm 3) has close to the optimal query complexity for ℓ_1-costs with its $O(L_\varepsilon + \sqrt{L_\varepsilon}D)$ queries. This lower bound also applies to any ℓ_p cost with $p > 1$, but in Section 4 we show lower bounds for $p > 1$ that substantially improve this result.

3.1.3 SPECIAL CASES

Here we present a number of special cases that require minor modifications to Algorithms 1 and 3 primarily as preprocessing steps.

Revisiting Linear Classifiers: Lowd and Meek originally developed a method for reverse engineering linear classifiers for an ℓ_1 cost. First their method isolates a sequence of points from $\mathbf{x}^-$ to $\mathbf{x}^A$

Algorithm 3 *K*-STEP MULTI-LINE SEARCH

KMLS $\left(\mathcal{W}, \mathbf{x}^A, \mathbf{x}^-, C_0^+, C_0^-, \varepsilon, K\right)$
$\mathbf{x}^* \leftarrow \mathbf{x}^-$
$t \leftarrow 0$
while $C_t^- / C_t^+ > 1 + \varepsilon$ **do**
  Choose a direction $\mathbf{e} \in \mathcal{W}$
  $B^+ \leftarrow C_t^+$
  $B^- \leftarrow C_t^-$
  for K steps **do**
    $B \leftarrow \sqrt{B^+ \cdot B^-}$
    Query: $f_{\mathbf{e}} \leftarrow f\left(\mathbf{x}^A + B \cdot \mathbf{e}\right)$
    if $f_{\mathbf{e}} =$ '+' **then** $B^+ \leftarrow B$
    else $B^- \leftarrow B$ **and** $\mathbf{x}^* \leftarrow \mathbf{x}^A + B \cdot \mathbf{e}$
  end for
  for all $\mathbf{i} \in \mathcal{W} \setminus \{\mathbf{e}\}$ **do**
    Query: $f_{\mathbf{i}}^t \leftarrow f\left(\mathbf{x}^A + (B^+) \cdot \mathbf{i}\right)$
    if $f_{\mathbf{i}}^t =$ '−' **then**
      $\mathbf{x}^* \leftarrow \mathbf{x}^A + (B^+) \cdot \mathbf{i}$
      Prune $\mathbf{k}$ from $\mathcal{W}$ if $f_{\mathbf{k}}^t =$ '+'
      break for-loop
    end if
  end for
  $C_{t+1}^- \leftarrow B^-$
  if $\forall \mathbf{i} \in \mathcal{W}\ f_{\mathbf{i}}^t =$ '+' **then** $C_{t+1}^+ \leftarrow B^+$
  else $C_{t+1}^- \leftarrow B^+$
  $t \leftarrow t + 1$
end while
return: $\mathbf{x}^*$

Figure 4: Algorithm for multi-line search. It performs simultaneous binary searches along multiple unit search directions emanating from $\mathbf{x}^A$. Algorithm 3 is asympototically more efficient than Algorithm 1 when $K = \lceil \sqrt{L_\varepsilon} \rceil$ and can be used as a substitute for it in Algorithm 2.

that cross the classifier's boundary and then the method estimates the hyperplane's parameters using D binary line searches. However, as a consequence of the ability to efficiently minimize our objective when $\mathcal{X}_f^+$ is convex, we immediately have an alternative method for linear classifiers. Because linear classifiers are a special case of convex-inducing classifiers, Algorithm 2 can be applied, and our *K*-STEP MULTILINESEARCH algorithm improves on complexity of Lowd and Meek's reverse-engineering technique's $O(L_\varepsilon \cdot D)$ queries and applies to a broader family of classifiers.

While Algorithm 2 has superior complexity, it uses $2 \cdot D$ search directions rather than the D directions used in the approach of Lowd and Meek, which may require our technique to issue more queries in some practical settings. However, for some restrictive classifier families, it is also pos-

sible to eliminate search directions proved to be infeasible based on the current set of queries. For instance, given a set $\mathcal{W}$ of search directions, t queries $\{\mathbf{x}^i\}_{i=1}^t$ and their corresponding responses $\{y^i\}_{i=1}^t$, a search direction $\mathbf{e}$ can be eliminated from $\mathcal{W}$ if for all $C_t^+ \leq \alpha < C_t^-$ there does not exist any classifier $f \in \mathcal{F}$ consistent with all previous queries (i.e., $f(\mathbf{x}^-) = '-', f(\mathbf{x}^A) = '+'$ and for all $i \in \{1,\ldots,t\}, f(\mathbf{x}^i) = y^i$) that also satisfies $f(\alpha \cdot \mathbf{e}) = '-'$ and $f(\alpha \cdot \mathbf{i}) = '+'$ for every $\mathbf{i} \in \mathcal{W} \setminus \{\mathbf{e}\}$. That is, $\mathbf{e}$ is feasible if and only if it is the only search direction among the set of remaining search directions, $\mathcal{W}$, that would be classified as a negative for a cost α by some consistent classifier. Further, since subsequent queries only restrict the feasible space of α and the set of consistent classifiers $\hat{\mathcal{F}}$, pruning these infeasible directions is sound for the remainder of the search.

For restrictive families of convex-inducing classifiers, these feasibility conditions can be efficiently verified and may be used to prune search directions without issuing further queries. In fact, for the family of linear classifiers written as $f(\mathbf{x}) = \text{sign}(\mathbf{w}^\top \mathbf{x} + b)$ for a normal vector $\mathbf{w}$ and displacement b, the above conditions become a set of linear inequalities along with quadratic inequalities corresponding to the constraint involving search directions. This can be cast as the following optimization program with respect to α, $\mathbf{w}$ and b:

$$\begin{aligned}
\min_{\alpha,\mathbf{w},b} \quad & \alpha \cdot \mathbf{w}^\top \mathbf{e} + b \\
& \alpha \in [C_t^+, C_t^-) \\
& \mathbf{w}^\top \mathbf{x}^- + b \leq 0 \\
\text{s.t.} \quad & \mathbf{w}^\top \mathbf{x}^A + b \geq 0 \\
& y^i(\mathbf{w}^\top \mathbf{x}^i + b) \geq 0 \quad \forall\, i \in \{1,\ldots,t\} \\
& \alpha \cdot \mathbf{w}^\top \mathbf{i} + b \geq 0 \quad \forall\, \mathbf{i} \in \mathcal{W} \setminus \{\mathbf{e}\}.
\end{aligned}$$

If the resulting minimum is less than zero, direction $\mathbf{e}$ is feasible, otherwise, $\mathbf{e}$ can be pruned. Such programs can be efficiently solved and may allow the adversary to rapidly eliminate infeasible search directions without issuing additional queries. However, refining these pruning procedures further is beyond the scope of this paper.

Extending MULTILINESEARCH *Algorithms to Weights $c_d = \infty$ or $c_d = 0$:* In Algorithm 2, we reweighted the d^{th} axis-aligned directions by a factor $\frac{1}{c_d}$ to make unit cost vectors by implicitly assuming $c_d \in (0, \infty)$. The case of immutable features where $c_d = \infty$ is dealt with by simply removing those features from the set of search directions $\mathcal{W}$ used in the MULTILINESEARCH. In the case of useless or unconstrained features when $c_d = 0$, MULTILINESEARCH-like algorithms no longer ensure near-optimality because they implicitly assume that cost balls are bounded sets. If $c_d = 0$, then $\mathcal{B}^0(A)$ is no longer bounded and 0 cost can be achieved if $\mathcal{X}_f^-$ anywhere intersects the subspace spanned by the 0-cost features—this makes near-optimality unachievable unless a negative 0-cost instance can be found. In the worst case, such an instance could be arbitrarily far in any direction within the 0-cost subspace making search for such an instance intractable. Nonetheless, one possible search strategy is to assign all 0-cost features a non-zero weight that decays quickly toward 0 (e.g., $c_d = 2^{-t}$ in the t^{th} iteration) as we repeatedly rerun MULTILINESEARCH on the altered objective for T iterations. We will either find a negative instance that only alters 0-cost features (and hence is a 0-*IMAC*), or it terminates with a non-zero cost instance, which is an ε-*IMAC* if no 0-cost negative instances exist. This algorithm does not ensure near-optimality but may be suitable for practical settings using some fixed T runs.

Lack of an Initial Lower Cost Bound: Thus far, to find an ε-*IMAC* our algorithms have searched between initial bounds C_0^+ and C_0^-, but, in general, C_0^+ may not be known to a real-world adversary.

We now present the SPIRALSEARCH algorithm (Algorithm 4) that efficiently establishes a lower bound on the *MAC* if one exists. This algorithm performs a halving search on the exponent along a single direction to find a positive example, then queries the remaining directions at this candidate bound. Either the lower bound is verified or directions that were positive can be pruned for the remainder of the search.

Algorithm 4 SPIRAL SEARCH

SpiralSearch $\left(\mathcal{W}, \mathbf{x}^A, C_0^-\right)$
$t \leftarrow 0$ and $\mathcal{V} \leftarrow \emptyset$
repeat
  Choose a direction $\mathbf{e} \in \mathcal{W}$
  Remove $\mathbf{e}$ from $\mathcal{W}$ and $\mathcal{V} \leftarrow \mathcal{V} \cup \{\mathbf{e}\}$
  Query: $f_\mathbf{e} \leftarrow f\left(\mathbf{x}^A + C_0^- \cdot 2^{-2^t} \cdot \mathbf{e}\right)$
  if $f_\mathbf{e} = \text{'}-\text{'}$ **then**
    $\mathcal{W} \leftarrow \mathcal{W} \cup \{\mathbf{e}\}$ and $\mathcal{V} \leftarrow \emptyset$
    $t \leftarrow t + 1$
  end if
until $\mathcal{W} = \emptyset$
$C_0^+ \leftarrow C_0^- \cdot 2^{-2^t}$
if $t > 0$ **then** $C_0^- \leftarrow C_0^- \cdot 2^{-2^{t-1}}$
return: $(\mathcal{V}, C_0^+, C_0^-)$

Figure 5: Algorithm for establishing an initial lower bound on the cost.

At the t^{th} iteration of SPIRALSEARCH, a direction is selected and queried at the candidate lower bound of $(C_0^-)2^{-2^t}$. If the query is positive, that direction is added to the set $\mathcal{V}$ of directions consistent with the lower bound. Otherwise, all positive directions in $\mathcal{V}$ are pruned, a new upper bound is established, and the candidate lower bound is reduced with an exponentially decreasing exponent. By definition of the *MAC*, this algorithm will terminate after $t = \left\lceil \log_2 \log_2 \frac{C_0^-}{MAC(f,A)} \right\rceil$ iterations. Further, in this algorithm, multiple directions are probed only during iterations with positive queries and it makes at most one positive query for each direction. Thus, given that some lower bound $C_0^+ > 0$ does exist, SPIRALSEARCH will establish a lower bound with $O(L'_\varepsilon + D)$ queries, where L'_ε is given by Equation (4) defined using $C_0^+ = MAC(f,A)$; the largest possible lower bound.

This algorithm can be used as a precursor to any of the previous searches.[4] Upon completion, the upper and lower bounds it establishes have a multiplicative gap of $2^{2^{t-1}}$ for $t > 0$ or 2 for $t = 0$. From the definition of t provided above in terms of the *MAC*, MULTILINESEARCH can hence proceed using $L_\varepsilon = L'_\varepsilon$. Further, the search directions pruned by SPIRALSEARCH are also invalid for the subsequent MULTILINESEARCH so the set $\mathcal{V}$ returned by SPIRALSEARCH will be used as the initial set $\mathcal{W}$ for the subsequent search. Thus, the query complexity of the subsequent search is the same as if it had started with the best possible lower bound.

4. If no lower bound on the cost exists, no algorithm can find an ε-*IMAC*. As presented, this algorithm would not terminate, but in practice the search would be terminated after sufficiently many iterations.

Lack of a Negative Example: Our algorithms can also naturally be adapted to the case when the adversary has no negative example $\mathbf{x}^-$. This is accomplished by querying ℓ_1 balls of doubly exponentially increasing cost until a negative instance is found. During the t^{th} iteration, we probe along every search direction at a cost $(C_0^+)2^{2^t}$; either all probes are positive (and we have a new lower bound) or at least one is negative and we can terminate the search. Once a negative example is located (having probed for T iterations), we must have $(C_0^+)2^{2^{T-1}} < MAC(f,A) \leq (C_0^+)2^{2^T}$; thus, $T = \left\lceil \log_2 \log_2 \frac{MAC(f,A)}{C_0^+} \right\rceil$. We can subsequently perform MULTILINESEARCH with $C_0^+ = 2^{2^{T-1}}$ and $C_0^- = 2^{2^T}$; that is, $\log_2 G_0 = 2^{T-1}$. This precursor step requires at most $|\mathcal{W}| \cdot T$ queries to initialize the MULTILINESEARCH algorithm with a gap such that $L_\varepsilon = \left\lceil (T-1) + \log_2 \frac{1}{\log_2(1+\varepsilon)} \right\rceil$ according to Equation (4).

If there is neither an initial upper bound or lower bound, we proceed by probing each search direction at unit cost using an additional $|\mathcal{W}|$ queries. We will subsequently have either an upper or lower bound and can proceed accordingly.

3.2 ε-*IMAC* Learning for a Convex $\mathcal{X}_f^-$

Here, we minimize a convex cost function A with bounded cost balls (we focus on weighted ℓ_1 costs in Equation 1) when the feasible set $\mathcal{X}_f^-$ is convex. Any convex function can be efficiently minimized within a known convex set (e.g., using an ellipsoid or interior point method; see Boyd and Vandenberghe 2004). However, in our problem, the convex set is only accessible via membership queries. We use a randomized polynomial algorithm of Bertsimas and Vempala (2004) to minimize the cost function A given an initial point $\mathbf{x}^- \in \mathcal{X}_f^-$. For any fixed cost, C^t, we use their algorithm to determine (with high probability) whether $\mathcal{X}_f^-$ intersects with $\mathcal{B}^{C^t}(A)$; that is, whether C^t is a new lower or upper bound on the *MAC*. With high probability, this approach can find an ε-*IMAC* in no more than L_ε repetitions using binary search. The following theorem is the main result of this section.

Theorem 6 *Let cost function A be convex and have bounded balls; that is, bounded sublevel sets. Let the feasible set $\mathcal{X}_f^-$ be convex and assume there is some $r > 0$ and $\mathbf{y} \in \mathcal{X}_f^-$ such that $\mathcal{X}_f^-$ contains the cost ball $\mathcal{B}^r(A;\mathbf{y})$. Then given access to an oracle returning separating hyperplanes for the A cost balls, Algorithm 7 will find an ε-IMAC using $O^*\left(D^5\right)$ queries with high probability.*[5]

The proof of this result is outlined in the remainder of this section, and is based on Bertsimas and Vempala (2004, Theorem 14). We first introduce their randomized ellipsoid algorithm, then we elaborate on their procedure for efficient sampling from a convex body, and finally we present our application to optimization. In this section, we focus only on weighted ℓ_1 costs (Equation 1) and return to more general cases in Section 4.2.

3.2.1 INTERSECTION OF CONVEX SETS

Bertsimas and Vempala present a query-based procedure for determining whether two convex sets (e.g., $\mathcal{X}_f^-$ and the A_1-ball of radius C^t) intersect. Their INTERSECTSEARCH procedure, which we

5. $O^*(\cdot)$ denotes the standard complexity notation $O(\cdot)$ without logarithmic terms. The dependence on ε is in these logarithmic terms, see Bertsimas and Vempala (2004) for details.

Algorithm 5 INTERSECT SEARCH

IntersectSearch $\left(\mathcal{P}^0, Q = \left\{\mathbf{x}^j \in \mathcal{P}^0\right\}, \mathbf{x}^A, C\right)$
for $s = 1$ to T **do**
(1) Generate $2N$ samples $\left\{\mathbf{x}^j\right\}_{j=1}^{2N}$
Choose $\mathbf{x}$ from Q
$\mathbf{x}^j \leftarrow$ *HitRun* $\left(\mathcal{P}^{s-1}, Q, \mathbf{x}^j\right)$
(2) If any $\mathbf{x}^j$, $A\left(\mathbf{x}^j - \mathbf{x}^A\right) \leq C$ terminate the for-loop
(3) Put samples into 2 sets of size N
$\mathcal{R} \leftarrow \left\{\mathbf{x}^j\right\}_{j=1}^{N}$ and $\mathcal{S} \leftarrow \left\{\mathbf{x}^j\right\}_{j=N+1}^{2N}$
(4) $\mathbf{z}^s \leftarrow \frac{1}{N}\sum_{\mathbf{x}^j \in \mathcal{R}} \mathbf{x}^j$
(5) Compute $\mathcal{H}_{\mathbf{z}^s}$ using Equation (7)
(6) $\mathcal{P}^s \leftarrow \mathcal{P}^{s-1} \cap \mathcal{H}_{\mathbf{z}^s}$
(7) Keep samples in $\mathcal{P}^s$
$Q \leftarrow \{\mathbf{x} \in \mathcal{S} \wedge \mathbf{x} \in \mathcal{P}^s\}$
end for
Return: the found $[\mathbf{x}_j, \mathcal{P}^s, Q]$; or No Intersect

Algorithm 6 HIT-AND-RUN

HitRun $\left(\mathcal{P}, \left\{\mathbf{y}^j\right\}, \mathbf{x}^0\right)$
for $i = 1$ to K **do**
(1) Choose a random direction:
$\nu_j \sim \mathrm{N}(0,1)$
$\mathbf{v} \leftarrow \sum_j \nu_j \cdot \mathbf{y}^j$
(2) Sample uniformly along $\mathbf{v}$ using rejection sampling:
Choose $\hat{\omega}$ s.t. $\mathbf{x}^{i-1} + \hat{\omega} \cdot \mathbf{v} \notin \mathcal{P}$
repeat
$\omega \sim$ *Unif* $(0, \hat{\omega})$
$\mathbf{x}^i \leftarrow \mathbf{x}^{i-1} + \omega \cdot \mathbf{v}$
$\hat{\omega} \leftarrow \omega$
until $\mathbf{x}^i \in \mathcal{P}$
end for
Return: $\mathbf{x}^K$

Figure 6: Algorithms used for the randomized ellipsoid algorithm of Bertsimas and Vempala. INTERSECTSEARCH is used to find the intersection between a pair of convex sets: $\mathcal{P}^0$ is queryable and $\mathcal{B}$ provides has a separating hyperplane from Equation (7). Note that the ROUNDING algorithm discussed in Section 3.2.2 can be used as a preprocessing step so that $\mathcal{P}^0$ is near-isotropic and to obtain the samples for Q. The HIT-AND-RUN algorithm is used to efficiently obtain uniform samples from a bounded near-isotropic convex set, $\mathcal{P}$, based on a set of uniform samples from it, $\left\{\mathbf{y}^j\right\}$, and a starting point $\mathbf{x}^0$.

present as Algorithm 5, is a randomized ellipsoid method for determining whether there is an intersection between two bounded convex sets: $\mathcal{P}$ is only accessible through membership queries and $\mathcal{B}$ provides a separating hyperplane for any point outside it (for our problem these sets correspond to $\mathcal{X}_f^-$ and $\mathcal{B}^{C^t}(A_1)$ respectively). They use efficient query-based approaches to uniformly sample from $\mathcal{P}$ to obtain sufficiently many samples such that cutting $\mathcal{P}$ through the centroid of these samples with a separating hyperplane from $\mathcal{B}$ significantly reduces the volume of $\mathcal{P}$ with high probability. Their technique thus constructs a sequence of progressively smaller feasible sets $\mathcal{P}^s \subset \mathcal{P}^{s-1}$ until either the algorithm finds a point in $\mathcal{P} \cap \mathcal{B}$ or it is highly likely that the intersection is empty.

Our problem reduces to finding the intersection between $\mathcal{X}_f^-$ and $\mathcal{B}^{C^t}(A_1)$. Though $\mathcal{X}_f^-$ may be unbounded, we are minimizing a cost with bounded cost balls, so we can instead use the set $\mathcal{P}^0 = \mathcal{X}_f^- \cap \mathcal{B}^{2R}(A_1; \mathbf{x}^-)$ (where $R = A\left(\mathbf{x}^- - \mathbf{x}^A\right) > C^t$), which is a convex bounded subset of $\mathcal{X}_f^-$. Since, by the triangle inequality, the ball $\mathcal{B}^{2R}(A_1; \mathbf{x}^-)$ centered at $\mathbf{x}^-$ envelops all of $\mathcal{B}^{C^t}\left(A_1; \mathbf{x}^A\right)$ centered at $\mathbf{x}^A$, the set $\mathcal{P}^0$ contains the entirety of the desired intersection, $\mathcal{X}_f^- \cap \mathcal{B}^{C^t}(A_1)$, if it exists. We also assume that there is some $r > 0$ such that there is an r-ball contained in the convex set $\mathcal{X}_f^-$; that is, there exists $\mathbf{y} \in \mathcal{X}_f^-$ such that the r-ball centered at $\mathbf{y}$, $\mathcal{B}^r(A_1; \mathbf{y})$, is a subset of $\mathcal{X}_f^-$. This assumption both ensures that $\mathcal{X}_f^-$ has a non-empty interior (a requirement for the HIT-AND-RUN

Algorithm 7 CONVEX $\mathcal{X}_f^-$ SET SEARCH

SetSearch $\left(\mathcal{P}, Q = \left\{\mathbf{x}^j \in \mathcal{P}\right\}, \mathbf{x}^A, \mathbf{x}^-, C_0^+, C_0^-, \varepsilon\right)$
$\mathbf{x}^* \leftarrow \mathbf{x}^-$ and $t \leftarrow 0$
while $C_t^- / C_t^+ > 1 + \varepsilon$ **do**
  $C_t \leftarrow \sqrt{C_t^- \cdot C_t^+}$
  $[\mathbf{x}^*, \mathcal{P}', Q'] \leftarrow$ *IntersectSearch* $\left(\mathcal{P}, Q, \mathbf{x}^A, C_t\right)$
  if intersection found **then**
    $C_{t+1}^- \leftarrow A\left(\mathbf{x}^* - \mathbf{x}^A\right)$ and $C_{t+1}^+ \leftarrow C_t^+$
    $\mathcal{P} \leftarrow \mathcal{P}'$ and $Q \leftarrow Q'$
  else
    $C_{t+1}^- \leftarrow C_t^-$ and $C_{t+1}^+ \leftarrow C_t$
  end if
  $t \leftarrow t + 1$
end while
Return: $\mathbf{x}^*$

Figure 7: Algorithm that efficiently implements the randomized ellipsoid algorithm of Bertsimas and Vempala. SETSEARCH performs a binary search for an ε-*IMAC* using the randomized INTERSECTSEARCH procedure to determine, with high probability, whether or not $\mathcal{X}_f^-$ contains any points less than a specified cost, C_t. Note that the ROUNDING algorithm discussed in Section 3.2.2 can be used as a preprocessing step so that $\mathcal{P}$ is near-isotropic and to obtain the samples for Q.

algorithm discussed below) and it provides a stopping condition for the overall intersection search algorithm.

The foundation of Bertsimas and Vempala's search algorithm is the capability to sample uniformly from an unknown but bounded convex body by means of the HIT-AND-RUN random walk technique (Algorithm 6) introduced by Smith (1996). Given an instance $\mathbf{x}^j \in \mathcal{P}^{s-1}$, HIT-AND-RUN selects a random direction $\mathbf{v}$ through $\mathbf{x}^j$ (we return to the selection of $\mathbf{v}$ in Section 3.2.2). Since $\mathcal{P}^{s-1}$ is a bounded convex set, the set $\Omega = \left\{\omega \geq 0 \;\middle|\; \mathbf{x}^j + \omega\mathbf{v} \in \mathcal{P}^{s-1}\right\}$ is a bounded interval indexing all feasible points along direction $\mathbf{v}$ through $\mathbf{x}^j$. Sampling ω uniformly from Ω (using rejection sampling) yields the next step of the random walk $\mathbf{x}^j + \omega\mathbf{v}$. As noted above, this random walk will not make progress if the interior of $\mathcal{P}^{s-1}$ is empty (which we preclude by assuming that $\mathcal{X}_f^-$ contains an r-ball), and efficient sampling also requires that $\mathcal{P}^{s-1}$ is sufficiently round. However, under the conditions discussed in Section 3.2.2, the HIT-AND-RUN random walk generates a sample uniformly from the convex body after $O^*\left(D^3\right)$ steps (Lovász and Vempala, 2004). We now detail the overall INTERSECTSEARCH procedure (Algorithm 5) and then discuss the mechanism used to maintain efficient sampling after each successive cut. It is worth noting that Algorithm 5 requires $\mathcal{P}^0$ to be in near-isotropic position and that Q is a set of samples from it; these requirements are met by using the ROUNDING algorithm of Lovász and Vempala discussed at the end of Section 3.2.2.

Randomized Ellipsoid Method: We use HIT-AND-RUN to obtain $2N$ samples $\left\{\mathbf{x}^j\right\}$ from $\mathcal{P}^{s-1} \subset \mathcal{X}_f^-$ for a single phase of the randomized ellipsoid method. If any sample $\mathbf{x}^j$ satisfies $A_I\left(\mathbf{x}^j - \mathbf{x}^A\right) \leq$

C^t, then $\mathbf{x}^j$ is in the intersection of $\mathcal{X}_f^-$ and $\mathcal{B}^{C^t}(A_1)$ and the procedure is complete. Otherwise, we want to significantly reduce the size of $\mathcal{P}^{s-1}$ without excluding any of $\mathcal{B}^{C^t}(A_1)$ so that sampling concentrates toward the intersection (if it exists)—for this we need a separating hyperplane for $\mathcal{B}^{C^t}(A_1)$. For any point $\mathbf{y} \notin \mathcal{B}^{C^t}(A_1)$, the (sub)gradient of the ℓ_1 cost is given by

$$h_d^{\mathbf{y}} = c_d \operatorname{sign}\left(y_d - x_d^A\right) \quad , \tag{6}$$

and is a separating hyperplane for $\mathbf{y}$ and $\mathcal{B}^{C^t}(A_1)$.

To achieve efficiency, we choose a point $\mathbf{z} \in \mathcal{P}^{s-1}$ so that cutting $\mathcal{P}^{s-1}$ through $\mathbf{z}$ with the hyperplane $\mathbf{h}^{\mathbf{z}}$ eliminates a significant fraction of $\mathcal{P}^{s-1}$. To do so, $\mathbf{z}$ must be centrally located within $\mathcal{P}^{s-1}$. We use the empirical centroid $\mathbf{z} = N^{-1}\sum_{\mathbf{x}\in\mathcal{R}}\mathbf{x}$ of the half of our samples in $\mathcal{R}$; the other half will be used in Section 3.2.2. We cut $\mathcal{P}^{s-1}$ with the hyperplane $\mathbf{h}^{\mathbf{z}}$ through $\mathbf{z}$; that is, $\mathcal{P}^s = \mathcal{P}^{s-1} \cap \mathcal{H}_{\mathbf{z}}$ where $\mathcal{H}_{\mathbf{z}}$ is the halfspace

$$\mathcal{H}_{\mathbf{z}} = \left\{ \mathbf{x} \;\middle|\; \mathbf{x}^\top \mathbf{h}^{\mathbf{z}} < \mathbf{z}^\top \mathbf{h}^{\mathbf{z}} \right\} \quad . \tag{7}$$

As shown by Bertsimas and Vempala, this cut achieves $vol(\mathcal{P}^s) \leq \frac{2}{3} vol\left(\mathcal{P}^{s-1}\right)$ with high probability if $N = O^*(D)$ and $\mathcal{P}^{s-1}$ is near-isotropic (see Section 3.2.2). Since the ratio of volumes between the initial circumscribing and inscribing balls of the feasible set is $(R/r)^D$, the algorithm can terminate after $T = O(D\log(R/r))$ unsuccessful iterations with a high probability that the intersection is empty.

Because every iteration in Algorithm 5 requires $N = O^*(D)$ samples, each of which need $K = O^*\left(D^3\right)$ random walk steps, and there are $T = O^*(D)$ iterations, the total number of membership queries required by Algorithm 5 is $O^*\left(D^5\right)$.

3.2.2 SAMPLING FROM A QUERYABLE CONVEX BODY

In the randomized ellipsoid method, random samples are used for two purposes: estimating the convex body's centroid and maintaining the conditions required for the HIT-AND-RUN sampler to efficiently generate points uniformly from a sequence of shrinking convex bodies. Until this point, we assumed the HIT-AND-RUN random walk efficiently produces uniformly random samples from any bounded convex body $\mathcal{P}$ accessible through membership queries. However, if the body is severely elongated, randomly selected directions will rarely align with the long axis of the body and our random walk will take small steps (relative to the long axis) and mix slowly. For the sampler to mix effectively, we need the convex body $\mathcal{P}$ to be sufficiently round, or more formally *near-isotropic*: for any unit vector $\mathbf{v}$, $\mathbb{E}_{\mathbf{x}\sim\mathcal{P}}\left[\left(\mathbf{v}^\top(\mathbf{x} - \mathbb{E}_{\mathbf{x}\sim\mathcal{P}}[\mathbf{x}])\right)^2\right]$ is bounded between $1/2$ and $3/2$ of $vol(\mathcal{P})$.

If the body is not near-isotropic, we must rescale $\mathcal{X}$ with an appropriate affine transformation $\mathbf{T}$ so the resulting transformed body $\mathcal{P}'$ is near-isotropic. With sufficiently many samples from $\mathcal{P}$ we can estimate $\mathbf{T}$ as their empirical covariance matrix. Instead, we rescale $\mathcal{X}$ implicitly using a technique described by Bertsimas and Vempala (2004). We maintain a set Q of sufficiently many uniform samples from the body $\mathcal{P}^s$, and in the HIT-AND-RUN algorithm (Algorithm 6), we sample the direction $\mathbf{v}$ based on this set. Intuitively, because the samples in Q are distributed uniformly in $\mathcal{P}^s$, the directions we sample based on the points in Q implicitly reflect the covariance structure of $\mathcal{P}^s$. This is equivalent to sampling the direction $\mathbf{v}$ from a normal distribution with zero mean and covariance of $\mathcal{P}$.

We must ensure Q is a set of sufficiently many samples from $\mathcal{P}^s$ after each cut taking $\mathcal{P}^s \leftarrow \mathcal{P}^{s-1} \cap \mathcal{H}_{\mathbf{z}^s}$. To do so, we initially resample $2N$ points from $\mathcal{P}^{s-1}$ using HIT-AND-RUN—half of

these, $\mathcal{R}$, are used to estimate the centroid $\mathbf{z}^s$ for the cut and the other half, $\mathcal{S}$, are used to repopulate $\mathcal{Q}$ after the cut. Because $\mathcal{S}$ contains independent uniform samples from $\mathcal{P}^{s-1}$, those in $\mathcal{P}^s$ after the cut constitute independent uniform samples from $\mathcal{P}^s$ (i.e., rejection sampling). By choosing N sufficiently large, our cut will be sufficiently deep and we will have sufficiently many points to resample $\mathcal{P}^s$ after the cut.

Finally, for this sampling approach to succeed, we need the initial set $\mathcal{P}^0$ to be transformed into near-isotropic position and we also need an initial set $\mathcal{Q}$ of uniform samples from the transformed $\mathcal{P}^0$ as input to Algorithm 5. However in our problem, we only have a single point $\mathbf{x}^- \in \mathcal{X}_f^-$ and our set, $\mathcal{P}^0$, need not be near-isotropic. Fortunately, there is an iterative procedure that uses the HIT-AND-RUN algorithm to simultaneously transform the initial convex set, $\mathcal{P}^0$, into a near-isotropic position and construct our initial set of samples, $\mathcal{Q}$. This algorithm, the ROUNDING algorithm as described by Lovász and Vempala (2003), uses $O^*\left(D^4\right)$ membership queries to find a transformation that places $\mathcal{P}^0$ into a near-isotropic position and produces an initial set of samples from it. We use this as a preprocessing step for Algorithms 5 and 7; that is, given $\mathcal{X}_f^-$ and $\mathbf{x}^- \in \mathcal{X}_f^-$, we construct $\mathcal{P}^0 = \mathcal{X}_f^- \cap \mathcal{B}^{2R}(A_1; \mathbf{x}^-)$ and then can use the ROUNDING algorithm to transform $\mathcal{P}^0$ and produce an initial uniform sample from it; that is, $\mathcal{Q} = \left\{\mathbf{x}^j \in \mathcal{P}^0\right\}$. These sets are then the inputs to our search algorithms.

3.2.3 OPTIMIZATION OVER ℓ_1 BALLS

We now revisit the outermost optimization loop (for searching the minimum feasible cost) of the algorithm and suggest improvements. These improvements are reflected in our final procedure SET-SEARCH in Algorithm 7—the total number of queries required is also $O^*\left(D^5\right)$. Again, Algorithm 7 requires $\mathcal{P}$ to be near-isotropic and that $\mathcal{Q}$ is a set of samples from it, which is accomplished by the ROUNDING algorithm discussed at the end of Section 3.2.2. First, notice that $\mathbf{x}^A$ and $\mathbf{x}^-$ are the same for every iteration of the optimization procedure. Further, in each iteration of Algorithm 7, the new set, $\mathcal{P}$, remains near-isotropic and the new $\mathcal{Q}$ is a set of samples from it since the sets returned by Algorithm 5 retain these properties. Thus, the set, $\mathcal{P}$, and the set of samples, $\mathcal{Q} = \left\{\mathbf{x}^j \in \mathcal{P}\right\}$, maintained by Algorithm 7 are sufficient to initialize INTERSECTSEARCH at each stage of its overall binary search over C^t, and we only need to execute the ROUNDING procedure once as a preprocessing step rather than re-invoking it before every invocation of INTERSECTSEARCH. Second, the separating hyperplane $\mathbf{h}^{\mathbf{y}}$ given by Equation (6) does not depend on the target cost C^t but only on $\mathbf{x}^A$, the common center of all the ℓ_1 balls used in this search. In fact, the separating hyperplane at point $\mathbf{y}$ is valid for all ℓ_1-balls of cost $C < A\left(\mathbf{y} - \mathbf{x}^A\right)$. Further, if $C < C^t$, we have $\mathcal{B}^C(A_1) \subset \mathcal{B}^{C^t}(A_1)$. Thus, the final state from a successful call to INTERSECTSEARCH for the C^t-ball can be used as the starting state for any subsequent call to INTERSECTSEARCH for all $C < C^t$. Hence, in Algorithm 7, we update $\mathcal{P}$ and $\mathcal{Q}$ only when Algorithm 5 succeeds.

4. Evasion for General ℓ_p Costs

Here we further extend ε-*IMAC* searchability over the family of convex-inducing classifiers to the full family of ℓ_p costs for any $0 < p \leq \infty$. As we demonstrate in this section, many ℓ_p costs are not generally ε-*IMAC* searchable for all $\varepsilon > 0$ over the family of convex-inducing classifiers (i.e., we show that finding an ε-*IMAC* for this family can require exponentially many queries in D and L_ε). In

fact, only the weighted ℓ_1 costs have known (randomized) polynomial query strategies when either the positive or negative set is convex.

4.1 Convex Positive Set

Here we explore the ability of the MULTILINESEARCH and K-STEP MULTILINESEARCH algorithms presented in Section 3.1 to find solutions to the near-optimal evasion problem for ℓ_p cost functions with $p \neq 1$. Particularly for $p > 1$ we will be exploring the consequences of using the MULTILINESEARCH algorithms using more search directions than just the $2 \cdot D$ axis-aligned directions. Figure 8 demonstrates how queries can be used to construct upper and lower bounds on general ℓ_p costs. The following lemma also summarizes well-known bounds on general ℓ_p costs using an ℓ_1 cost.

Lemma 7 *The largest ℓ_p ($p > 1$) ball enclosed within a C-cost ℓ_1 ball has a cost of $C \cdot D^{\frac{1-p}{p}}$ and for $p = \infty$ the cost is $C \cdot D^{-1}$.*

4.1.1 BOUNDING ℓ_p BALLS

In general, suppose we probe along some set of M unit directions and at some point we have at least one negative point supporting an upper bound of C_0^- and M positive points supporting a lower bound of C_0^+. The lower bound provided by those M positive points is the cost of the largest ℓ_p cost ball that fits entirely within their convex hull; let's say this cost is $C^\dagger \leq C_0^+$. In order to achieve ε-multiplicative optimality, we need $\frac{C_0^-}{C^\dagger} \leq 1+\varepsilon$, which we can rewrite as

$$\left(\frac{C_0^-}{C_0^+}\right)\left(\frac{C_0^+}{C^\dagger}\right) \leq 1+\varepsilon \ .$$

This allows us to break the problem into two parts. The first ratio C_0^-/C_0^+ is controlled solely by the accuracy ε achieved by running the multiline search algorithm for L_ε steps whereas the second ratio $C_0^+/C^\dagger$ depends only on how well the ℓ_p ball is approximated by the convex hull of the M search directions. These two ratios separate our task into choosing M and L_ε so that their product is less than $1+\varepsilon$. First we can choose parameters $\alpha \geq 0$ and $\beta \geq 0$ so that $(1+\alpha)(1+\beta) \leq 1+\varepsilon$. Then we choose M so that $\frac{C_0^+}{C^\dagger} = 1+\beta$ and use L_α steps so that multiline search with M directions will achieve $\frac{C_0^-}{C_0^+} = 1+\alpha$. In doing so, we create a generalized multiline search that can achieve ε-multiplicative optimality.

In the case of $p = 1$, we previously saw that choosing $M = 2 \cdot D$ allows us to exactly reconstruct the ℓ_1 ball so that $C_0^+/C^\dagger = 1$ (i.e., $\beta = 0$). Thus by taking $\alpha = \varepsilon$, we recover our original multiline search result.

We now address costs where $\beta > 0$. For a MULTILINESEARCH algorithm to be efficient, it is necessary that $\frac{C_0^+}{C^\dagger} = 1+\beta$ can be achieved with polynomially-many search directions (in D and L_ε) for some $\beta \leq \varepsilon$; otherwise, $(1+\alpha)(1+\beta) > 1+\varepsilon$ and the MULTILINESEARCH approach cannot succeed for any $\alpha > 0$. Thus, we quantify how many search directions (or queries) are required to achieve $\frac{C_0^+}{C^\dagger} \leq 1+\varepsilon$. Note that this ratio is independent of the relative size of these costs, so without loss of generality we will only consider bounds for unit-cost balls. Thus, we compute the largest value of $C^\dagger$ that can be achieved for the unit-cost ℓ_p ball (i.e., we make $C_0^+ = 1$) within the convex

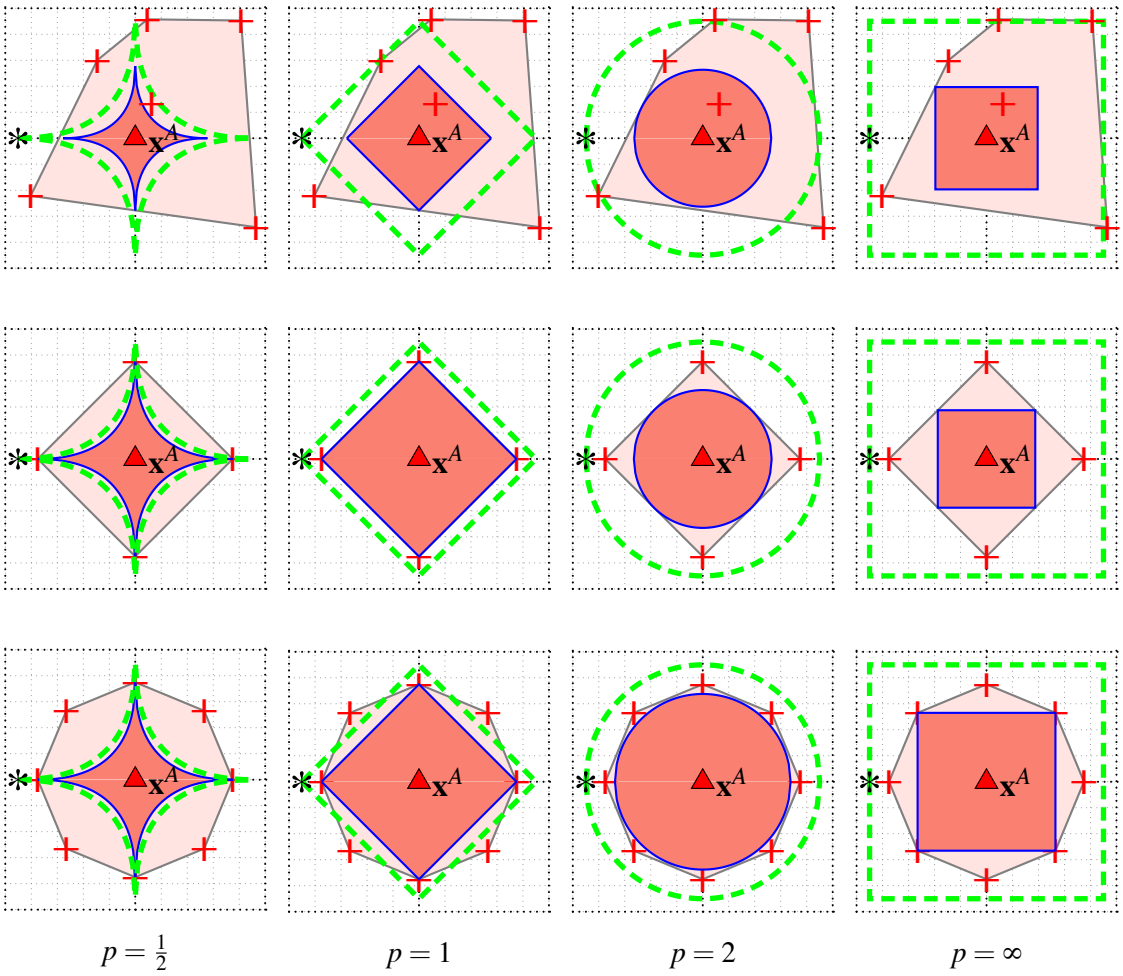


Figure 8: Convex hull for a set of queries and the resulting bounding balls for several ℓ_p costs. Each row represents a unique set of positive (red '+' points) and negative (black '$*$' points) queries and each column shows the implied upper bound (the green dashed ball) and lower bound (the solid blue ball) for a different ℓ_p cost. In the first row, the body is defined by a random set of seven queries, in the second, the queries are along the coordinate axes, and in the third, the queries are around a circle.

hull of M queries. In particular, we quantify how many queries are required to achieve

$$C^\dagger \geq \frac{1}{1+\varepsilon} \ .$$

We would like to show that only polynomially-many are required for at least some values of ε as this is sufficient for a MULTILINESEARCH approach to be efficient.

Lemma 8 *If there exists a configuration of M unit search directions with a convex hull that yields a bound $C^\dagger$ for the cost function A, then* MULTILINESEARCH *algorithms can use those search*

directions to achieve ε*-multiplicative optimality with a query complexity that is polynomial in* M *and* $L_\varepsilon^{(*)}$ *for any*

$$\varepsilon > \frac{1}{C^\dagger} - 1 \ .$$

Moreover, if the M *search directions yield* $C^\dagger = 1$ *for the cost function* A*, then* MULTILINESEARCH *algorithms can achieve* ε*-multiplicative optimality with a query complexity that is polynomial in* M *and* $L_\varepsilon^{(*)}$ *for any* $\varepsilon > 0$.

Notice that this lemma also reaffirms that for $p = 1$ using the $M = 2 \cdot D$ axis-aligned directions allows MULTILINESEARCH algorithms to achieve ε-multiplicative optimality for any $\varepsilon > 0$ with a query complexity that is polynomial in M and $L_\varepsilon^{(*)}$.

4.1.2 MULTILINE SEARCH FOR $0 < p < 1$

A simple result holds here. Namely, since the unit ℓ_1 ball bounds any unit ℓ_p balls with $0 < p < 1$, we can achieve $C_0^+ / C^\dagger = 1$ using only the $2 \cdot D$ axis-aligned search directions. Thus, for any $0 < p < 1$, we can efficiently search for any value of $\varepsilon > 0$. Whether or not any ℓ_p $(0 < p < 1)$ cost function can be efficiently optimized with fewer search directions is an open question.

4.1.3 MULTILINE SEARCH FOR $p > 1$

For this case, we can trivially use the ℓ_1 bound on ℓ_p balls as summarized by the following corollary.

Corollary 9 *For* $1 < p < \infty$ *and* $\varepsilon \in \left(D^{\frac{p-1}{p}} - 1, \infty\right)$ *any multi-line search algorithm can achieve* ε*-multiplicative optimality on* A_p *using* $M = 2 \cdot D$ *search directions. Similarly for* $\varepsilon \in (D-1, \infty)$ *any multi-line search algorithm can achieve* ε*-multiplicative optimality on* A_∞ *also using* $M = 2 \cdot D$ *directions*.

Proof From Lemma 7, the largest co-centered ℓ_p ball contained within the unit ℓ_1 ball has radius $D^{\frac{1-p}{p}}$ cost (or D for $p = \infty$). The bounds on ε then follow from Lemma 8. ■

Unfortunately, this result only applies for a range of ε that grows with D, which is insufficient for ε-*IMAC* searchability. In fact, for some fixed values of ε, there is no query-based strategy that can bound ℓ_p costs using polynomially-many queries in D as the following result shows.

Theorem 10 *For* $p > 1, D > 0$*, any initial bounds* $0 < C_0^+ < C_0^-$ *on the MAC, and* $\varepsilon \in \left(0, 2^{\frac{p-1}{p}} - 1\right)$ *(or* $\varepsilon \in (0,1)$ *for* $p = \infty$*), all algorithms must submit at least* $\alpha_{p,\varepsilon}^D$ *membership queries (for some constant* $\alpha_{p,\varepsilon} > 1$*) in the worst case to be* ε*-multiplicatively optimal on* $\mathcal{F}^{\text{convex},'+'}$ *for* ℓ_p *costs*.

The proof of this theorem and the definition of $\alpha_{p,\varepsilon}$ are provided in Appendix C. A consequence of this result is that there is no query-based algorithm that can efficiently find an ε-*IMAC* of any ℓ_p cost $(p > 1)$ for any *fixed* ε within the range $0 < \varepsilon < 2^{\frac{p-1}{p}} - 1$ (or $0 < \varepsilon < 1$ for $p = \infty$) on the family $\mathcal{F}^{\text{convex},'+'}$. However, from Theorem 9 and Lemma 8, multiline-search type algorithms efficiently find the ε-*IMAC* of any ℓ_p cost $(p > 1)$ for any $\varepsilon \in \left(D^{\frac{p-1}{p}} - 1, \infty\right)$ (or $D - 1 < \varepsilon < \infty$ for $p = \infty$). It is generally unclear if efficient algorithms exist for any values of ε between these intervals, but in the following section we derive a stronger bound for the case $p = 2$.

4.1.4 MULTILINE SEARCH FOR $p = 2$

Theorem 11 *For any $D > 1$, any initial bounds $0 < C_0^+ < C_0^-$ on the MAC, and $0 < \varepsilon < \frac{C_0^-}{C_0^+} - 1$, all algorithms must submit at least $\alpha_\varepsilon^{\frac{D-2}{2}}$ membership queries (where $\alpha_\varepsilon = \frac{(1+\varepsilon)^2}{(1+\varepsilon)^2-1} > 1$) in the worst case to be ε-multiplicatively optimal on $\mathcal{F}^{\text{convex},'+'}$ for ℓ_2 costs.*

The proof of this result is in Appendix D.

This result says that there is no algorithm that can generally achieve ε-multiplicative optimality for ℓ_2 costs for any *fixed* $\varepsilon > 0$ using only polynomially-many queries in D since the ratio $\frac{C_0^-}{C_0^+}$ could be arbitrarily large. It may appear that Theorem 11 contradicts Corollary 9. However, Corollary 9 only applies for an interval of ε that depends on D; that is, $\varepsilon > \sqrt{D} - 1$. Interestingly, substituting this lower bound on ε into the bound given by Theorem 11, we get that the number of required queries for $\varepsilon > \sqrt{D} - 1$ need only be

$$M \quad \geq \quad \left(\frac{(1+\varepsilon)^2}{(1+\varepsilon)^2 - 1}\right)^{\frac{D-2}{2}} \quad = \quad \left(\frac{D}{D-1}\right)^{\frac{D-2}{2}} \quad ,$$

which is a monotonically increasing function in D that asymptotes at $\sqrt{e} \approx 1.64$. Thus, Theorem 11 and Corollary 9 are in agreement since for $\varepsilon > \sqrt{D} - 1$, the former only requires that we need at least 2 queries.

4.2 Convex Negative Set

Algorithm 7 generalizes immediately to all weighted ℓ_p costs ($p \geq 1$) centered at $\mathbf{x}^A$ since they are convex. For these costs, an equivalent separating hyperplane for $\mathbf{y}$ can be used in place of Equation (6). They are given by the equivalent (sub)-gradients for ℓ_p cost balls:

$$\begin{aligned} h^{\mathbf{y}}_{p,d} &= c_d \cdot \text{sign}\left(y_d - x^A_d\right) \cdot \left(\frac{|y_d - x^A_d|}{A_p^{(\mathbf{c})}\left(\mathbf{y} - \mathbf{x}^A\right)}\right)^{p-1} \quad , \\ h^{\mathbf{y}}_{\infty,d} &= c_d \cdot \text{sign}\left(y_d - x^A_d\right) \cdot \mathbb{I}\left\{|y_d - x^A_d| = A_\infty^{(\mathbf{c})}\left(\mathbf{y} - \mathbf{x}^A\right)\right\} \quad . \end{aligned}$$

By only changing the cost function A and the separating hyperplane $\mathbf{h}^{\mathbf{y}}$ used for the halfspace cut in Algorithms 5 and 7, the randomized ellipsoid method can also be applied for any weighted ℓ_p cost $A_p^{(\mathbf{c})}$ with $p > 1$.

For more general convex costs A, we still have that every C-cost ball is a convex set (i.e., the sublevel set of a convex function is a convex set; see Boyd and Vandenberghe 2004, Chapter 3) and thus has a separating hyperplane. Further, since for any $D > C$, $\mathcal{B}^C(A) \subset \mathcal{B}^D(A)$, the separating hyperplane of the D-cost ball is also a separating hyperplane of the C cost ball and can be re-used in our Algorithm 7. Thus, this procedure is applicable for any convex cost function, A, so long as we can compute the separating hyperplanes of any cost ball of A for any point $\mathbf{y}$ not in the cost ball.

For non-convex costs A such as weighted ℓ_p costs with $0 < p < 1$, minimization over a convex set $\mathcal{X}_f^-$ is generally hard. However, there may be special cases when minimizing such a cost can be accomplished efficiently.

5. Conclusions and Future Work

In this paper, we study ε-*IMAC* searchability of convex-inducing classifiers. We present membership query algorithms that efficiently accomplish ε-*IMAC* search on this family. When the positive class is convex, we demonstrate efficient techniques that outperform the previous reverse-engineering approaches for linear classifiers in a continuous space. When the negative class is convex, we apply the randomized ellipsoid method introduced by Bertsimas and Vempala to achieve efficient ε-*IMAC* search. If the adversary is unaware of which set is convex, they can trivially run both searches to discover an ε-*IMAC* with a combined polynomial query complexity. We also show our algorithms can be efficiently extended for a number of special circumstances. Most importantly, we demonstrate that these algorithms can succeed without reverse engineering the classifier. Instead, these approaches systematically eliminate inconsistent hypotheses and progressively concentrate their efforts in an ever-shrinking neighborhood of a *MAC* instance. By doing so, these algorithms only require polynomially-many queries in spite of the size of the family of all convex-inducing classifiers.

We also consider the family of ℓ_p costs and show that $\mathcal{F}^{convex}$ is only generally ε-*IMAC* searchable for all $\varepsilon > 0$ when $p = 1$. For $0 < p < 1$, the MULTILINESEARCH algorithms of Section 3.1 achieve identical results when the positive set is convex, but the non-convexity of these ℓ_p costs precludes the use of the randomized ellipsoid method when the negative set is convex. The ellipsoid method does provide an efficient solution for convex negative sets when $p > 1$ (since these costs are convex). However, for convex positive sets, our results show that for $p > 1$ there is no algorithm that can efficiently find an ε-*IMAC* for all $\varepsilon > 0$. Moreover, for $p = 2$ we prove that there is no efficient algorithm for finding an ε-*IMAC* for any fixed value of ε.

By studying ε-*IMAC* searchability, we provide a broader picture of how machine learning techniques are vulnerable to query-based evasion attacks. Exploring near-optimal evasion is important for understanding how an adversary may circumvent learners in security-sensitive settings. In such an environment, system developers are hesitant to trust procedures that may create vulnerabilities. The algorithms we present are invaluable tools not for an adversary to develop better attacks but rather for analysts to better understand the vulnerabilities of their filters: our framework provides the query complexity in the worst-case setting when an adversary can directly query the classifier. However, our analysis and algorithms do not completely answer the evasion problem and also generally can not be easily used by an adversary since there are several real-world obstacles that are not incorporated into our framework. Queries may only be partially observable or noisy, and the feature set may only be partially known. Most importantly, an adversary may not be able to query all $\mathbf{x} \in \mathcal{X}$; instead their queries must be legitimate objects (such as email) that are mapped into $\mathcal{X}$. A real-world adversary must invert the feature-mapping—a generally difficult task. These limitations necessitate further research on the impact of partial observability and approximate querying on ε-*IMAC* search, and to design more secure filters. Broader open problems include: is ε-*IMAC* search possible on other classes of learners such as SVMs (linear in a large possibly infinite feature space)? Can an adversary efficiently perform ε-*IMAC* search when his cost is defined in an alternate feature space to the classifier's? Is ε-*IMAC* search feasible against an online learner that adapts as it is queried? Can learners be made resilient to these threats and how does this impact learning performance? These and other open problems for near-optimal evasion are discussed in Nelson et al. (2011).

Acknowledgments

We would like to thank Shing-hon Lau, Anthony Tran, Peter Bartlett, Marius Kloft, Peter Bodík, and the JMLR editor and reviewers for their helpful feedback on this project.

We gratefully acknowledge the support of our sponsors. This work was supported in part by TRUST (Team for Research in Ubiquitous Secure Technology), which receives support from the National Science Foundation (NSF award #CCF-0424422) and AFOSR (#FA9550-06-1-0244); RAD Lab, which receives support from California state MICRO grants (#06-148 and #07-012); DETER-lab (cyber-DEfense Technology Experimental Research laboratory), which receives support from DHS HSARPA (#022412) and AFOSR (#FA9550-07-1-0501); NSF award #DMS-0707060; the Siebel Scholars Foundation; and the following organizations: Amazon, BT, Cisco, DoCoMo USA Labs, EADS, ESCHER, Facebook, Google, HP, IBM, iCAST, Intel, Microsoft, NetApp, ORNL, Pirelli, Qualcomm, Sun, Symantec, TCS, Telecom Italia, United Technologies, and VMware. The opinions expressed in this paper are solely those of the authors and do not necessarily reflect the opinions of any funding agency, the State of California, or the U.S. government.

Appendix A. Query Complexity for K-STEP MULTILINESEARCH Algorithm

We consider the evasion problem as a game between *classifier* (playing first) and *adversary* (playing second) who wishes to evade detection by the classifier. To analyze the worst-case query complexity of K-STEP MULTILINESEARCH (Algorithm 3), we consider a *worst-case classifier* that seeks to maximize the number of queries submitted by the adversary. The worst-case classifier is completely aware of the state of the adversary; that is, the dimension of the space D, the adversary's goal L_ε, the cost function A, the bounds on the cost function C_t^+ and C_t^-, and so forth.

Proof of Theorem 4 At each iteration of Algorithm 3, the adversary chooses some direction, $\mathbf{e}$ not yet eliminated from $\mathcal{W}$. Every direction in $\mathcal{W}$ is feasible (i.e., could yield an ε-*IMAC*) and the worst-case classifier, by definition, will make this choice as costly as possible. During the K steps of binary search along this direction, regardless of which direction $\mathbf{e}$ is selected or how the worst-case classifier responds, the candidate multiplicative gap (see Section 2.2) along $\mathbf{e}$ will shrink by an exponent of 2^{-K}; that is,

$$\begin{aligned} \frac{B^-}{B^+} &= \left(\frac{C^-}{C^+}\right)^{2^{-K}}, \\ \log(G'_{t+1}) &= \log(G_t) \cdot 2^{-K}. \end{aligned}$$

The primary decision for the worst-case classifier occurs when the adversary begins querying other directions beside $\mathbf{e}$. At iteration t, the worst-case classifier has two options:

Case 1 ($t \in \mathcal{C}_1$): Respond with '+' for all remaining directions. Here the bound candidates B^+ and B^- are verified and thus the new gap is reduced by an exponent of 2^{-K}; however, no directions are eliminated from the search.

Case 2 ($t \in \mathcal{C}_2$): Choose at least one direction to respond with '−'. Here since only the value of C^- changes, the worst-case classifier can choose to respond to the first K queries so that the gap decreases by a negligible amount (by always responding with '+' during the first K queries along $\mathbf{e}$, the gap only decreases by an exponent of

$\left(1-2^{-K}\right)$). However, the worst-case classifier must choose some number $E_t \geq 1$ of directions that will be eliminated.

We conservatively assume that the gap only decreases for case 1, which decouples the analysis of the queries for $\mathcal{C}_1$ and $\mathcal{C}_2$ and allows us to upper bound the total number of queries. By this assumption, if $t \in \mathcal{C}_1$ we have $G_t = G_{t-1}^{2^{-K}}$ whereas if $t \in \mathcal{C}_2$ then $G_t = G_{t-1}$. By analyzing the gap before and after the final iteration T, it can be shown that

$$|\mathcal{C}_1| = \lceil L_\varepsilon / K \rceil \tag{8}$$

since, for the algorithm to terminate, there must be a total of at least L_ε binary search steps made during case one iterations and every case one iteration takes exactly K steps.

At every iteration in case one, the adversary makes exactly $K + |\mathcal{W}'_t| - 1$ queries where $\mathcal{W}'_t$ is the set of feasible directions remaining at the t^{th} iteration. While $\mathcal{W}'_t$ is controlled by the worst-case classifier, we can apply the bound $|\mathcal{W}'_t| \leq |\mathcal{W}|$. Using this and the relation from Equation (8), we can bound the number of queries, Q_1, used in case 1 by

$$\begin{aligned} Q_1 &\leq \sum_{t \in \mathcal{C}_1} (K + |\mathcal{W}| - 1) \\ &= \left\lceil \frac{L_\varepsilon}{K} \right\rceil \cdot (K + |\mathcal{W}| - 1) \\ &\leq \left(\frac{L_\varepsilon}{K} + 1 \right) \cdot K + \left\lceil \frac{L_\varepsilon}{K} \right\rceil \cdot (|\mathcal{W}| - 1) \\ &= L_\varepsilon + K + \left\lceil \frac{L_\varepsilon}{K} \right\rceil \cdot (|\mathcal{W}| - 1) \quad . \end{aligned}$$

For each case two iteration, we make exactly $K + E_t$ queries, and each eliminates $E_t \geq 1$ directions; hence, $|\mathcal{W}'_{t+1}| = |\mathcal{W}'_t| - E_t$. A worst-case classifier will always make $E_t = 1$ since that maximally limits how much the adversary gains. Nevertheless, since case 2 requires the elimination of at least 1 direction, we have $|\mathcal{C}_2| \leq |\mathcal{W}| - 1$ and moreover, regardless of the choice of E_t we have $\sum_{t \in \mathcal{C}_2} E_t \leq |\mathcal{W}| - 1$ since each direction can be eliminated no more than once and at least one direction must remain. Thus,

$$\begin{aligned} Q_2 &= \sum_{i \in \mathcal{C}_2} (K + E_t) \\ &\leq |\mathcal{C}_2| \cdot K + |\mathcal{W}| - 1 \\ &\leq (|\mathcal{W}| - 1)(K + 1) \quad . \end{aligned}$$

The total number of queries used by Algorithm 3 is then

$$\begin{aligned} Q = Q_1 + Q_2 &\leq L_\varepsilon + K + \left\lceil \frac{L_\varepsilon}{K} \right\rceil \cdot (|\mathcal{W}| - 1) + (|\mathcal{W}| - 1)(K + 1) \\ &= L_\varepsilon + \left\lceil \frac{L_\varepsilon}{K} \right\rceil \cdot |\mathcal{W}| + K \cdot |\mathcal{W}| + |\mathcal{W}| - \left\lceil \frac{L_\varepsilon}{K} \right\rceil - 1 \\ &\leq L_\varepsilon + \left(\left\lceil \frac{L_\varepsilon}{K} \right\rceil + K + 1 \right) |\mathcal{W}| \quad . \end{aligned}$$

Finally, choosing $K = \lceil\sqrt{L_\varepsilon}\rceil$ minimizes this expression. By substituting this K into Q's bound and using the bound $L_\varepsilon/\lceil\sqrt{L_\varepsilon}\rceil \leq \sqrt{L_\varepsilon}$, we have

$$Q \leq L_\varepsilon + \left(2\lceil\sqrt{L_\varepsilon}\rceil + 1\right)|\mathcal{W}| \ ,$$

establishing the result. ■

Appendix B. Proof of Lower Bound

Here we prove the lower bound from Section 3.1.2. Recall that D is the dimension of the space, $A : \Re^D \to \Re^+$ is any positive convex function, and $0 < C_0^+ < C_0^-$ are initial upper and lower bounds on the *MAC*. We also have that $\hat{\mathcal{F}}^{\text{convex,'+'}} \subset \mathcal{F}^{\text{convex,'+'}}$ is the set of classifiers consistent with the constraints on the *MAC*; that is, for $f \in \hat{\mathcal{F}}^{\text{convex,'+'}}$ we have $\mathcal{X}_f^+$ is convex, $\mathcal{B}^{C_0^+}(A) \subset \mathcal{X}_f^+$, and $\mathcal{B}^{C_0^-}(A) \not\subset \mathcal{X}_f^+$. As above, we consider a worst-case classifier.

Proof of Theorem 5 Suppose a query-based algorithm submits $N < D+1$ membership queries $\mathbf{x}^1, \ldots, \mathbf{x}^N \in \Re^D$ to the classifier. For the algorithm to be ε-optimal, these queries must constrain all consistent classifiers $\hat{\mathcal{F}}^{\text{convex,'+'}}$ to have a common point among their ε-*IMAC* sets. Suppose that the responses to the queries are consistent with the classifier f defined as:

$$f(\mathbf{x}) = \begin{cases} +1, & \text{if } A\left(\mathbf{x} - \mathbf{x}^A\right) < C_0^- \\ -1, & \text{otherwise} \end{cases} \ .$$

For this classifier, $\mathcal{X}_f^+$ is convex since A is a convex function, $\mathcal{B}^{C_0^+}(A) \subset \mathcal{X}_f^+$ since $C_0^+ < C_0^-$, and $\mathcal{B}^{C_0^-}(A) \not\subset \mathcal{X}_f^+$ since $\mathcal{X}_f^+$ is the open C_0^--ball whereas $\mathcal{B}^{C_0^-}(A)$ is the closed C_0^--ball. Moreover, since $\mathcal{X}_f^+$ is the open C_0^--ball, $\nexists\, \mathbf{x} \in \mathcal{X}_f^-$ s.t. $A\left(\mathbf{x} - \mathbf{x}^A\right) < C_0^-$ therefore $MAC(f, A) = C_0^-$, and any ε-optimal points $\mathbf{x}' \in \varepsilon\text{-}IMAC^{(*)}(f, A)$ must satisfy $C_0^- \leq A\left(\mathbf{x}' - \mathbf{x}^A\right) \leq (1+\varepsilon)C_0^-$.

Consider an alternative classifier g that responds identically to f for $\mathbf{x}^1, \ldots, \mathbf{x}^N$ but has a different convex positive set $\mathcal{X}_g^+$. Without loss of generality, suppose the first $M \leq N$ queries are positive and the remainder are negative. Let $\mathcal{G} = conv\left(\mathbf{x}^1, \ldots, \mathbf{x}^M\right)$; that is, the convex hull of the M positive queries. Now let $\mathcal{X}_g^+$ be the convex hull of $\mathcal{G}$ and the C_0^+-ball of A: $\mathcal{X}_g^+ = conv\left(\mathcal{G} \cup \mathcal{B}^{C_0^+}(A)\right)$. Since $\mathcal{G}$ contains all positive queries and $C_0^+ < C_0^-$, the convex set $\mathcal{X}_g^+$ is consistent with the observed responses, $\mathcal{B}^{C_0^+}(A) \subset \mathcal{X}_g^+$ by definition, and $\mathcal{B}^{C_0^-}(A) \not\subset \mathcal{X}_g^+$ since the positive queries are all inside the open C_0^--sublevel set. Further, since $M \leq N < D+1$, $\mathcal{G}$ is contained in a proper linear subspace of $\Re^D$ and hence the interior of $\mathcal{G}$ is empty; that is, $int(\mathcal{G}) = \emptyset$. Hence, there is always some point from $\mathcal{B}^{C_0^+}(A)$ that is on the boundary of $\mathcal{X}_g^+$; that is, $\mathcal{B}^{C_0^+}(A) \not\subset int(\mathcal{G})$ because $int(\mathcal{G}) = \emptyset$ and $\mathcal{B}^{C_0^+}(A) \neq \emptyset$. Hence, there must be at least one point from $\mathcal{B}^{C_0^+}(A)$ on the boundary of the convex hull of $\mathcal{B}^{C_0^+}(A)$ and $\mathcal{G}$. Hence, $MAC(g, A) = \inf_{\mathbf{x} \in \mathcal{X}_g^-}\left[A\left(\mathbf{x} - \mathbf{x}^A\right)\right] = C_0^+$. Since the accuracy $\varepsilon < \frac{C_0^-}{C_0^+} - 1$, any $\mathbf{x} \in \varepsilon\text{-}IMAC^{(*)}(g, A)$ must have

$$A\left(\mathbf{x} - \mathbf{x}^A\right) \leq (1+\varepsilon)C_0^+ < \frac{C_0^-}{C_0^+}C_0^+ = C_0^- \ ,$$

whereas any $\mathbf{y} \in \varepsilon\text{-}IMAC^{(*)}(f,A)$ must have $A\left(\mathbf{y}-\mathbf{x}^A\right) \geq C_0^-$. Thus, $\varepsilon\text{-}IMAC^{(*)}(f,A) \cap \varepsilon\text{-}IMAC^{(*)}(g,A) = \emptyset$ and we have constructed two convex-inducing classifiers f and g both consistent with the query responses with no common ε-$IMAC^{(*)}$.

Suppose instead that a query-based algorithm submits $N < L_\varepsilon^{(*)}$ membership queries. Recall our definitions: C_0^- is the initial upper bound on the MAC, C_0^+ is the initial lower bound on the MAC, and $G_t^{(*)} = C_t^-/C_t^+$ is the gap between the upper bound and lower bound at iteration t. Here, the worst-case classifier f responds with

$$f\left(\mathbf{x}^t\right) = \begin{cases} +1\,, & \text{if } A\left(\mathbf{x}^t - \mathbf{x}^A\right) \leq \sqrt{C_{t-1}^- \cdot C_{t-1}^+} \\ -1\,, & \text{otherwise} \end{cases} .$$

When the classifier responds with '+', C_t^+ increases to no more than $\sqrt{C_{t-1}^- \cdot C_{t-1}^+}$ and so $G_t \geq \sqrt{G_{t-1}}$. Similarly when this classifier responds with '$-$', C_t^- decreases to no less than $\sqrt{C_{t-1}^- \cdot C_{t-1}^+}$ and so again $G_t \geq \sqrt{G_{t-1}}$. These responses ensure the invariant $G_t \geq \sqrt{G_{t-1}}$ and since the algorithm can not terminate until $G_N \leq 1+\varepsilon$, we have $N \geq L_\varepsilon^{(*)}$ from Equation (4). Otherwise, there are still two convex-inducing classifiers with consistent query responses but with no common ε-$IMAC$. The first classifier's positive set is the smallest cost-ball enclosing all positive queries, while the second classifier's positive set is the largest cost-ball enclosing all positive queries but no negatives. The MAC values for these classifiers differ by more than a factor of $(1+\varepsilon)$ if $N < L_\varepsilon^{(*)}$, so they have no common ε-$IMAC$. ■

Appendix C. Proof of Theorem 10

First we introduce the following lemma for the D-dimensional *hypercube graphs*—a collection of 2^D nodes of the form $(\pm 1, \pm 1, \ldots, \pm 1)$ where each node has an edge to every other node that is Hamming distance 1 from it.

Lemma 12 *For any $0 < \delta \leq 1/2$ and $D \geq 1$, to cover a D-dimensional hypercube graph so that every vertex has a Hamming distance of at most $\lfloor \delta D \rfloor$ to some vertex in the covering, the number of vertices in the covering must be*

$$Q(D,h) \geq 2^{D(1-H(\delta))} ,$$

where $H(\delta) = -\delta \log_2 \delta - (1-\delta)\log_2(1-\delta)$ is the entropy *of* δ.

Proof There are 2^D vertices in the D-dimensional hypercube graph. Each vertex in the covering is within a Hamming distance of at most h for exactly $\sum_{k=0}^{h} \binom{D}{k}$ vertices. Thus, one needs at least $2^D / \left(\sum_{k=0}^{h} \binom{D}{k}\right)$ to cover the hypercube graph. Now we apply the following bound (see Flum and Grohe, 2006, Page 427)

$$\sum_{k=0}^{\lfloor \delta D \rfloor} \binom{D}{k} \leq 2^{H(\delta)D}$$

to the denominator,[6] which is valid for any $0 < \delta \leq 1/2$. ■

6. Gottlieb et al. (2011) present a better entropy bound on this sum of binomial coefficients, but it is unnecessary for our result.

Lemma 13 *The minimum of the ℓ_p cost function A_p from the target $\mathbf{x}^A$ to the halfspace $\mathcal{H}_{\mathbf{w},\mathbf{b}} = \left\{\mathbf{x} \;\middle|\; \mathbf{x}^\top\mathbf{w} \geq \mathbf{b}^\top\mathbf{w}\right\}$ can be expressed in terms of the equivalent hyperplane $\mathbf{x}^\top\mathbf{w} \geq d$ parameterized by a normal vector $\mathbf{w}$ and displacement $d = \left(\mathbf{b}-\mathbf{x}^A\right)^\top\mathbf{w}$ as*

$$\min_{\mathbf{x}\in\mathcal{H}_{\mathbf{w},d}} A_p\left(\mathbf{x}-\mathbf{x}^A\right) = \begin{cases} d\cdot\|\mathbf{w}\|_{\frac{p}{p-1}}^{-1}, & \text{if } d>0 \\ 0, & \text{otherwise} \end{cases} \tag{9}$$

for all $1<p<\infty$ and for $p=\infty$ it is

$$\min_{\mathbf{x}\in\mathcal{H}_{\mathbf{w},d}} A_\infty\left(\mathbf{x}-\mathbf{x}^A\right) = \begin{cases} d\cdot\|\mathbf{w}\|_1^{-1}, & \text{if } d>0 \\ 0, & \text{otherwise} \end{cases} . \tag{10}$$

Proof For $1<p<\infty$, minimizing A_p on the halfspace $\mathcal{H}_{\mathbf{w},\mathbf{b}}$ is equivalent to finding a minimizer for

$$\min_{\mathbf{x}} \frac{1}{p}\sum_{i=1}^{D}|x_i|^p \quad \text{s.t.} \quad \mathbf{x}^\top\mathbf{w} \leq d \ .$$

Clearly, if $d\leq 0$ then the vector $\mathbf{0}$ (corresponding to $\mathbf{x}^A$ in the transformed space) trivially satisfies the constraint and minimizes the cost function with cost 0 which yields the second case of Equation (9). For the case $d>0$, we construct the Lagrangian

$$\mathcal{L}(\mathbf{x},\lambda) \triangleq \frac{1}{p}\sum_{i=1}^{D}|x_i|^p - \lambda\left(\mathbf{x}^\top\mathbf{w}-d\right) \ .$$

Differentiating this with respect to $\mathbf{x}$ and setting that partial derivative equal to zero yields $x_i^* = \text{sign}(w_i)\left(\lambda|w_i|\right)^{\frac{1}{p-1}}$. Plugging this back into the Lagrangian yields

$$\mathcal{L}(\mathbf{x}^*,\lambda) = \frac{1-p}{p}\lambda^{\frac{p}{p-1}}\sum_{i=1}^{D}|w_i|^{\frac{p}{p-1}} + \lambda d \ ,$$

which we now differentiate with respect to λ and set the derivative equal to zero to yield $\lambda^* = \left(\frac{d}{\sum_{i=1}^{D}|w_i|^{\frac{p}{p-1}}}\right)^{p-1}$. Plugging this solution into the formula for $\mathbf{x}^*$ yields the solution $x_i^* = \text{sign}(w_i)\left(\frac{d}{\sum_{i=1}^{D}|w_i|^{\frac{p}{p-1}}}\right)|w_i|^{\frac{1}{p-1}}$. The ℓ_p cost of this optimal solution is given by $A_p\left(\mathbf{x}^*-\mathbf{x}^A\right) = d\cdot\|\mathbf{w}\|_{\frac{p}{p-1}}^{-1}$, which is the first case of Equation (9).

For $p=\infty$, once again if $d\leq 0$ then the vector $\mathbf{0}$ trivially satisfies the constraint and minimizes the cost function with cost 0 which yields the second case of Equation (10). For the case $d>0$, we use the geometry of hypercubes (the equi-cost balls of a ℓ_∞ cost function) to derive the second case of Equation (10). Any optimal solution must occur at a point where the hyperplane given by $\mathbf{x}^\top\mathbf{w} = \mathbf{b}^\top\mathbf{w}$ is tangent to a hypercube about $\mathbf{x}^A$—this can either occur along a side (face) of the hypercube or at a corner. However, if the plane is tangent along a side (face) it is also tangent at a corner of the hypercube. Hence, there is always an optimal solution at some corner of the optimal cost hypercube.

At a corner of the hypercube, we have the following property:

$$|x_1^*| = |x_2^*| = \ldots = |x_D^*| \ ;$$

that is, the magnitude of all coordinates of this optimal solution is the same value. Further, the sign of the optimal solution's i^{th} coordinate must agree with the sign of the hyperplane's i^{th} coordinate, w_i. These constraints, along with the hyperplane constraint, lead to the following formula for an optimal solution: $x_i = d \cdot \text{sign}(w_i)\|\mathbf{w}\|_1^{-1}$ for all i. The ℓ_∞ cost of this solution is simply $d \cdot \|\mathbf{w}\|_1^{-1}$. ■

Finally, for the proof of Theorem 10, we use the orthants (centered at $\mathbf{x}^A$)—an *orthant* is the D-dimensional generalization of a quadrant in 2-dimensions. There are 2^D orthants in a D-dimensional space. We represent each orthant by its *canonical representation* which is a vector of D positive or negative ones; that is, the orthant represented by $\mathbf{a} = (\pm 1, \pm 1, \ldots, \pm 1)$ contains the point $\mathbf{x}^A + \mathbf{a}$ and is the set of all points $\mathbf{x}$ satisfying:

$$x_i \in \begin{cases} [0, +\infty], & \text{if } a_i = +1 \\ [-\infty, 0], & \text{if } a_i = -1 \end{cases} .$$

Proof of Theorem 10 Suppose a query-based algorithm submits N membership queries $\mathbf{x}^1, \ldots, \mathbf{x}^N \in \Re^D$ to the classifier. Again, for the algorithm to be ε-optimal, these queries must constrain all consistent classifiers $\hat{\mathcal{F}}^{\text{convex},'+'}$ to have a common point among their ε-*IMAC* sets. The responses described above are consistent with the classifier f defined as

$$f(\mathbf{x}) = \begin{cases} +1, & \text{if } A_p\left(\mathbf{x} - \mathbf{x}^A\right) < C_0^- \\ -1, & \text{otherwise} \end{cases} .$$

For this classifier, $\mathcal{X}_f^+$ is convex since A_p is a convex function for $p \geq 1$, $\mathcal{B}^{C_0^+}(A_p) \subset \mathcal{X}_f^+$ since $C_0^+ < C_0^-$, and $\mathcal{B}^{C_0^-}(A_p) \not\subset \mathcal{X}_f^+$ since $\mathcal{X}_f^+$ is the open C_0^--ball whereas $\mathcal{B}^{C_0^-}(A_p)$ is the closed C_0^--ball. Moreover, since $\mathcal{X}_f^+$ is the open C_0^--ball, $\nexists\, \mathbf{x} \in \mathcal{X}_f^-$ s.t. $A_p\left(\mathbf{x} - \mathbf{x}^A\right) < C_0^-$ therefore $MAC(f, A_p) = C_0^-$, and any ε-optimal points $\mathbf{x}' \in \varepsilon\text{-}IMAC^{(*)}(f, A_p)$ must satisfy $C_0^- \leq A_p\left(\mathbf{x}' - \mathbf{x}^A\right) \leq (1+\varepsilon)C_0^-$.

Now consider an alternative classifier g that responds identically to f for $\mathbf{x}^1, \ldots, \mathbf{x}^N$ but has a different convex positive set $\mathcal{X}_g^+$. Without loss of generality suppose the first $M \leq N$ queries are positive and the remaining are negative. Here we consider a set which is a convex hull of the orthants of all M positive queries; that is,

$$\mathcal{G} = conv\left(orth\left(\mathbf{x}^1\right) \cap \mathcal{X}_f^+, orth\left(\mathbf{x}^2\right) \cap \mathcal{X}_f^+, \ldots, orth\left(\mathbf{x}^M\right) \cap \mathcal{X}_f^+\right)$$

where $orth(\mathbf{x})$ is some orthant that $\mathbf{x}$ lies within relative to the center, $\mathbf{x}^A$ (a data point may lie within more than one orthant but, to cover it, we need only have one orthant that contains it). By intersecting each data point's orthant with the set $\mathcal{X}_f^+$ and taking the convex hull of these regions, $\mathcal{G}$ is convex , contains $\mathbf{x}^A$ and is a subset of $\mathcal{X}_f^+$ consistent with all the query responses of f; that is, each of the M positive queries are in $\mathcal{X}_g^+$ and all the negative queries are in $\mathcal{X}_g^-$. Moreover, $\mathcal{G}$ contains the convex hull of the M positive queries. Thus, by finding the largest enclosed ℓ_p ball within the $\mathcal{G}$, we upper bound $MAC(g, A_p)$.

We now represent each orthant as a vertex in a D-dimensional hypercube graph—the Hamming distance between any pair of orthants is the number of different coordinates in their canonical representations and two orthants are adjacent in the graph if and only if they have Hamming distance of one. Using this notion of Hamming distance, we will seek a K-covering of the hypercube. We refer to the orthants used in $\mathcal{G}$ to cover the M positive queries as *covering orthants* and their corresponding vertices form a covering of the hypercube. Suppose the M covering orthants are sufficient for a K covering but not a $K-1$ covering; then there must be at least one vertex not in the covering that has at least a K Hamming distance to every vertex in the covering. This vertex corresponds to an empty orthant that differs from all covered orthants in at least K coordinates of their canonical vertices. Without loss of generality, suppose this uncovered orthant has the canonical vertex of all positive ones which we scale to $C_0^-(+1,+1,\ldots,+1)$. Consider the hyperplane with normal vector $\mathbf{w} = (+1,+1,\ldots,+1)$ and displacement

$$d \quad = \quad \begin{cases} C_0^-(D-K)^{\frac{p-1}{p}} & \text{if } 1 < p < \infty \\ C_0^-(D-K) & \text{if } p = \infty \end{cases}$$

that specifies the function $s(\mathbf{x}) = \mathbf{x}^\top \mathbf{w} - d = \sum_{i=1}^{D} \mathbf{x}_i - d$. For this hyperplane, the vertex $C_0^-(+1,+1,\ldots,+1)$ yields $s\left(C_0^-(+1,+1,\ldots,+1)\right) = C_0^- D - d > 0$. Also for any orthant $\mathbf{a}$ with Hamming distance at least K from this uncovered orthant, we have that for any $\mathbf{x} \in orth(\mathbf{a}) \cap \mathcal{X}_f^+$, by definition of the orthant and $\mathcal{X}_f^+$, the function s yields

$$s(\mathbf{x}) \quad = \quad \sum_{\{i \mid a_i = +1\}} \underbrace{x_i}_{\geq 0} + \sum_{\{i \mid a_i = -1\}} \underbrace{x_i}_{\leq 0} - d \ .$$

Since all the terms in the second summation are non-positive, the second sum is at most 0. Thus, by maximizing the first summation, we upper bound $s(\mathbf{x})$. The summation $\sum_{\{i \mid a_i = +1\}} x_i$ (with the constraint that $\|\mathbf{x}\|_p < C_0^-$) has at most $D-K$ terms and is maximized by $x_i = C_0^-(D-K)^{-1/p}$ (or $x_i = C_0^-$ for $p = \infty$) for which the first summation is upper bounded by $C_0^-(D-K)^{\frac{p-1}{p}}$ or $C_0^-(D-K)$ for $p = \infty$; that is, it is upper bounded by d. Thus, we have that $s(\mathbf{x}) \leq 0$, and this hyperplane separates the scaled vertex $C_0^-(+1,+1,\ldots,+1)$ from each set $orth(\mathbf{a}) \cap \mathcal{X}_f^+$ where $\mathbf{a}$ is the canonical representation of any orthant with a Hamming distance of at least K. This hyperplane also separates the scaled vertex from $\mathcal{G}$ by the properties of the convex hull. Since the displacement d defined above is greater than 0, by applying Lemma 13, this separating hyperplane upper bounds the cost of the largest ℓ_p ball enclosed in $\mathcal{G}$ as

$$MAC(g, A_p) \leq C_0^-(D-K)^{\frac{p-1}{p}} \cdot \|\mathbf{w}\|_{\frac{p}{p-1}}^{-1} = C_0^- \left(\frac{D-K}{D}\right)^{\frac{p-1}{p}}$$

for $1 < p < \infty$ and

$$MAC(g, A_p) \leq C_0^-(D-K) \cdot \|\mathbf{1}\|_1^{-1} = C_0^- \frac{D-K}{D}$$

for $p = \infty$. Since we have an upper bound on the *MAC* of g and the *MAC* of f is C_0^-, in order to have a common ε-*IMAC* between these classifiers, we must have

$$(1+\varepsilon) \geq \begin{cases} \left(\frac{D}{D-K}\right)^{\frac{p-1}{p}}, & \text{if } 1 < p < \infty \\ \frac{D}{D-K}, & \text{if } p = \infty \end{cases} \ .$$

Solving for the value of K required to achieve a desired accuracy of $1+\varepsilon$ we have

$$K \leq \begin{cases} \frac{(1+\varepsilon)^{\frac{p}{p-1}}-1}{(1+\varepsilon)^{\frac{p}{p-1}}} D, & \text{if } 1 < p < \infty \\ \frac{\varepsilon}{1+\varepsilon} D, & \text{if } p = \infty \end{cases},$$

which bounds the size of the covering required to achieve the desired accuracy.

For the case $1 < p < \infty$, by Lemma 12, there must be

$$M \geq \exp\left\{ \ln(2) \cdot D \left(1 - H\left(\frac{(1+\varepsilon)^{\frac{p}{p-1}} - 1}{(1+\varepsilon)^{\frac{p}{p-1}}} \right) \right) \right\}$$

vertices of the hypercube in the covering to achieve any accuracy $0 < \varepsilon < 2^{\frac{p-1}{p}} - 1$, for which $\delta = \frac{(1+\varepsilon)^{\frac{p}{p-1}}-1}{(1+\varepsilon)^{\frac{p}{p-1}}} < \frac{1}{2}$ as required by the lemma. Moreover, since $H(\delta) < 1$ for $\delta < \frac{1}{2}$,

$$\alpha_{p,\varepsilon} \quad = \quad \exp\left\{ \ln(2) \left(1 - H\left(\frac{(1+\varepsilon)^{\frac{p}{p-1}} - 1}{(1+\varepsilon)^{\frac{p}{p-1}}} \right) \right) \right\} > 1$$

and we have $M \geq \alpha_{p,\varepsilon}^D$.

Similarly for $p = \infty$, Lemma 12 can be applied yielding $M \geq 2^{D\left(1-H\left(\frac{\varepsilon}{1+\varepsilon}\right)\right)}$ to achieve any desired accuracy $0 < \varepsilon < 1$ (for which $\varepsilon/(1+\varepsilon) < 1/2$ as required by the Lemma). Again, by the properties of entropy, the constant $\alpha_{\infty,\varepsilon} = 2^{\left(1-H\left(\frac{\varepsilon}{1+\varepsilon}\right)\right)} > 1$ for $0 < \varepsilon < 1$ and we have $M \geq \alpha_{\infty,\varepsilon}^D$. ■

Appendix D. Proof of Theorem 11

For this proof, we build on previous results for covering hyperspheres. The proof is based on the following covering number result by Wyner and Shannon, which bounds the minimum number of spherical caps required to cover a hypersphere. A D-dimensional *spherical cap* is the outward region formed by the intersection of a hypersphere and a halfspace as depicted in Figure 9. We parameterize the caps by the hypersphere's radius R and the half-angle ϕ about a central radius (through the caps's peak) as in the right-most diagram of Figure 9.

We now derive a bound on the number of spherical caps of half-angle ϕ required to cover the sphere, mirroring the result of Wyner (1965).

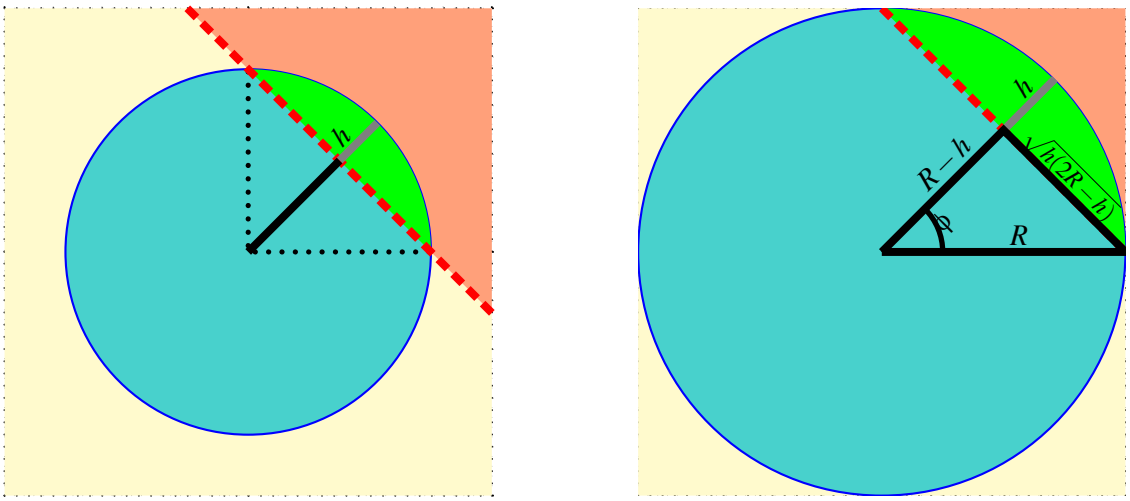
Lemma 14 ***(Result based on Wyner 1965)*** *Covering the surface of D-dimensional hypersphere of radius R requires at least*

$$\left(\frac{1}{\sin\phi} \right)^{D-2}$$

spherical caps of half-angle $\phi \in (0, \frac{\pi}{2})$.

Proof In *Capabilities of Bounded Discrepancy Decoding*, Wyner showed that the minimal number, M, of spherical caps of half-angle ϕ required to cover D-dimensional hypersphere of radius R is given by

$$M \geq \frac{D\sqrt{\pi}\Gamma\left(\frac{D+1}{2}\right)}{(D-1)\Gamma\left(1+\frac{D}{2}\right)} \left[\int_0^{\phi} \sin^{D-2}(t)dt \right]^{-1} .$$



(a) (b)

Figure 9: This figure depicts the geometry of spherical caps. (a) A spherical cap of height h, which is created by a plane passing through the sphere. The green region represents the area of the cap. (b) The geometry of the spherical cap; the intersecting halfspace forms a right triangle with the centroid of the hypersphere. The length of the side of this triangle adjacent to the centroid is $R-h$, its hypotenuse has length R, and the side opposite the centroid has length $\sqrt{h(2R-h)}$. The half angle ϕ, given by $\sin(\phi) = \frac{\sqrt{h(2R-h)}}{R}$, of the right circular cone is used to parameterize the cap.

where $\Gamma(x)$ is the usual gamma function. This result follows directly from computing the surface area of the hypersphere and that of each spherical cap.

We continue by lower bounding the above integral for a looser but more interpretable bound. Integrals of the form $\int_0^\phi \sin^D(t)dt$ also arise in computing the volume of a spherical cap. This volume (and thus the integral) can be bounded by enclosing the cap within a hypersphere; compare with Ball (1997). This yields the following bound:

$$\int_0^\phi \sin^D(t)dt \leq \frac{\sqrt{\pi}\Gamma\left(\frac{D+1}{2}\right)}{\Gamma\left(1+\frac{D}{2}\right)} \cdot \sin^D\phi \ .$$

Using this bound on the integral, our bound on the size of the covering becomes

$$M \geq \frac{D\sqrt{\pi}\Gamma\left(\frac{D+1}{2}\right)}{(D-1)\Gamma\left(1+\frac{D}{2}\right)} \left[\frac{\sqrt{\pi}\Gamma\left(\frac{D-1}{2}\right)}{\Gamma\left(\frac{D}{2}\right)} \cdot \sin^{D-2}\phi\right]^{-1} \ .$$

Now using properties of the gamma function, it can be shown that $\frac{\Gamma\left(\frac{D+1}{2}\right)\Gamma\left(\frac{D}{2}\right)}{\Gamma\left(1+\frac{D}{2}\right)\Gamma\left(\frac{D-1}{2}\right)} = \frac{D-1}{D}$ so that after canceling terms we arrive at our result:

$$M \geq \left(\frac{1}{\sin\phi}\right)^{D-2} \ .$$

■

Proof of Theorem 11 Suppose a query-based algorithm submits N membership queries $\mathbf{x}^1,\ldots,\mathbf{x}^N \in \Re^D$ to the classifier. For the algorithm to be ε-optimal, these queries must constrain all consistent classifiers, $\hat{\mathcal{F}}^{\text{convex,'+'}}$, to have a common point among their ε-*IMAC* sets. Suppose that all the responses are consistent with the classifier f defined as

$$f(\mathbf{x}) = \begin{cases} +1, & \text{if } A_2\left(\mathbf{x}-\mathbf{x}^A\right) < C_0^- \\ -1, & \text{otherwise} \end{cases} .$$

For this classifier, $\mathcal{X}_f^+$ is convex since A_2 is a convex function, $\mathcal{B}^{C_0^+}(A_2) \subset \mathcal{X}_f^+$ since $C_0^+ < C_0^-$, and $\mathcal{B}^{C_0^-}(A_2) \not\subset \mathcal{X}_f^+$ since $\mathcal{X}_f^+$ is the open C_0^--ball whereas $\mathcal{B}^{C_0^-}(A_2)$ is the closed C_0^--ball. Moreover, since $\mathcal{X}_f^+$ is the open C_0^--ball, $\nexists\, \mathbf{x} \in \mathcal{X}_f^-$ such that $A_2\left(\mathbf{x}-\mathbf{x}^A\right) < C_0^-$. Therefore, $MAC(f,A_2) = C_0^-$, and any ε-optimal points $\mathbf{x}' \in \varepsilon\text{-}IMAC^{(*)}(f,A_2)$ must satisfy $C_0^- \le A_2\left(\mathbf{x}'-\mathbf{x}^A\right) \le (1+\varepsilon)C_0^-$.

Now consider an alternative classifier g that responds identically to f for $\mathbf{x}^1,\ldots,\mathbf{x}^N$ but has a different convex positive set $\mathcal{X}_g^+$. Without loss of generality, suppose the first $M \le N$ queries are positive and the remaining are negative. Let $\mathcal{G} = conv\left(\mathbf{x}^1,\ldots,\mathbf{x}^M\right)$ be the convex hull of these M positive queries. We will assume $\mathbf{x}^A \in \mathcal{G}$, since otherwise, we construct the set $\mathcal{X}_g^+$ as in the proof for Theorem 5 above and achieve $MAC(f,A_2) = C_0^+$ thereby achieving our desired result. Now consider the projection of each of the positive queries onto the surface of the ℓ_2 ball $\mathcal{B}^{C_0^-}(A_2)$, given by the points $\mathbf{z}^i = C_0^- \frac{\mathbf{x}^i}{A_2(\mathbf{x}^i-\mathbf{x}^A)}$. Since each positive query lies along the line between $\mathbf{x}^A$ and its projection $\mathbf{z}^i$, by convexity and the fact that $\mathbf{x}^A \in \mathcal{G}$, we have $\mathcal{G} \subset conv\left(\mathbf{z}^1,\mathbf{z}^2,\ldots,\mathbf{z}^M\right)$—we will call this enlarged hull $\hat{\mathcal{G}}$. These M projected points $\left\{\mathbf{z}^i\right\}$ must form a covering of the C_0^--hypersphere as the locii of caps of half-angle $\phi_\varepsilon^* = \arccos\left((1+\varepsilon)^{-1}\right)$. If not, then there exists some point on the surface of this hypersphere that is at least an angle ϕ_ε^* from all $\mathbf{z}^i$ points and the resulting ϕ_ε^*-cap centered at this uncovered point is not in $\hat{\mathcal{G}}$ (since a cap is defined as the intersection of the hypersphere and a halfspace). Moreover, by definition of the ϕ_ε^*-cap, it achieves a minimal ℓ_2 cost of $C_0^- \cos\phi_\varepsilon^*$. Thus, if we fail to achieve a ϕ_ε^*-covering of the C_0^--hypersphere, the alternative classifier g has $MAC(g,A_2) < C_0^- \cos\phi_\varepsilon^* = C_0^-/(1+\varepsilon)$ and any $\mathbf{x} \in \varepsilon\text{-}IMAC^{(*)}(g,A_2)$ must have

$$A_2\left(\mathbf{x}-\mathbf{x}^A\right) \le (1+\varepsilon)MAC < (1+\varepsilon)\frac{C_0^-}{1+\varepsilon} = C_0^- ,$$

whereas any $\mathbf{y} \in \varepsilon\text{-}IMAC^{(*)}(f,A)$ must have cost $A\left(\mathbf{y}-\mathbf{x}^A\right) \ge C_0^-$. Thus, we would have $\varepsilon\text{-}IMAC^{(*)}(f,A) \cap \varepsilon\text{-}IMAC^{(*)}(g,A) = \emptyset$ and would fail to achieve ε-multiplicative optimality. Hence, we have shown that an ϕ_ε^*-covering is necessary for ε-multiplicative optimality. Moreover, from our definition of ϕ_ε^*, for any $\varepsilon \in (0,\infty)$, $\phi_\varepsilon^* \in (0,\frac{\pi}{2})$ and thus, Lemma 14 is applicable for all ε. From Lemma 14, to have an ϕ_ε^*-covering we must have

$$M \ge \left(\frac{1}{\sin\phi_\varepsilon^*}\right)^{D-2}$$

queries. Using the trigonometric identity $\sin(\arccos(x)) = \sqrt{1-x^2}$, we can substitute for ϕ_ε^* and find

$$\begin{aligned} M &\geq \left(\frac{1}{\sin\left(\arccos\left(\frac{1}{1+\varepsilon}\right)\right)} \right)^{D-2} \\ &\geq \left(\frac{(1+\varepsilon)^2}{(1+\varepsilon)^2 - 1} \right)^{\frac{D-2}{2}} . \end{aligned}$$

■

References

Dana Angluin. Queries and concept learning. *Machine Learning*, 2:319–342, 1988.

Keith Ball. An elementary introduction to modern convex geometry. In *Flavors of Geometry*, volume 31 of *MSRI Publications*, pages 1–58. Cambridge University Press, 1997.

Dimitris Bertsimas and Santosh Vempala. Solving convex programs by random walks. *Journal of the ACM*, 51(4):540–556, 2004.

Christopher M. Bishop. *Pattern Recognition and Machine Learning*. Springer, 2006.

Stephen Boyd and Lieven Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004.

Richard P. Brent. *Algorithms for Minimization without Derivatives*. Prentice-Hall, 1973.

Michael Brückner and Tobias Scheffer. Nash equilibria of static prediction games. In *Advances in Neural Information Processing Systems (NIPS)*, volume 22, pages 171–179. 2009.

Richard L. Burden and J. Douglas Faires. *Numerical Analysis*. Brooks Cole, 7th edition, 2000.

Nilesh Dalvi, Pedro Domingos, Mausam, Sumit Sanghai, and Deepak Verma. Adversarial classification. In *Proceedings of the* 10^{th} *International Conference on Knowledge Discovery and Data Mining*, pages 99–108, 2004.

Sanjoy Dasgupta, Adam Tauman Kalai, and Claire Monteleoni. Analysis of perceptron-based active learning. *Journal of Machine Learning Research*, 10:281–299, 2009.

Vitaly Feldman. On the power of membership queries in agnostic learning. *Journal of Machine Learning Research*, 10:163–182, 2009.

Jörg Flum and Martin Grohe. *Parameterized Complexity Theory*. Texts in Theoretical Computer Science. An EATCS Series. Springer-Verlag, Secaucus, NJ, USA, 2006.

Lee-Ad Gottlieb, Aryeh Kontorovich, and Elchanan Mossel. VC bounds on the cardinality of nearly orthogonal function classes. Technical Report arXiv:1007.4915v2 [math.CO], arXiv, 2011.

Donald R. Jones. A taxonomy of global optimization methods based on response surfaces. *Journal of Global Optimization*, 21(4):345–383, 2001.

Donald R. Jones, Cary D. Perttunen, and Bruce E. Stuckman. Lipschitzian optimization without the Lipschitz constant. *Journal Optimization Theory and Application*, 79(1):157–181, 1993.

Murat Kantarcioglu, Bowei Xi, and Chris Clifton. Classifier evaluation and attribute selection against active adversaries. Technical Report 09-01, Purdue University, 2009.

Tamara G. Kolda, Robert Michael Lewis, and Virginia Torczon. Optimization by direct search: New perspectives on some classical and modern methods. *SIAM Review*, 45(3):385–482, 2003.

Anukool Lakhina, Mark Crovella, and Christophe Diot. Diagnosing network-wide traffic anomalies. In *Proceedings of the Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications (SIGCOMM)*, pages 219–230, 2004.

László Lovász and Santosh Vempala. Simulated annealing in convex bodies and an $O^*(n^4)$ volume algorithm. In *Proceedings of the* 44^{th} *Annual IEEE Symposium on Foundations of Computer Science (FOCS '03)*, pages 650–659, 2003.

László Lovász and Santosh Vempala. Hit-and-run from a corner. In *Proceedings of the* 36^{th} *Annual ACM Symposium on Theory of Computing (STOC)*, pages 310–314, 2004.

Daniel Lowd and Christopher Meek. Adversarial learning. In *Proceedings of the* 11^{th} *International Conference on Knowledge Discovery in Data Mining*, pages 641–647, 2005.

John A. Nelder and Roger Mead. A simplex method for function minimization. *The Computer Journal*, 7(4):308–313, 1965.

Blaine Nelson, Benjamin I. P. Rubinstein, Ling Huang, Anthony D. Joseph, Shing hon Lau, Steven Lee, Satish Rao, Anthony Tran, and J. D. Tygar. Near-optimal evasion of convex-inducing classifiers. In *Proceedings of the* 13^{th} *International Conference on Artificial Intelligence and Statistics (AISTATS)*, pages 549–556, 2010a.

Blaine Nelson, Benjamin I. P. Rubinstein, Ling Huang, Anthony D. Joseph, Steven Lee, Satish Rao, and J. D. Tygar. Query strategies for evading convex-inducing classifiers. Technical Report arXiv:1007.0484v1 [cs.LG], arXiv, 2010b.

Blaine Nelson, Benjamin I. P. Rubinstein, Ling Huang, Anthony D. Joseph, and J. D. Tygar. Classifier evasion: Models and open problems. In *Privacy and Security Issues in Data Mining and Machine Learning*, volume 6549 of *Lecture Notes in Computer Science*, pages 92–98. 2011.

Luis Rademacher and Navin Goyal. Learning convex bodies is hard. In *Proceedings of the* 22^{nd} *Annual Conference on Learning Theory (COLT)*, pages 303–308, 2009.

Greg Schohn and David Cohn. Less is more: Active learning with support vector machines. In *Proceedings of the* 17^{th} *International Conference on Machine Learning (ICML)*, pages 839–846, 2000.

Burr Settles. Active learning literature survey. Computer Sciences Technical Report 1648, University of Wisconsin–Madison, 2009.

Robert L. Smith. The hit-and-run sampler: A globally reaching Markov chain sampler for generating arbitrary multivariate distributions. In *Proceedings of the* 28^{th} *Conference on Winter Simulation (WSC '96)*, pages 260–264, 1996.

Kymie M. C. Tan, Kevin S. Killourhy, and Roy A. Maxion. Undermining an anomaly-based intrusion detection system using common exploits. In *Proceedings of the* 5^{th} *International Conference on Recent Advances in Intrusion Detection (RAID)*, volume 2516 of *Lecture Notes in Computer Science*, pages 54–73, 2002.

David Wagner and Paolo Soto. Mimicry attacks on host-based intrusion detection systems. In *Proceedings of the* 9^{th} *ACM Conference on Computer and Communications Security*, pages 255–264, 2002.

Ke Wang and Salvatore J. Stolfo. Anomalous payload-based network intrusion detection. In *Proceedings of the* 7^{th} *International Conference on Recent Advances in Intrusion Detection (RAID)*, volume 3224 of *Lecture Notes in Computer Science*, pages 203–222, 2004.

Ke Wang, Janak J. Parekh, and Salvatore J. Stolfo. Anagram: A content anomaly detector resistant to mimicry attack. In *Proceedings of the* 9^{th} *International Conference on Recent Advances in Intrusion Detection (RAID)*, volume 4219 of *Lecture Notes in Computer Science*, pages 226–248, 2006.

Aaron D. Wyner. Capabilities of bounded discrepancy decoding. *The Bell System Technical Journal*, 44:1061–1122, 1965.

Journal of Machine Learning Research 13 (2012) 1333-1371 Submitted 10/09; Revised 6/11; Published 5/12

Transfer in Reinforcement Learning via Shared Features

George Konidaris GDK@CSAIL.MIT.EDU
Computer Science and Artificial Intelligence Laboratory
Massachusetts Institute of Technology
32 Vassar Street
Cambridge MA 02139

Ilya Scheidwasser SCHEIDWASSER.I@HUSKY.NEU.EDU
Department of Mathematics
Northeastern University
360 Huntington Avenue
Boston MA 02115

Andrew G. Barto BARTO@CS.UMASS.EDU
Department of Computer Science
University of Massachusetts Amherst
140 Governors Drive
Amherst MA 01003

Editor: Ronald Parr

Abstract

We present a framework for transfer in reinforcement learning based on the idea that related tasks share some common features, and that transfer can be achieved via those shared features. The framework attempts to capture the notion of tasks that are related but distinct, and provides some insight into when transfer can be usefully applied to a problem sequence and when it cannot. We apply the framework to the knowledge transfer problem, and show that an agent can learn a portable shaping function from experience in a sequence of tasks to significantly improve performance in a later related task, even given a very brief training period. We also apply the framework to skill transfer, to show that agents can learn portable skills across a sequence of tasks that significantly improve performance on later related tasks, approaching the performance of agents given perfectly learned problem-specific skills.

Keywords: reinforcement learning, transfer, shaping, skills

1. Introduction

One aspect of human problem-solving that remains poorly understood is the ability to appropriately generalize knowledge and skills learned in one task and apply them to improve performance in another. This effective use of prior experience is one of the reasons that humans are effective learners, and is therefore an aspect of human learning that we would like to replicate when designing machine learning algorithms.

Although reinforcement learning researchers study algorithms for improving task performance with experience, we do not yet understand how to effectively *transfer* learned skills and knowledge from one problem setting to another. It is not even clear which problem sequences allow transfer, which do not, and which do not need to. Although the idea behind transfer in reinforcement learning

seems intuitively clear, no definition or framework exists that usefully formalises the notion of "related but distinct" tasks—tasks that are similar enough to allow transfer but different enough to require it.

In this paper we present a framework for transfer in reinforcement learning based on the idea that related tasks share some common features and that transfer can take place through functions defined only over those shared features. The framework attempts to capture the notion of tasks that are related but distinct, and it provides some insight into when transfer can be usefully applied to a problem sequence and when it cannot. We then demonstrate the framework's use in producing algorithms for knowledge and skill transfer, and we empirically demonstrate the resulting performance benefits.

This paper proceeds as follows. Section 2 briefly introduces reinforcement learning, hierarchical reinforcement learning methods, and the notion of transfer. Section 3 introduces our framework for transfer, which is applied in Section 4 to transfer knowledge learned from earlier tasks to improve performance on later tasks, and in Section 5 to learn transferrable high-level skills. Section 7 discusses the implications and limitations of this work, and Section 8 concludes.

2. Background

The following sections briefly introduce the reinforcement learning problem, hierarchical reinforcement learning methods, and the transfer problem.

2.1 Reinforcement Learning

Reinforcement learning (Sutton and Barto, 1998) is a machine learning paradigm where an agent attempts to learn how to maximize a numerical reward signal over time in a given environment. As a reinforcement learning agent interacts with its environment, it receives a reward (or sometimes incurs a cost) for each action taken. The agent's goal is to use this information to learn to act so as to maximize the cumulative reward it receives over the future.

When the agent's environment is characterized by a finite number of distinct states, it is usually modeled as a finite Markov Decision Process (Puterman, 1994) described by a tuple $M = \langle S, A, P, R \rangle$, where S is the finite set of environment *states* that the agent may encounter; A is a finite set of *actions* that the agent may execute; $P(s'|s,a)$ is the probability of moving to state $s' \in S$ from state $s \in S$ given action $a \in A$; and R is a *reward function*, which given states s and s' and action a returns a scalar reward signal to the agent for executing action a in s and moving to s'.

The agent's objective is to maximize its cumulative reward. If the reward received by the agent at time k is denoted r_k, we denote this cumulative reward (termed *return*) from time t as $R_t = \sum_{i=0}^{\infty} \gamma^i r_{t+i+1}$, where $0 < \gamma \leq 1$ is a *discount factor* that expresses the extent to which the agent prefers immediate reward over delayed reward.

Given a *policy* π mapping states to actions, a reinforcement learning agent may learn a *value function*, V, mapping states to expected return. If the agent is given or learns models of P and R, then it may update its policy as follows:

$$\pi(s) = \arg\max_a \sum_{s'} P(s'|s,a)[R(s,a,s') + \gamma V(s')], \forall s \in S. \quad (1)$$

Once the agent has updated its policy, it must learn a new estimate of V. The repeated execution these two steps (value function learning and policy updates) is known as *policy iteration*. Under

certain conditions (Sutton and Barto, 1998), policy iteration is guaranteed to converge to an *optimal policy* π^* that maximizes return from every state. Policy iteration is usually performed implicitly: the agent simply defines its policy as Equation 1, effectively performing policy iteration after each value function update.

In some applications, states are described by vectors of real-valued features, making the state set a multidimensional continuous state space. (Hereafter we use the term *state space* to refer to both discrete state sets and continuous state spaces.) This creates two problems. First, one must find a way to compactly represent a value function defined on a multi-dimensional real-valued feature space. Second, that representation must facilitate *generalization*: in a continuous state space the agent may never encounter the same state twice and must instead generalize from experiences in nearby states when encountering a novel one.

The most common approximation scheme is *linear function approximation* (Sutton and Barto, 1998). Here, V is approximated by the weighted sum of a vector Φ of *basis functions*:

$$V(\mathbf{s}) = \mathbf{w} \cdot \Phi(\mathbf{s}) = \sum_{i=1}^{n} w_i \phi_i(\mathbf{s}), \tag{2}$$

where ϕ_i is the ith basis function. Thus learning entails obtaining a weight vector $\mathbf{w}$ such that the weighted sum in Equation 2 accurately approximates V. Since V is linear in $\mathbf{w}$, when V's approximation as a weighted sum is not degenerate there is exactly one such optimal $\mathbf{w}$; however, we may represent complex value functions this way because each basis function ϕ_i may be an arbitrarily complex function of $\mathbf{s}$.

The most common family of reinforcement learning methods, and the methods used in this work, are *temporal difference methods* (Sutton and Barto, 1998). Temporal difference methods perform value function learning (and hence policy learning) online, through direct interaction with the environment. For more details see Sutton and Barto (1998).

2.2 Hierarchical Reinforcement Learning and the Options Framework

Much recent research has focused on hierarchical reinforcement learning (Barto and Mahadevan, 2003), where, apart from a given set of primitive actions, an agent can acquire and use higher-level macro actions built out of primitive actions. This paper adopts the options framework (Sutton et al., 1999) for hierarchical reinforcement learning; however, our approach could also be applied in other frameworks, for example the MAXQ (Dietterich, 2000) or Hierarchy of Abstract Machines (HAM) (Parr and Russell, 1997) formulations.

An option o consists of three components:

$$\begin{array}{lll} \pi_o: & (s,a) & \mapsto [0,1], \\ I_o: & s & \mapsto \{0,1\}, \\ \beta_o: & s & \mapsto [0,1], \end{array}$$

where π_o is the *option policy* (which describes the probability of the agent executing action a in state s, for all states in which the option is defined), I_o is the *initiation set* indicator function, which is 1 for states where the option can be executed and 0 elsewhere, and β_o is the *termination condition*, giving the probability of the option terminating in each state (Sutton et al., 1999). The options framework provides methods for learning and planning using options as temporally extended actions in the standard reinforcement learning framework (Sutton and Barto, 1998).

Algorithms for learning new options must include a method for determining when to create an option or alter its initiation set, how to define its termination condition, and how to learn its policy. Policy learning is usually performed by an off-policy reinforcement learning algorithm so that the agent can update many options simultaneously after taking an action (Sutton et al., 1998).

Creation and termination are usually performed by the identification of goal states, with an option created to reach a goal state and terminate when it does so. The initiation set is then the set of states from which the goal is reachable. Previous research has selected goal states by a variety of methods, for example: visit frequency and reward gradient (Digney, 1998), visit frequency on successful trajectories (McGovern and Barto, 2001), variable change frequency (Hengst, 2002), relative novelty (Şimşek and Barto, 2004), clustering algorithms and value gradients (Mannor et al., 2004), local graph partitioning (Şimşek et al., 2005), salience (Singh et al., 2004), causal decomposition (Jonsson and Barto, 2005), and causal analysis of expert trajectories (Mehta et al., 2008). Other research has focused on extracting options by exploiting commonalities in collections of policies over a single state space (Thrun and Schwartz, 1995; Bernstein, 1999; Perkins and Precup, 1999; Pickett and Barto, 2002).

2.3 Transfer

Consider an agent solving a sequence of n problems, in the form of a sequence of Markov Decision Processes $M_1, ..., M_n$. If these problems are somehow "related", and the agent has solved problems $M_1, ..., M_{n-1}$, then it seems intuitively reasonable that the agent should be able to use knowledge gained in their solutions to solve M_n faster than it would be able to otherwise. The transfer problem is the problem of how to obtain, represent and apply such knowledge.

Since transfer hinges on the tasks being related, the nature of that relationship will define how transfer can take place. For example, it is common to assume that all of the tasks have the same state space, action set and transition probabilities but differing reward functions, so that for any i, $M_i = \langle S, A, P, R_i \rangle$. In that case, skills learned in the state space and knowledge about the structure of the state space from previous tasks can be transferred, but knowledge about the optimal policy cannot.

In many transfer settings, however, each task in the sequence has a distinct state space, but the tasks nevertheless seem intuitively related. In the next section, we introduce a framework for describing the commonalities between tasks that have different state spaces and action sets.

3. Related Tasks Share Common Features

Successful transfer requires an agent that must solve a sequence of tasks that are related but distinct—different, but not so different that experience in one is irrelevant to experience in another. How can we define such a sequence? How can we use such a definition to perform transfer?

Consider the illustrative example of an indoor mobile robot required to perform many learning tasks over its lifetime. Although the robot might be equipped with a very rich set of sensors—for example, laser range finders, temperature and pressure gauges—when facing a particular task it will construct a task-specific representation that captures that task's essential features. Such a task-specific representation ensures that the resulting learning task is no more difficult than necessary, and depends on the complexity of the problem rather than the robot (adding more sensors or actuators should not make an easy task hard). In the reinforcement learning setting, a plausible design for such a robot would use a task-specific MDP, most likely designed to be as small as possible

(without discarding information necessary for a solution), and discrete (so that the task does not require function approximation).

Thus, a robot given the tasks of searching for a particular type of object in two different buildings B_1 and B_2 might form two completely distinct discrete MDPs, M_1 and M_2, most likely as topological maps of the two buildings. Then even though the robot should be able to share information between the two problems, without further knowledge there is no way to transfer information between them *based only on their description as MDPs*, because the state labels and transition probabilities of M_1 and M_2 need have no relation at all.

We argue that finding relationships between pairs of arbitrary MDPs is both unnecessarily difficult and misses the connection between these problems. The problems that such a robot might encounter are all related *because they are faced by the same agent*, and therefore the same sensor features are present in each, even if those shared features are abstracted away when the problems are framed as MDPs. If the robot is seeking a heat-emitting object in both B_1 and B_2, it should be able to learn after solving B_1 that its temperature gauge is a good predictor of the object's location, and use it to better search B_2, even though its temperature gauge reading does not appear as a feature in either MDP.

When trying to solve a single problem, we aim to create a minimal problem-specific representation. When trying to transfer information across a sequence of problems, we should instead concentrate on what is common across the sequence. We therefore propose that *what makes tasks related is the existence of a feature set that is shared and retains the same semantics across tasks*. To define what we mean by a feature having the same semantics across tasks, we define the notion of a *sensor*.

Consider a parametrized class of tasks $\Gamma(\theta)$, where Γ returns a task instance given parameter $\theta \in \Theta$. For example, Γ might be the class of square gridworlds, and θ might fix obstacle and goal locations and size. We can obtain a sequence of tasks $M_1, ..., M_n$ via a sequence of task parameters $\theta_1, ..., \theta_n$.

Definition 1 *A sensor ξ is a function mapping a task instance parameter $\theta \in \Theta$ and state $s_\theta \in S_\Theta$ of the task obtained using θ to a real number f:*

$$\xi : (\theta, s_\theta) \mapsto f.$$

The important property of ξ is that it is a function defined over all tasks in Γ: it produces a feature, f, that describes some property of an environment given that environment's parameters and current state. For example, f might describe the distance from a robot in a building to the nearest wall; this requires both the position of the robot in the building (the problem state) and the layout of the building (the environment parameters). The feature f has the same semantics across tasks because it is generated by the same function in each task instance.[1]

An agent may in general be equipped with a suite of such sensors, from which it can read at any point to obtain a feature vector. We call the space generated by the resulting features an *agent-space*, because it is a property of the agent rather than any of the tasks individually, as opposed to the problem-specific state space used to solve each problem (which we call a *problem-space*).

We note that in some cases the agent-space and problem-spaces used for a sequence of tasks may be related, for example, each problem-space might be formed by appending a task-specific amount

1. We exclude degenerate cases, for example where ξ simply uses θ as an index and produces completely different outputs for different values of θ, or where ξ returns a constant, or completely random, value.

of memory to agent-space. However, in general it may not be possible to recover an agent-space descriptor from a problem-space descriptor, or vice versa. The functions mapping the environment to each descriptor are distinct and must be designed (or learned outside of the reinforcement learning process) with different objectives.

We now model each problem in a sequence as an MDP augmented with an agent-space, writing the augmented MDP corresponding to the ith problem as:

$$M_i = \langle S_i, A_i, P_i, R_i, D \rangle,$$

where D (the agent-space) is a feature space defined across all tasks. For any state in any of the environments, the agent also obtains an observation (or *descriptor*) $d \in D$, the features of which have the same semantics across all tasks.

The core idea of our framework is that task learning occurs in problem-space, and transfer can occur via agent-space. If we have an MDP augmented with features that are known to be shared, we can use those shared features as a bridge across which knowledge can be transferred. This leads to the following definition:

Definition 2 *A sequence of tasks is* related *if that sequence has a non-empty agent-space—that is, if a set of shared features exist in all of the tasks.*

A further definition will prove useful in understanding when the transfer of information about the value function is useful:

Definition 3 *We define a sequence of related tasks to be* reward-linked *if the reward function for all tasks is the same sensor, so that rewards are allocated the same way for all tasks (for example, reward is always x for finding food).*

A sequence of tasks must be (at least approximately) reward-linked if we aim to transfer information about the optimal value function: if the reward functions in two tasks use different sensors then there is no reason to hope that their value functions contain useful information about each other.

If a sequence of tasks is related, we may be able to perform effective transfer by taking advantage of the shared space. If no such space exists, we cannot transfer across the sequence because there is no view (however abstract or lossy) in which the tasks share common features. If we can find an agent-space that is also usable as a problem-space for every task in the sequence, then we can treat the sequence as a set of tasks in the same space (by using D directly as a state space) and perform transfer directly by learning about the structure of this space. If in addition the sequence is reward-linked, then the tasks are not distinct and transfer is trivial because we can view them as a single problem. However, there may be cases where a shared problem-space exists but results in slow learning, and using task-specific problem-spaces coupled with a transfer mechanism is more practical.

We can therefore define the working hypothesis of this paper as follows:

> We can usefully describe two tasks as related when they share a common feature space, which we term an *agent-space*. If learning to solve each individual task is possible and feasible in agent-space, then transfer is trivial: the tasks are effectively a single task and we can learn a single policy in agent-space for all tasks. If it is not, then transfer between two tasks can nevertheless be effected through agent-space, either through the

transfer of knowledge about the value function (when the tasks are reward-linked), or through the transfer of skills defined in agent-space.

In the following sections we use this framework to build agents that perform these two different types of transfer. Section 4 shows that an agent can transfer value-functions learned in agent-space to significantly decrease the time taken to find an initial solution to a task, given experience in a sequence of related and reward-linked tasks. In Section 5 we show that an agent can learn portable high-level skills directly in agent-space which can dramatically improve task performance, given experience in a sequence of related tasks.

4. Knowledge Transfer

In this section, we show that agents that must repeatedly solve the same type of task (in the form of a sequence of related, reward-linked tasks) can transfer useful knowledge in the form of a *portable shaping function* that acts as an initial value function and thereby endows the agent with an initial policy. This significantly improves initial performance in later tasks, resulting in agents that can, for example, learn to solve difficult tasks quickly after being given a set of relatively easy training tasks.

We empirically demonstrate the effects of knowledge transfer using a relatively simple demonstration domain (a rod positioning task with an artificial agent space) and a more challenging domain (Keepaway). We argue (in Section 4.5) that this has the effect of creating agents which can learn their own heuristic functions.

4.1 Shaping

Shaping is a popular method for speeding up reinforcement learning in general, and goal-directed exploration in particular (Dorigo and Colombetti, 1998). Although this term has been applied to a variety of different methods within the reinforcement learning community, only two are relevant here. The first is the gradual increase in complexity of a single task toward some given final level (for example, Randløv and Alstrøm 1998; Selfridge et al. 1985), so that the agent can safely learn easier versions of the same task and use the resulting policy to speed learning as the task becomes more complex.[2] Unfortunately, this type of shaping does not generally transfer between tasks—it can only be used to gently introduce an agent to a single task, and is therefore not suited to a sequence of distinct tasks.

Alternatively, the agent's reward function could be augmented through the use of intermediate shaping rewards or "progress indicators" (Matarić, 1997) that provide an augmented (and hopefully more informative) reinforcement signal to the agent. This has the effect of shortening the reward horizon of the problem—the number of correct actions the agent must execute before receiving a useful reward signal (Laud and DeJong, 2003). Ng et al. (1999) proved that an arbitrary externally specified reward function could be included as a potential-based shaping function in a reinforcement learning system without modifying its optimal policy. Wiewiora (2003) showed that this is equivalent to using the same reward function as a non-uniform initial state value function, or with

2. We note that this definition of shaping is closest to its original meaning in the psychology literature, where it refers to a process by which an experimenter rewards an animal for behavior that progresses toward the completion of a complex task, and thereby guides the animal's learning process. As such it refers to a training technique, not a learning mechanism (see Skinner, 1938).

a small change in action selection, as an initial state-action value function (Wiewiora et al., 2003). Thus, we can use any function we like as an initial value function for the agent, even if (as is often the case in function approximation) it is not possible to directly initialize the value function. The major drawback is that designing such a shaping function requires significant engineering effort. In the following sections we show that an agent can learn its own shaping function from experience across several related, reward-linked tasks without having it specified in advance.

4.2 Learning Portable Shaping Functions

As before, consider an agent solving n problems with MDPs $M_1, ..., M_n$, each with their own state space, denoted $S_1, ..., S_n$ and augmented with agent-space features. We associate a four-tuple σ_i^j with the ith state in M_j:

$$\sigma_i^j = \langle s_i^j, d_i^j, r_i^j, v_i^j \rangle,$$

where s_i^j is the usual problem-space state descriptor (sufficient to distinguish this state from the others in S_j), d_i^j is the agent-space descriptor, r_i^j is the reward obtained at the state and v_i^j is the state's value (expected total reward for action starting from the state). The goal of value-function based reinforcement learning is to obtain the v_i^j values for each state in the form of a value function V_j:

$$V_j : s_i^j \mapsto v_i^j.$$

V_j maps problem-specific state descriptors to expected return, but it is not portable between tasks, because the form and meaning of s_i^j (as a problem-space descriptor) may change from one task to another. However, the form and meaning of d_i^j (as an agent-space descriptor) does not change. Since we want an estimator of return that is portable across tasks, we introduce a new function L that is similar to each V_j, but that estimates expected return given an agent-space descriptor:

$$L : d_i^j \mapsto v_i^j.$$

L is also a value function, but it is defined over portable agent-space descriptors rather than problem-specific state space descriptors. As such, we could consider it a form of feature-based value function approximation and update it online (using a suitable reinforcement learning algorithm) during each task. Alternatively, once an agent has completed some task S_j and has learned a good approximation of the value of each state using V_j, it can use its (d_i^j, v_i^j) pairs as training examples for a supervised learning algorithm to learn L. Since L is portable, we can in addition use samples from multiple related, reward-linked tasks.

After a reasonable amount of training, L can be used to estimate a value for newly observed states in any future related and reward-linked tasks. Thus, when facing a new task M_k, the agent can use L to provide a good initial estimate for V_k that can be refined using a standard reinforcement learning algorithm. Alternatively (and equivalently), L could be used as an external shaping reward function.

4.3 A Rod Positioning Experiment

In this section we empirically evaluate the potential benefits of a learned shaping function in a rod positioning task (Moore and Atkeson, 1993), where we add a simple artificial agent space that can be easily manipulated for experimental purposes.

Each task consists of a square workspace that contains a rod, some obstacles, and a target. The agent is required to maneuver the rod so that its tip touches the target (by moving its base coordinate or its angle of orientation) while avoiding obstacles. An example 20x20 unit task and solution path are shown in Figure 1.

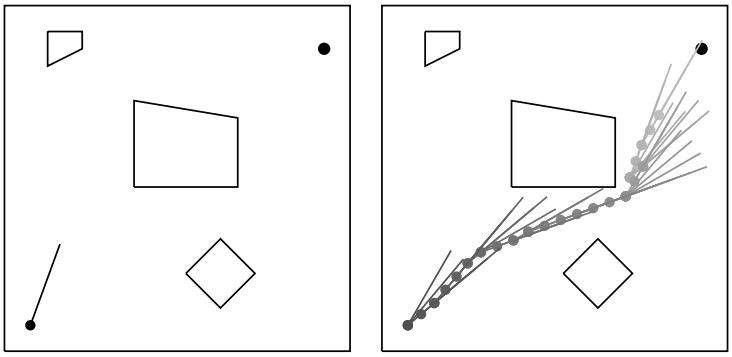


Figure 1: A 20x20 rod positioning task and one possible solution path.

Following Moore and Atkeson (1993), we discretize the state space into unit x and y coordinates and 10° angle increments. Thus, each state in the problem-space can be described by two coordinates and one angle, and the actions available to the agent are movement of one unit in either direction along the rod's axis, or a 10^o rotation in either direction. If a movement causes the rod to collide with an obstacle, it results in no change in state, so the portions of the state space where any part of the rod would be interior to an obstacle are not reachable. The agent receives a reward of -1 for each action, and a reward of 1000 when reaching the goal (whereupon the current episode ends).

We augment the task environment with five beacons, each of which emits a signal that drops off with the square of the Euclidean distance from a strength of 1 at the beacon to 0 at a distance of 60 units. The tip of the rod has a sensor array that can detect the values of each of these signals separately at the adjacent state in each action direction. Since these beacons are present in every task, the sensor readings are an agent-space and we include an element in the agent that learns L and uses it to predict reward for each adjacent state given the five signal levels present there.

The usefulness of L as a reward predictor will depend on the relationship between beacon placement and reward across a sequence of individual rod positioning tasks. Thus we can consider the beacons as simple abstract signals present in the environment, and by manipulating their placement (and therefore their relationship to reward) across the sequence of tasks, we can experimentally evaluate the usefulness of various forms of L.

4.3.1 EXPERIMENTAL STRUCTURE

In each experiment, the agent is exposed to a sequence of training experiences, during which it is allowed to update L. After each training experience, it is evaluated in a large test case, during which it is *not* allowed to update L.

Each individual training experience places the agent in a small task, randomly selected from a randomly generated set of 100 such tasks, where it is given sufficient time to learn a good solution. Once this time is up, the agent updates L using the value of each visited state and the sensory signal

present at it, before it is tested on the much larger test task. All state value tables are cleared between training episodes.

Each agent performed reinforcement learning using Sarsa(λ) ($\lambda = 0.9, \alpha = 0.1, \gamma = 0.99$, $\varepsilon = 0.01$) in problem-space and used training tasks that were either 10x10 (where it was given 100 episodes to converge in each training task), or 15x15 (when it was given 150 episodes to converge), and tested in a 40x40 task.[3] L was a linear estimator of reward, using either the five beacon signal levels and a constant as features (requiring 6 parameters, and referred to as the linear model) or using those with five additional features for the square of each beacon value (requiring 11 parameters, referred to as the quadratic model). All parameters were initialized to 0, and learning for L was accomplished using gradient descent with $\alpha = 0.001$. We used two experiments with different beacon placement schemes.

4.3.2 FOLLOWING A HOMING BEACON

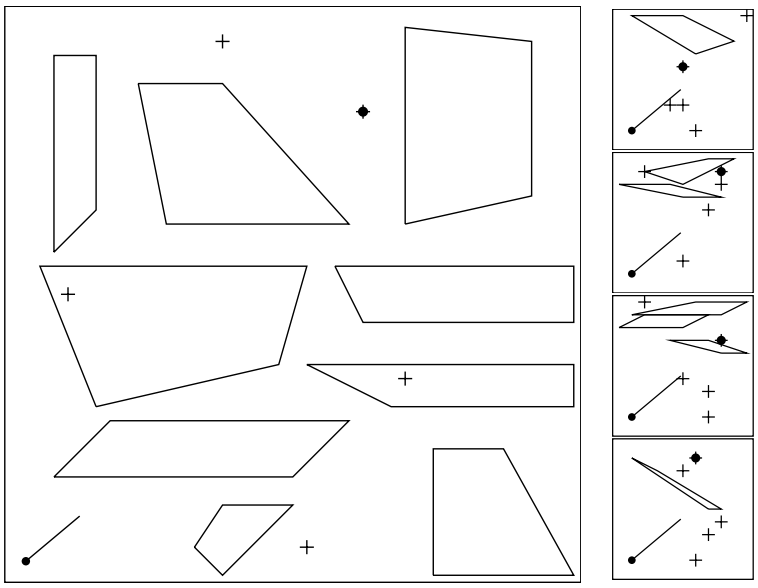
In the first experiment, we always placed the first beacon at the target location, and randomly distributed the remainder throughout the workspace. Thus a high signal level from the first beacon predicts high reward, and the others should be ignored. This is a very informative indication of reward that should be easy to learn, and can be well approximated even with a linear L. Figure 2 shows the 40x40 test task used to evaluate the performance of each agent, and four sample 10x10 training tasks.

Figure 3(a) shows the number of steps (averaged over 50 runs) required to first reach the goal in the test task, against the number of training tasks completed by the agent for the four types of learned shaping elements (linear and quadratic L, and either 10x10 or 15x15 training tasks). It also shows the average number of steps required by an agent with a uniform initial value of 0 (agents with a uniform initial value of 500 performed similarly while first finding the goal). Note that there is just a single data point for the uniform initial value agents (in the upper left corner) because their performance does not vary with the number of training experiences.

Figure 3(a) shows that training significantly lowers the number of steps required to initially find the goal in the test task in all cases, reducing it after one training experience from over $100,000$ steps to at most just over $70,000$, and by six episodes to between $20,000$ and $40,000$ steps. This difference is statistically significant (by a t-test, $p < 0.01$) for all combinations of L and training task sizes, even after just a single training experience. Figure 3(a) also shows that the complexity of L does not appear to make a significant difference to the long-term benefit of training (probably because of the simplicity of the reward indicator), but training task size does. The difference between the number of steps required to first find the goal for 10x10 and 15x15 training task sizes is statistically significant ($p < 0.01$) after 20 training experiences for both linear and quadratic forms of L, although this difference is clearer for the quadratic form, where it is significant after 6 training experiences.

Figure 3(b) shows the number of steps (averaged over 50 runs) required to reach the goal as the agents repeat episodes in the test task, after completing 20 training experiences (note that L is never updated in the test task), compared to the number of steps required by agents with value tables uniformly initialized to 0 and 500. This illustrates the difference in overall learning performance on a single new task between agents that have had many training experiences and agents that have

3. We note that in general the tasks used to train the agent need not be smaller than the task used to test it. We used small training tasks in this experiment to highlight the fact that the size of problem-space may differ between related tasks.



a b

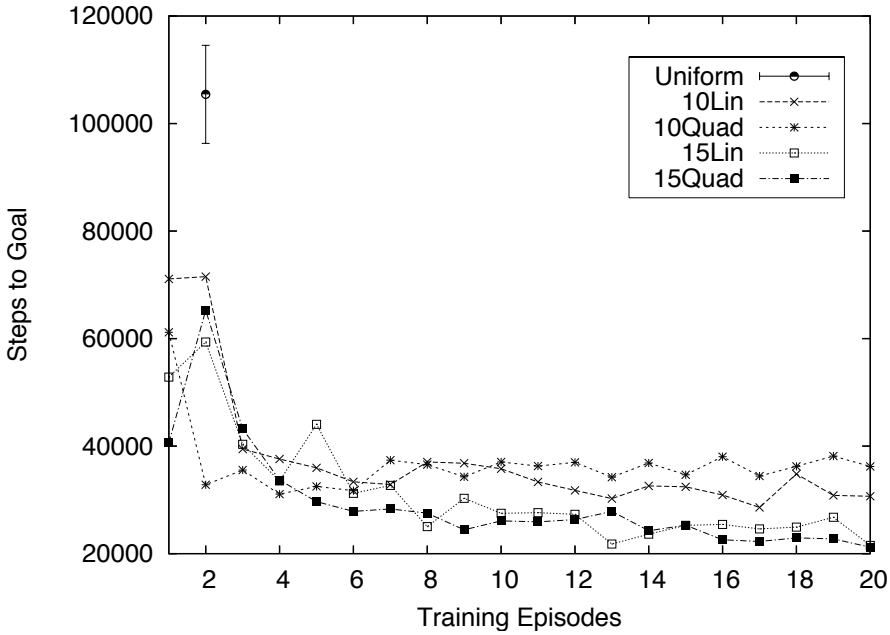
Figure 2: The homing experiment 40x40 test task (a) and four sample 10x10 training tasks (b). Beacon locations are shown as crosses, and the goal is shown as a large dot. Note that the first beacon is on the target in each task. The optimal solution for the test task requires 69 steps.

not. Figure 3(b) shows that the learned shaping function significantly improves performance during the first few episodes of learning, as expected. It also shows that the number of episodes required for convergence is roughly the same as that of an agent using a uniformly optimistic value table initialization of 500, and slightly longer than that of an agent using a uniformly pessimistic value table initialization of 0. This suggests that once a solution is found the agent must "unlearn" some of its overly optimistic estimates to achieve convergence. Note that a uniform initial value of 0 works well here because it discourages extended exploration, which is unnecessary in this domain.

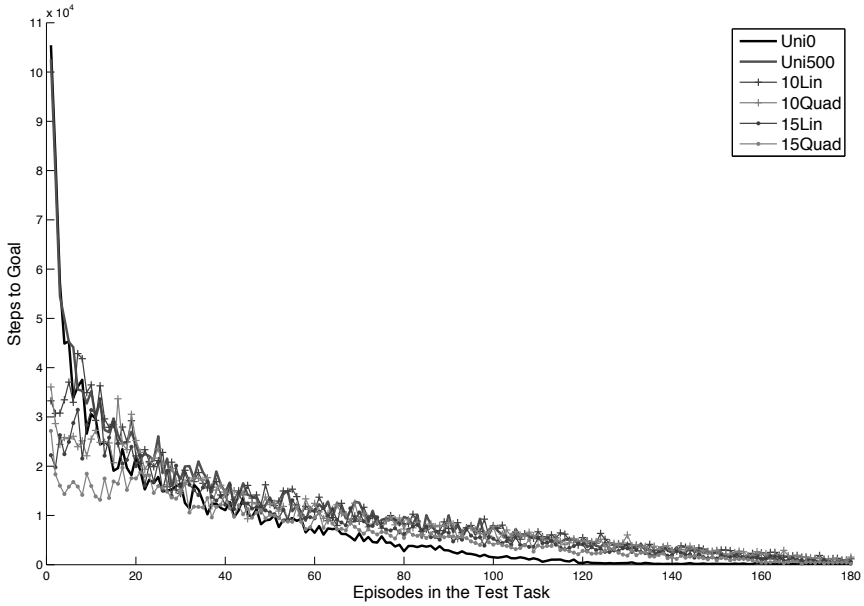
4.3.3 FINDING THE CENTER OF A BEACON TRIANGLE

In the second experiment, we arranged the first three beacons in a triangle at the edges of the task workspace, so that the first beacon lay to the left of the target, the second directly above it, and the third to its right. The remaining two were randomly distributed throughout the workspace. This provides a more informative signal, but results in a shaping function that is harder to learn. Figure 4 shows the 10x10 sample training tasks given in Figure 2 after modification for the triangle experiment. The test task was similarly modified.

Figure 5(a) shows the number of steps initially required to reach the goal for the triangle experiment, again showing that completing even a single training task results in a statistically significant ($p < 0.01$ in all cases) reduction from the number required by an agent using uniform initial values, from just over $100,000$ steps to at most just over $25,000$ steps after a single training episode. Figure



(a) The average number of steps required to first reach the goal in the homing test task, for agents that have completed varying numbers of training task episodes.



(b) Steps to reward against episodes in the homing test task for agents that have completed 20 training tasks.

Figure 3: Results for the homing task.

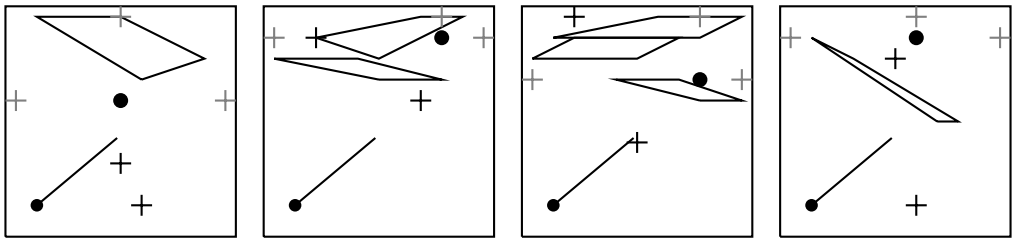


Figure 4: Sample 10x10 training tasks for the triangle experiment. The three beacons surrounding the goal in a triangle are shown in gray.

5(a) also shows that there is no significant difference between forms of L and size of training task. This suggests that extra information in the agent-space more than makes up for a shaping function being difficult to accurately represent—in all cases the performance of agents learning using the triangle beacon arrangement is better than that of those learning using the homing beacon arrangement. Figure 5(b) shows again that the initial few episodes of repeated learning in the test task are much faster, and again that the total number of episodes required to converge lies somewhere between the number required by an agent initializing its value table pessimistically to 0 and one initializing it optimistically to 500.

4.3.4 SENSITIVITY ANALYSIS

So far, we have used shared features that are accurate in the sense that they provide a signal that is uncorrupted by noise and that has exactly the same semantics across tasks. In this section, we empirically examine how sensitive a learned shaping reward might be to the presence of noise, both in the features and in their role across tasks.

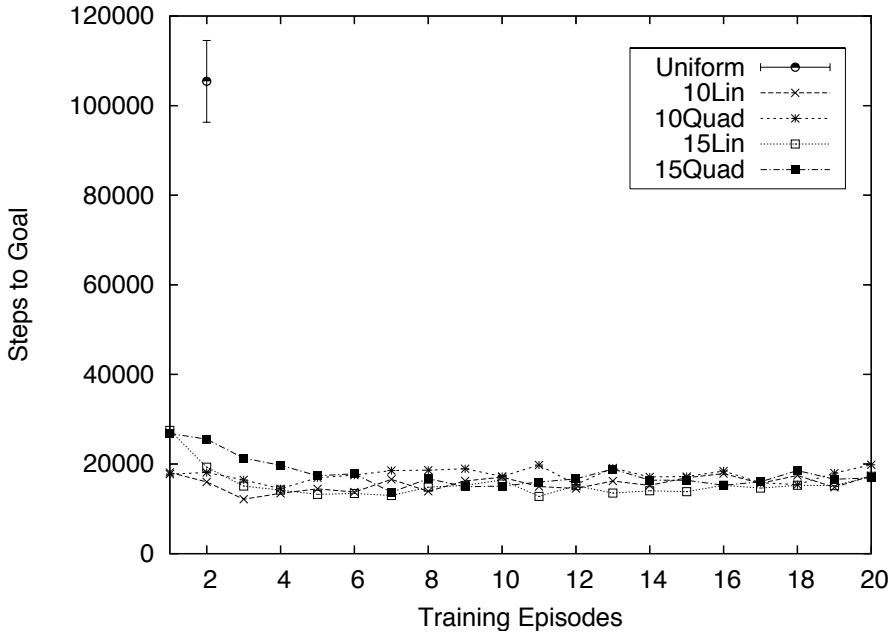
To do so, we repeat the above experiments (using training tasks of size 15, and a quadratic approximator) but with only a single beacon whose position is given by the following formula:

$$\mathbf{b} = (1-\eta)\mathbf{g} + \eta\mathbf{r},$$

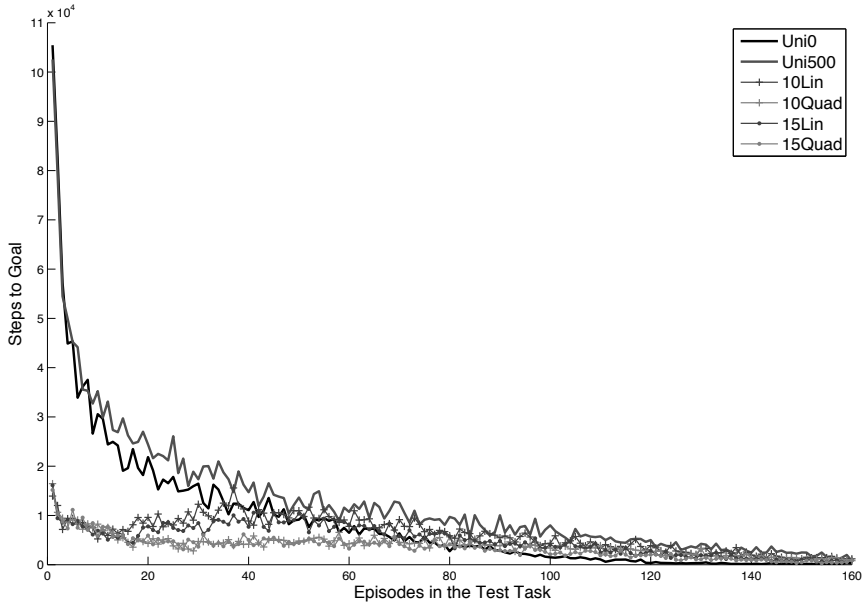
where $\mathbf{g}$ is the coordinate vector of the target, $\eta \in [0,1]$ is a noise parameter, and $\mathbf{r}$ is a co-ordinate vector generated uniformly at random. Thus, when $\eta = 0$ we have no noise and the beacon is always placed directly over the goal; when $\eta = 1$, the beacon is placed randomly in the environment. Varying η between 0 and 1 allows us to manipulate the amount of noise present in the beacon's placement, and hence in the shared feature used to learn a portable shaping function. We consider two scenarios.

In the first scenario, the same η value is used to place the beacon in both the training and the test problem. This corresponds to a signal that is perturbed by noise, but whose semantics remain the same in both source and target tasks. This measures how sensitive learned shaping rewards are to feature noise, and so we call this the *noisy-signal* task. The results are shown in Figure 6(a) and 6(b).

Figure 6(a) measures the number of steps required to complete the first episode in the large test problem, given experience in various numbers of training problems and varying levels of noise. The results show that transfer is fairly robust to noise, resulting in an improvement over starting from

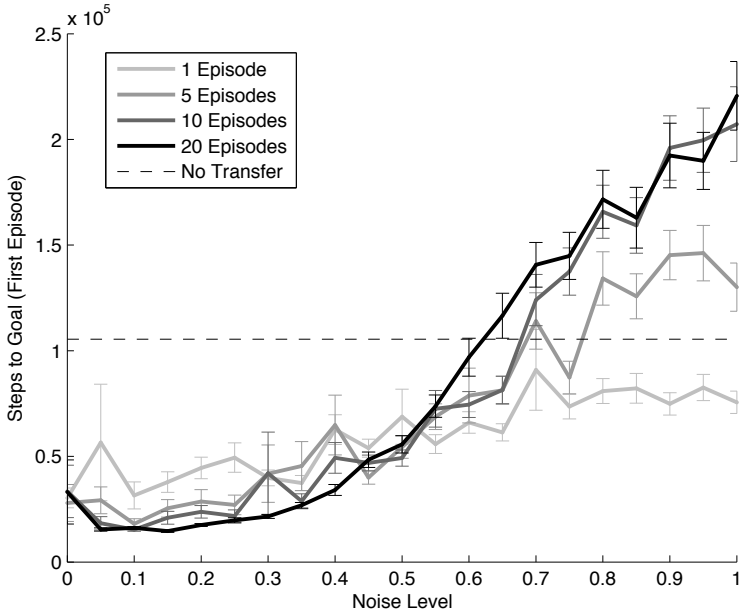


(a) The average number of steps required to first reach the goal in the triangle test task, for agents that have completed varying number of training task episodes.

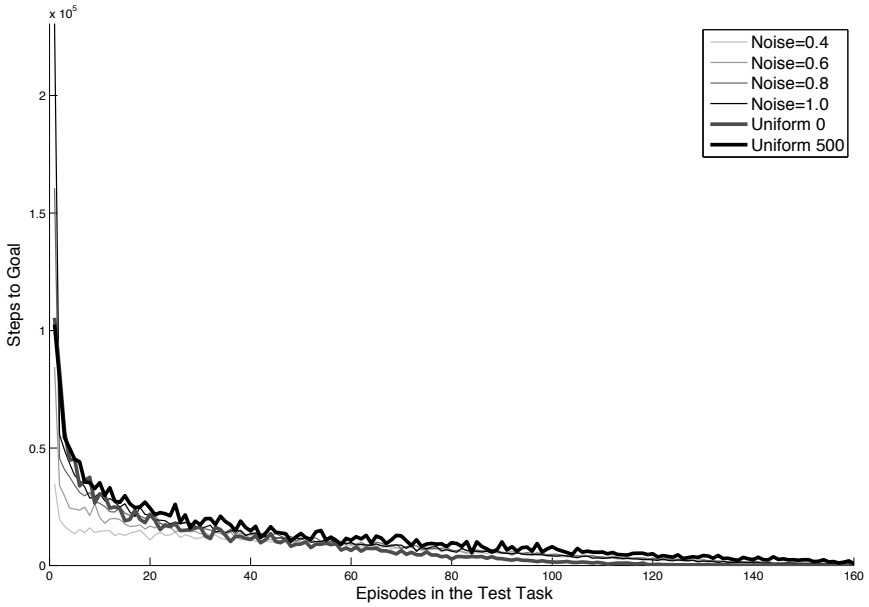


(b) Steps to reward against episodes in the triangle test task for agents that have completed 20 training tasks.

Figure 5: Results for the triangle task.



(a) The average number of steps required to first reach the test task goal given a predictor learned using a noisy signal.



(b) Steps to reward against episodes in the test task for agents that have completed 20 training task episodes using a noisy signal.

Figure 6: Results for the *noisy-signal* task.

scratch that drops with increased noise but still does better until $\eta > 0.6$, when the feature has more noise than signal.

Higher levels of noise more severely affects agents that have seen higher numbers of training problems, until a performance floor is reached between 5 and 10 training problems. This reflects the training procedure used to learn L, whereby each training problem results in a small adjustment of L's parameters and those adjustments accumulate over several training episodes.

Similarly, Figure 6(b) shows learning curves in the test problem for agents that have experienced 20 test problems, with varying amounts of noise. We see that, although these agents often do worse than learning from scratch in the first episode, they subsequently do better when $\eta < 1$, and again converge at roughly the same rate as agents that use an optimistic initial value function.

In the second scenario, η is zero in the training problems, but non-zero in the test problem. This corresponds to a feature which has slightly different semantics in the source and target tasks, and thus measures how learning is affected by an imperfect or approximate choice of agent space features. We call this the *noisy-semantics* task.

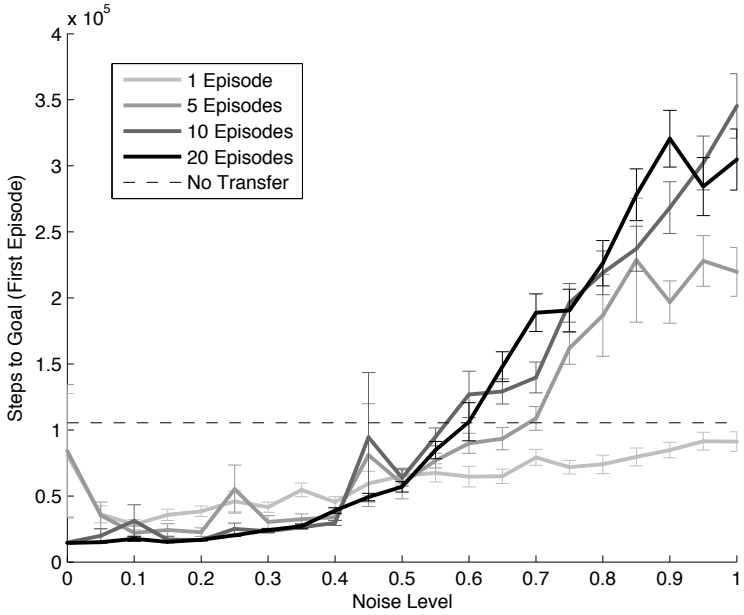
Results for the noisy-semantics task are given in Figures 7(a) and 7(b). These two graphs show that transfer achieves a performance benefit when $\eta < 0.5$—when there is at least as much signal as noise—and the more training problems the agent has solved, the worse its performance will be when $\eta = 1$. However, the possible performance penalty for high η is more severe—an agent using a learned shaping function that rewards it for following a beacon signal may take nearly four times as long to first solve the test problem when that feature becomes random (at $\eta = 1$). Again, however, when $\eta < 1$ the agents recover after their first episode to outperform agents that learn from scratch within the first few episodes.

4.3.5 SUMMARY

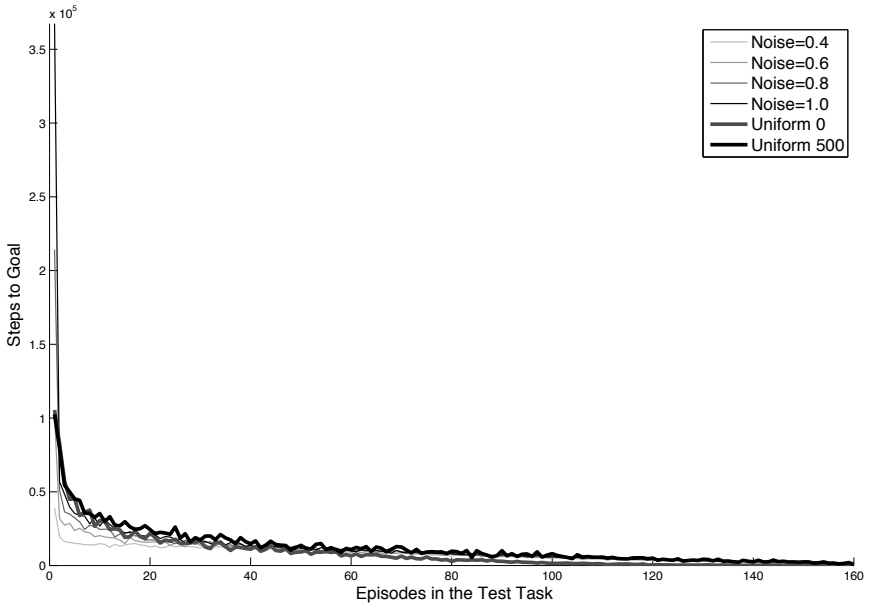
The first two experiments above show that an agent able to learn its own shaping rewards through training can use even a few training experiences to significantly improve its initial policy in a novel task. They also show that such training results in agents with convergence characteristics similar to that of agents using uniformly optimistic initial value functions. Thus, an agent that learns its own shaping rewards can improve its initial speed at solving a task when compared to an agent that cannot, but it will not converge to an approximately optimal policy in less time (as measured in episodes).

The results also seem to suggest that a better training environment is helpful but that its usefulness decreases as the signal predicting reward becomes more informative, and that increasing the complexity of the shaping function estimator does not appear to significantly improve the agent's performance. Although this is a very simple domain, this suggests that given a rich signal from which to predict reward, even a weak estimator of reward can greatly improve performance.

Finally, our third pair of experiments suggest that transfer is relatively robust to noise, both in the features themselves and in their relationship across tasks, resulting in performance benefits provided there is at least as much useful information in the features as there is noise. Beyond that, however, agents may experience negative transfer where either noisy features or an imperfect or approximate set of agent-space features result in poor learned shaping functions.



(a) The average number of steps required to first reach the test task goal given a predictor learned using features with imperfectly preserved semantics.



(b) Steps to reward against episodes in the test task for agents that have completed 20 training task episodes using features with imperfectly preserved semantics.

Figure 7: Results for the *noisy-semantics* task.

4.4 Keepaway

In this section we evaluate knowledge transfer using common features in Keepaway (Stone et al., 2005), a robot soccer domain implemented in the RoboCup soccer simulator. Keepaway is a challenging domain for reinforcement learning because it is multi-agent and has a high-dimensional continuous state space. We use Keepaway to illustrate the use of learned shaping rewards on a standard but challenging benchmark that has been used in other transfer studies (Taylor et al., 2007).

Keepaway has a square field of a given size, which contains players and a ball. Players are divided into two groups: keepers, who are originally in possession of the ball and try to stay in control of it, and takers, who attempt to capture the ball from the keepers. This arrangement is depicted in Figure 8.

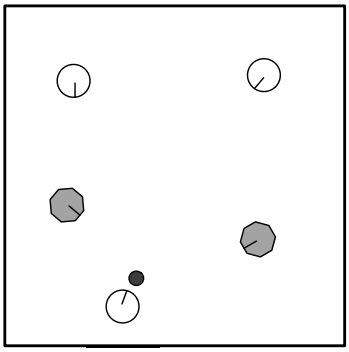


Figure 8: The Keepaway Task. The keepers (white circles) must keep possession of the ball and not allow the takers (gray octagons) to take it away. This diagram depicts 3v2 Keepaway, where there are 3 keepers and 2 takers.

Each episode begins with the takers in one corner of the field and the keepers randomly distributed. The episode ends when the ball goes out of bounds, or when a taker ends up in possession of the ball (i.e., within a small distance of the ball for a specified period of time). The goal of the keepers is then to maximize the duration of the episode. At each time step, the objective of learning is to modify the behavior of the keeper currently in possession of the ball. The takers and other keepers act according to simple hand-coded behaviors. Keepers not in possession of the ball try to open a clear shot from the keeper with the ball to themselves and attempt to receive the ball when it is passed to them. Takers either try to block keepers that are not holding the ball, try to take the ball from the keeper in possession, or try to intercept a pass.

Rather than using the primitive actions of the domain, keepers are given a set of predefined options. The options available to the keeper in possession of the ball are HoldBall (remain stationary while keeping the ball positioned away from the takers) and PassBall(k) (pass the ball to the kth other keeper). Since only the keeper in possession of the ball is acting according to the reinforcement learner at any given time, multiple keepers may learn during each episode; each keeper's learner runs separately.

The state variables are continuous and defined according to the center of the board and the location of the players, with the number of variables depending on the number of players. For example, 3v2 Keepaway (three keepers versus two takers) has thirteen state variables: the distance

from $K1$ (the keeper in possession) to each other player, the minimum angles BAC for each other keeper (where B is the other keeper, A is $K1$, and C is a taker—this measures how "open" each other keeper is to a pass), the distance from each player to the center, and the minimum distance from each other keeper to a taker. The number of state variables is $4K + 2T - 3$, for K keepers and T takers. We used a field measuring 20x20 units for 3v2 games, and a field measuring 30x30 for 4v3 and 5v4. For a more detailed description of the Keepaway domain we refer the reader to Stone et al. (2005).

4.4.1 EXPERIMENTAL STRUCTURE

In the previous section, we studied transferring portable shaping functions from a varying number of smaller randomly generated source tasks to a fixed larger target task. In Keepaway, instances of the domain are obtained by fixing the number of keepers and the number of takers. Since we cannot obtain experience in more than a few distinct source tasks, in this section we instead study the effect of varying amounts of training time in a source task on performance in a target task.

We thus studied transfer from 3v2 Keepaway to 4v3 and 5v4 Keepway, and from 4v3 to 5v4; these are the most common Keepaway configurations and are the same configurations studied by Taylor and Stone (2005). In all three cases we used the state variables from 5v4 as an agent-space. When a state variable is not defined (e.g., the distance to the 4th keeper in 3v2 Keepaway), we set distances and angles to keepers to 0, and distances and angles to takers to their maximum value, which effectively simulates their being present but not meaningfully involved in the current state. We employed linear function approximation with Sarsa (Sutton and Barto, 1998) using 32 radial basis functions per state variable, tiling each variable independently of the others, following and using the same parameters as Stone et al. (2005).

We performed 20 separate runs for each condition. We first ran 20 baseline runs for 3v2, 4v3, and 5v4 Keepaway, saving weights for the common space for each 3v2 and 4v3 run at 50, 250, 500, 1000, 2000, and 5000 episodes. Then for each set of common space weights from a given number of episodes, we ran 20 transfer runs. For example, for the 3v2 to 5v4 transfer with 250-episode weights, we ran 20 5v4 transfer runs, each of which used one of the 20 saved 250-episode 3v2 weights.

Because of Keepaway's high variance, and in order to provide results loosely comparable with Taylor and Stone (2005), we evaluated the performance of transfer in Keepaway by measuring the average time required to reach some benchmark performance. We selected a benchmark time for each setting (3v2, 4v3 or 5v4) which the baseline learner could consistently reach by about 5000 episodes. This benchmark time T is considered reached at episode n when the average of the times from $n - 500$ to $n + 500$ is at least T; this window averaging compensates Keepaway's high performance variance. The benchmark times for each domain were, in order, 12.5 seconds, 9.5 seconds, and 8.5 seconds.

4.4.2 RESULTS

Table 1 shows the results of performing transfer from 3v2 Keepaway to 4v3 Keepaway. Results are reported as time (in simulator hours) to reach the benchmark in the target task (4v3 Keepaway) given a particular number of training episodes in the source task (3v2 Keepaway), and the total time (source task training time plus time to reach the benchmark in the target task, in simulator hours). We can thereby evaluate whether the agents achieve *weak transfer*—where there is an improvement

in the target task with experience in the source task—by examining the third column (average 4v3 time), and *strong transfer*—where the sum of the time spent in both source and target tasks is lower than that taken when learning the target task in isolation—by examining the fourth column (average total time).

The results show that training in 3v2 decreases the amount of time required to reach the benchmark in 4v3, which shows that transfer is successful in this case and weak transfer is achieved. However, the total (source and target) time to benchmark never decreases with experience in the source task, so strong transfer is not achieved.

# 3v2 Episodes	Ave. 3v2 Time	Ave 4v3 Time	Ave. Total Time	Std. Dev.
0	0.0000	5.5616	**5.5616**	1.5012
50	0.0765	5.7780	5.8544	0.8870
250	0.3919	5.4763	5.8682	1.2399
500	0.8871	5.1192	6.0063	0.9914
1000	1.8166	4.7380	6.5546	1.2680
2000	3.9908	3.1295	7.1203	1.1465
5000	14.7554	**1.4236**	16.1790	0.2738

Table 1: Results of transfer from 3v2 Keepaway to 4v3 Keepaway.

Figure 9 shows sample learning curves for agents learning from scratch and agents using transferred knowledge from 5000 episodes of 3v2 Keepaway, demonstrating that agents that transfer knowledge start with better policies and learn faster.

Table 2 shows the results of transfer from 3v2 Keepaway (Table 1(a)) and 4v3 Keepaway (Table 1(b)) to 5v4 Keepaway. As before, in both cases more training on the easier task results in better performance in 5v4 Keepaway, demonstrating that weak transfer is achieved. However, the least total time (including training time on the source task) is obtained using a moderate amount of source task training, and so when transferring to 5v4 we achieve strong transfer.

Finally, Table 3 shows the results of transfer for shaping functions learned on both 3v2 and 4v3 Keepaway, applied to 5v4 Keepaway. Again, more training time obtains better results although over-training appears to be harmful.

These results show that knowledge transfer through agent-space can achieve effective transfer in a challenging problem and can do so in multiple problems through the same set of common features.

4.5 Discussion

The results presented above suggest that agents that employ reinforcement learning methods can be augmented to use their experience to learn their own shaping rewards. This could result in agents that are more flexible than those with pre-engineered shaping functions. It also creates the possibility of training such agents on easy tasks as a way of equipping them with knowledge that will make harder tasks tractable, and is thus an instance of an autonomous developmental learning system (Weng et al., 2000).

In some situations, the learning algorithm chosen to learn the shaping function, or the sensory patterns given to it, might result in an agent that is completely unable to learn anything useful. We do not expect such an agent to do much worse than one without any shaping rewards at all. Another

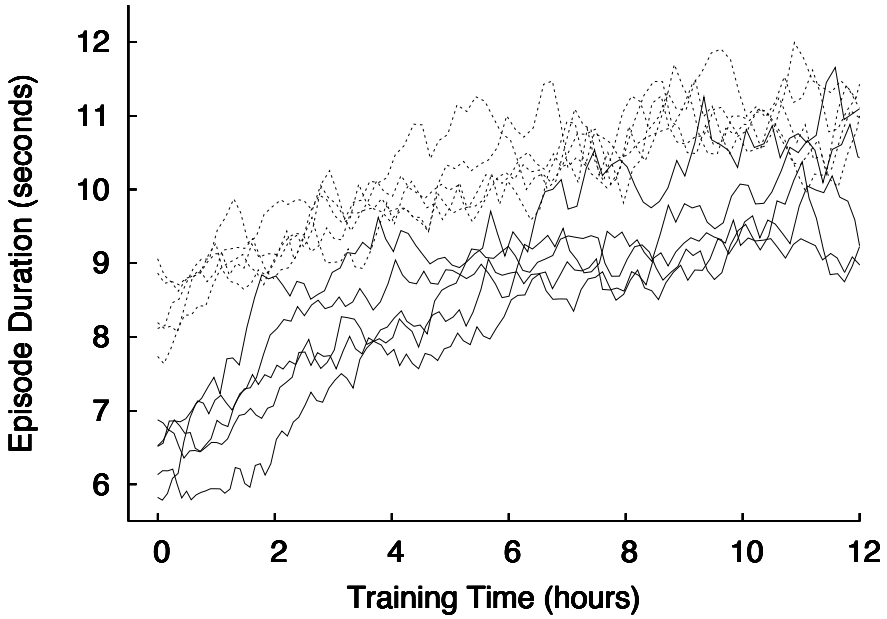


Figure 9: Sample learning curves for 4v3 Keepaway given no transfer (solid lines) or having experienced 5000 episodes of experience in 3v2 (dashed lines).

potential concern is the possibility that a maliciously chosen or unfortunate set of training tasks could result in an agent that performs worse than one with no training. Fortunately, such agents will still eventually be able to learn the correct value function (Ng et al., 1999).

All of the experiments reported in this paper use model-free learning algorithms. Given that an agent facing a sequence of tasks receives many example transitions between pairs of agent-space descriptors, it may prove efficient to instead learn an approximate transition model in agent-space and then use that model to obtain a shaping function via planning. However, learning a good transition model in such a scenario may prove difficult because the agent-space features are not Markov.

In standard classical search algorithms such as A*, a heuristic imposes an order in which nodes are considered during the search process. In reinforcement learning the state space is searched by the agent itself, but its initial value function (either directly or via a shaping function) acts to order the selection of unvisited nodes by the agent. Therefore, we argue that reinforcement learning agents using non-uniform initial value functions are using something very similar to a heuristic, and those that are able to learn their own portable shaping functions are in effect able to learn their own heuristics.

5. Skill Transfer

The previous section showed that we can effectively transfer knowledge about reward when a sequence of tasks is related and reward-linked, and that such knowledge can significantly improve

(a) Transfer results from 3v2 to 5v4 Keepaway.

# 3v2 Episodes	Ave. 3v2 Time	Ave 5v4 Time	Ave. Total Time	Std. Dev.
0	0.0000	7.4931	7.4931	1.5229
50	0.0765	6.3963	6.4728	1.0036
250	0.3919	5.6675	6.0594	0.7657
500	0.8870	5.9012	6.7882	1.1754
1000	1.8166	3.9817	**5.7983**	1.2522
2000	3.9908	3.9678	7.9586	1.8367
5000	14.7554	**3.9241**	18.6795	1.3228

(b) Transfer results from 4v3 to 5v4 Keepaway.

# 4v3 Episodes	Ave. 4v3 Time	Ave 5v4 Time	Ave. Total Time	Std. Dev.
0	0.0000	7.4931	7.4930	1.5229
50	0.0856	6.6268	6.7125	1.2162
250	0.4366	6.1323	**6.5689**	1.1198
500	0.8951	6.3227	7.2177	1.0084
1000	1.8671	6.0406	7.9077	1.0766
2000	4.0224	5.0520	9.0744	0.9760
5000	11.9047	**3.218**	15.1222	0.6966

Table 2: Results of transfer to 5v4 Keepaway.

# 3v2 Episodes	# 4v3 Episodes	Ave 5v4 Time	Ave. Total Time	Std. Dev.
500	500	6.1716	8.0703	1.1421
500	1000	5.6139	8.6229	0.9597
1000	500	**4.5395**	**7.3922**	0.6689
1000	1000	4.8648	8.8448	0.9517

Table 3: Results of transfer from both 3v2 Keepaway and 4v3 Keepaway to 5v4 Keepaway.

performance. We can apply the same framework to effect skill transfer by creating portable option policies. Most option learning methods work within the same state space as the problem the agent is solving at the time. Although this can lead to faster learning on later tasks in the same state space, learned options would be more useful if they could be reused in later tasks that are related but have distinct state spaces.

In this section we demonstrate empirically that an agent that learns portable options directly in agent-space can reuse those options in future related tasks to significantly improve performance. We also show that the best performance is obtained using portable options in conjunction with problem-specific options.

5.1 Options in Agent-Space

Following section 4.2, we consider an agent solving n problems $M_1, ..., M_n$ with state spaces $S_1, ..., S_n$, and action space A. Once again, we associate a four-tuple σ_i^j with the ith state in M_j:

$$\sigma_i^j = \langle s_i^j, d_i^j, r_i^j, v_i^j \rangle,$$

where s_i^j is the usual problem-space state descriptor (sufficient to distinguish this state from the others in S_j), d_i^j is the agent-space descriptor, r_i^j is the reward obtained at the state and v_i^j is the state's value (expected total reward for action starting from the state).

The agent is also either given, or learns, a set of higher-level options to reduce the time required to solve the task. Options defined using s_i^j are not portable between tasks because the form and meaning of s_i^j (as a problem-space descriptor) may change from one task to another. However, the form and meaning of d_i^j (as an agent-space descriptor) does not. Therefore we define agent-space option components as:

$$\begin{array}{lll} \pi_o: & (d_i^j, a) & \mapsto [0,1], \\ I_o: & d_i^j & \mapsto \{0,1\}, \\ \beta_o: & d_i^j & \mapsto [0,1]. \end{array}$$

Although the agent will be learning task and option policies in different spaces, both types of policies can be updated simultaneously as the agent receives both agent-space and problem-space descriptors at each state.

To support learning a portable shaping function, an agent space should contain some features that are correlated to return across tasks. To support successful skill policy learning, an agent space needs more: it must be suitable for directly learning control policies.

If that is the case, then why not perform task learning (in addition to option learning) in agent-space? There are two primary reasons why we might prefer to perform task learning in problem-space, even when given an agent-space suitable for control learning. The first is that agent-space may be very much larger than problem-space, making directly learning the entire task in agent-space inefficient or impractical. The second is that the agent-space may only be sufficient for learning control policies locally, rather than globally. In the next two sections we demonstrate portable skill learning on domains with each characteristic in turn.

5.2 The Lightworld Domain

The lightworld domain is a parameterizable class of discrete domains which share an agent-space that is much larger than any individual problem-space. In this section, we empirically examine whether learning portable skills can improve performance in such a domain.

An agent is placed in an environment consisting of a sequence of rooms, with each room containing a locked door, a lock, and possibly a key. In order to leave a room, the agent must unlock the door and step through it. In order to unlock the door, it must move up to the lock and press it, but if a key is present in the room the agent must be holding it to successfully unlock the door. The agent can obtain a key by moving on top of it and picking it up. The agent receives a reward of 1000 for leaving the door of the final room, and a step penalty of -1 for each action. Six actions are available: movement in each of the four grid directions, a pickup action and a press action. The environments are deterministic and unsuccessful actions (for example, moving into a wall) result in no change in state.

In order to specify an individual lightworld instance, we must specify the number of rooms, x and y sizes for each room, and the location of the room entrance, key (or lack therefore), lock and door in each. Thus, we may generate new lightworld instances by generating random values for each of these parameters.

We equip the agent with twelve light sensors, grouped into threes on each of its sides. The first sensor in each triplet detects red light, the second green and the third blue. Each sensor responds to light sources on its side of the agent, ranging from a reading of 1 when it is on top of the light source, to 0 when it is 20 squares away. Open doors emit a red light, keys on the floor (but not those held by the agent) emit a green light, and locks emit a blue light. Figure 10 shows an example.

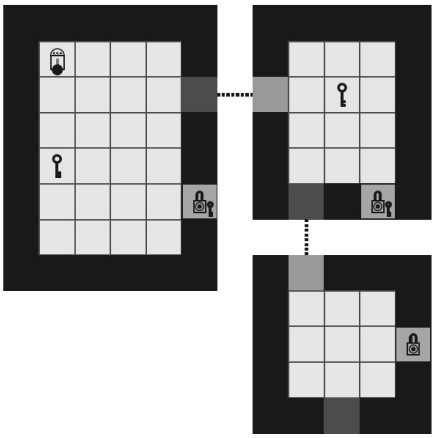


Figure 10: A small example lightworld.

Five pieces of data form a problem-space descriptor for any lightworld instance: the current room number, the x and y coordinates of the agent in that room, whether or not the agent has the key, and whether or not the door is open. We use the light sensor readings as an agent-space because their semantics are consistent across lightworld instances. In this case the agent-space (with 12 continuous variables) has much higher dimension than any individual problem-space, and it is impractical to perform task learning in it directly, even though the problem might in principle be solvable that way.

5.2.1 TYPES OF AGENT

We used five types of reinforcement learning agents: agents without options, agents with problem-space options, agents with perfect problem-space options, agents with agent-space options, and agents with both option types.

The agents without options used Sarsa(λ) with ε-greedy action selection ($\alpha = 0.1$, $\gamma = 0.99$, $\lambda = 0.9$, $\varepsilon = 0.01$) to learn a solution policy in problem-space, with each state-action pair assigned an initial value of 500.

Agents with problem-space options had an (initially unlearned) option for each pre-specified salient event (picking up each key, unlocking each lock, and walking through each door). Options were learned in problem-space and used the same parameters as the agent without options, but used

off-policy trace-based tree-backup updates (Precup et al., 2000) for intra-option learning. We used an option termination reward of 1 for successful completion, and a discount factor of 0.99 per action. Options could be executed only in the room in which they were defined and only in states where their value function exceeded a minimum threshold (0.0001). Because these options were learned in problem-space, they were useful but needed to be relearned for each individual lightworld.

Agents with perfect problem-space options were given options with pre-learned policies for each salient event, though they still performed option updates and were otherwise identical to the standard agent with options. They represent the ideal case of agents with that can perform perfect transfer, arriving in a new task with fully learned options.

Agents with agent-space options still learned their solution policies in problem-space but learned their option policies in agent-space. Each agent employed three options: one for picking up a key, one for going through an open door and one for unlocking a door, with each one's policy a function of the twelve light sensors. Since the sensor outputs are continuous we employed linear function approximation for each option's value function, performing updates using gradient descent ($\alpha = 0.01$) and off-policy trace-based tree-backup updates. We used an option termination reward of 1, a step penalty of 0.05 and a discount factor of 0.99. An option could be taken at a particular state when its value function there exceeded a minimum threshold of 0.1. Because these options were learned in agent-space, they could be transferred between lightworld instances.

Finally, agents with both types of options were included to represent agents that learn both general portable and specific non-portable skills simultaneously.

Note that all agents used discrete problem-space value functions to solve the underlying task instance, because their agent-space descriptors are only Markov in lightworlds with a single room, which were not present in our experiments.

5.2.2 EXPERIMENTAL STRUCTURE

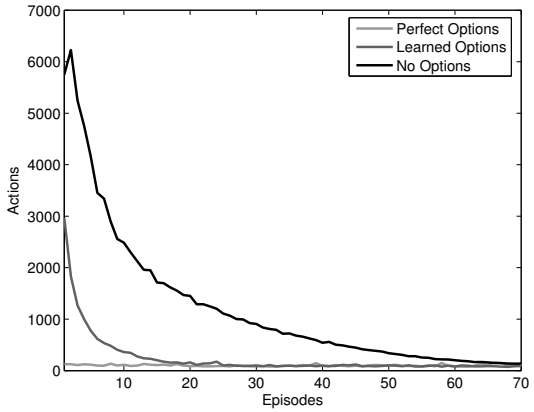
We generated 100 random lightworlds, each consisting of 2-5 rooms with width and height of between 5 and 15 cells. A door and lock were randomly placed on each room boundary, and $\frac{1}{3}$ of rooms included a randomly placed key. This resulted in state space with between 600 and approximately $20,000$ state-action pairs ($4,900$ on average). We evaluated each problem-space option agent type on 1000 lightworlds (10 samples of each generated lightworld).

To evaluate the performance of agent-space options as the agents gained more experience, we similarly obtained 1000 lightworld samples and test tasks, but for each test task we ran the agents once without training and then with between 1 and 10 training experiences. Each training experience for a test lightworld task consisted of 100 episodes in a training lightworld randomly selected from the remaining 99. Although the agents updated their options during evaluation in the test lightworld, these updates were discarded before the next training experience so the agent-space options never received prior training in the test lightworld.

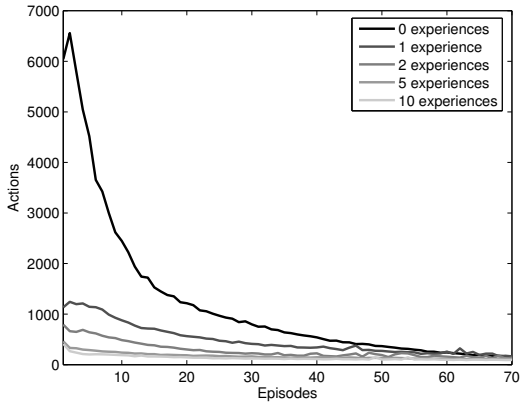
5.2.3 RESULTS

Figure 11(a) shows average learning curves for agents employing problem-space options, and Figure 11(b) shows the same for agents employing agent-space options. The first time an agent-space option agent encounters a lightworld, it performs similarly to an agent without options (as evidenced by the two topmost learning curves in each figure), but its performance rapidly improves with experience in other lightworlds. After experiencing a single training lightworld, the agent starts with

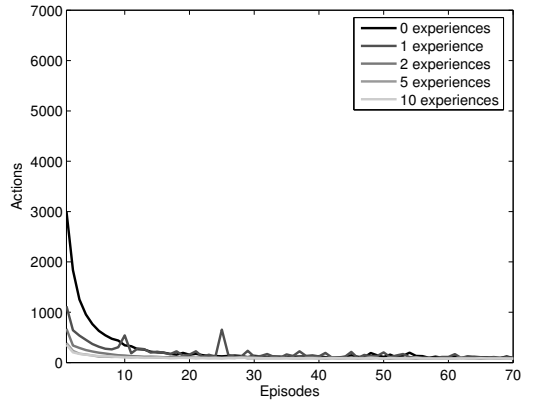
better performance than an agent using problem-space options alone, until by 5 experiences its learning curve is similar to that of an agent with perfect problem-space options (compare the learning curves in Figure 11(b) with the bottom-most learning curve of Figure 11(a)), even though its options are never trained in the same lightworld in which it is tested. The comparison between Figures 11(a) and 11(b) shows that agent-space options can be successfully transferred between lightworld instances.



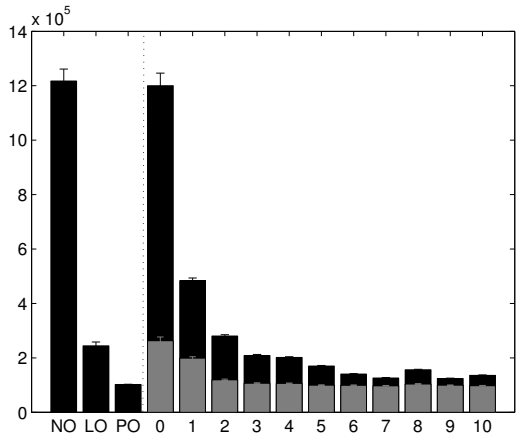
(a) Learning curves for agents with problem-space options.



(b) Learning curves for agents with agent-space options, with varying numbers of training experiences.



(c) Learning curves for agents with agent-space *and* problem-space options, with varying numbers of training experiences.



(d) Total steps over 70 episodes for agents with no options (NO), learned problem-space options (LO), perfect options (PO), agent-space options with 0-10 training experiences (dark bars), and both option types with 0-10 training experiences (light bars).

Figure 11: Results for the Lightworld Domain.

Figure 11(c) shows average learning curves for agents employing *both* types of options.[4] The first time such agents encounter a lightworld, they perform as well as agents using problem-space

4. In 8 of the more than 200,000 episodes used when testing agents with both types of options, an agent-space value function approximator diverged, and we restarted the episode. Although this is a known problem with the backup

options (compare with the second highest curve in Figure 11(a)), and thereafter rapidly improve their performance (performing better than agents using only agent-space options) and again by 5 experiences performed nearly as well as agents with perfect options. This improvement can be explained by two factors. First, the agent-space is much larger than any individual problem-space, so problem-space options are easier to learn from scratch than agent-space options. This explains why agents using only agent-space options and no training experiences perform more like agents without options than like agents with problem-space options. Second, options learned in our problem-space can represent exact solutions to specific subgoals, whereas options learned in our agent-space are general and must be approximated, and are therefore likely to be slightly less efficient for any specific subgoal. This explains why agents using both types of options perform better in the long run than agents using only agent-space options.

Figure 11(d) shows the mean total number of steps required over 70 episodes for agents using no options, problem-space options, perfect options, agent-space options, and both option types. Experience in training environments rapidly drops the number of total steps required to nearly as low as the number required for an agent with perfect options. It also clearly shows that agents using both types of options do consistently better than those using agent-space options alone. We note that the error bars in Figure 11(d) are small and decrease with experience, indicating consistent transfer.

In summary, these results show that learning using portable options can greatly improve performance over learning using problem-specific options. Given enough experience, learned portable options can perform similarly to perfect pre-learned problem-specific options, even when the agent-space is much harder to learn in than any individual problem-space. However, the best learning strategy is to learn using both problem-specific options and portable options.

5.3 The Conveyor Belt Domain

In the previous section we showed that an agent can use experience in related tasks to learn portable options, and that those options can improve performance in later tasks, when the agent has a high-dimensional agent-space. In this section we consider a task where the agent-space is not high-dimensional, but is only sufficient for local control.

In the conveyor belt domain, a conveyor belt system must move a set of objects from a row of feeders to a row of bins. There are two types of objects (triangles and squares), and each bin starts with a capacity for each type. The objects are issued one at a time from a feeder and must be directed to a bin. Dropping an object into a bin with a positive capacity for its type decrements that capacity.

Each feeder is directly connected to its opposing bin through a conveyor belt, which is connected to the belts above and below it at a pair of fixed points along its length. The system may either run the conveyor belt (which moves the current object one step along the belt) or try to move it up or down (which only moves the object if it is at a connection point). Each action results in a reward of -1, except where it causes an object to be dropped into a bin with spare capacity, in which case it results in a reward of 100. Dropping an object into a bin with zero capacity for that type results in the standard reward of -1.

method we used (Precup et al., 2000), it did not occur during the same number of samples obtained for agents with agent-space options only.

To specify an instance of the conveyor belt domain, we must specify the number of objects present, belts present, bin capacities, belt length, and where each adjacent pair of belts are connected. A small example conveyor belt system is shown in Figure 12.

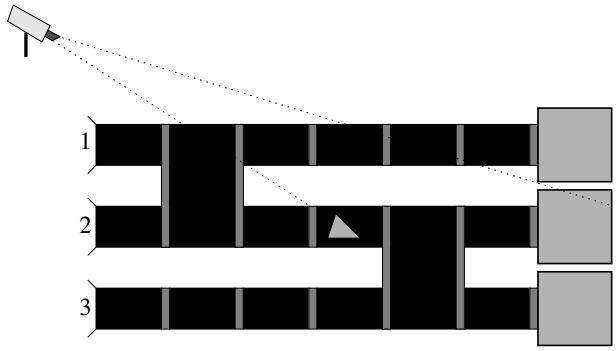


Figure 12: A small example conveyor belt problem.

Each system has a camera that tracks the current object and returns values indicating the distance (up to 15 units) to the bin and each connector along the current belt. Because the space generated by the camera is present in every conveyor-belt problem and retains the same semantics, it is an agent-space, and because it is discrete and relatively small (13,500 states), we can learn policies in it without function approximation. However, because it is non-Markov (due to its limited range and inability to distinguish between belts), it cannot be used as a problem-space.

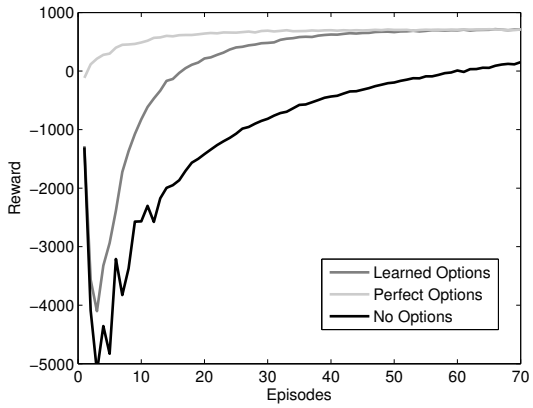
A problem-space descriptor for a conveyor belt instance consists of three numbers: the current object number, the belt it is on, and how far along that belt it lies (technically we should include the current capacity of each bin, but we can omit this and still obtain good policies). We generated 100 random instances with 30 objects and 20-30 belts (each of length 30-50) with randomly-selected interconnections, resulting in problem-spaces of 18,000-45,000 states.

We ran experiments where the agents learned three options: one to move the current object to the bin at the end of the belt it is currently on, one for moving it to the belt above it, and one for moving it to the belt below it. We used the same agent types and experimental structure as before, except that the agent-space options did not use function approximation.

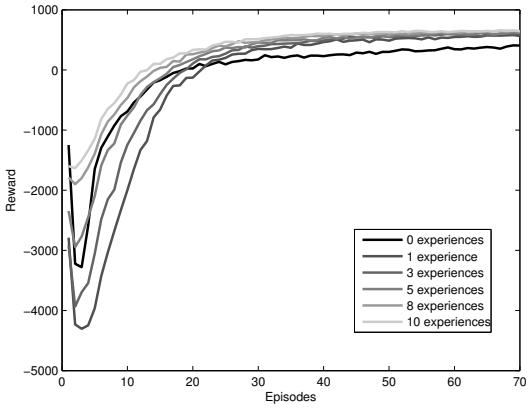
5.3.1 RESULTS

Figures 13(a), 13(b) and 13(c) show learning curves for agents employing no options, problem-space options and perfect options; agents employing agent-space options; and agents employing both types of options, respectively.

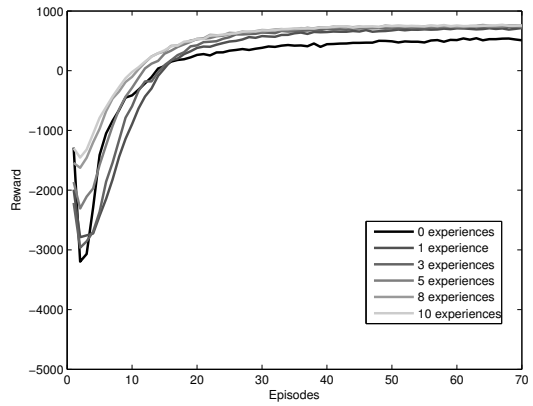
Figure 13(b) shows that the agents with agent-space options and no prior experience initially improve quickly but eventually obtain lower quality solutions than agents with problem-space options (Figure 13(a)). One or two training experiences result in roughly the same curve as agents using problem-space options, but by 5 training experiences the agent-space options are a significant improvement (although due to their limited range they are never as good as perfect options). This initial dip relatively to agents with no prior experience is probably due to the limited range of the agent-space options (due to the limited range of the camera) and the fact that they are only locally Markov, even for their own subgoals.



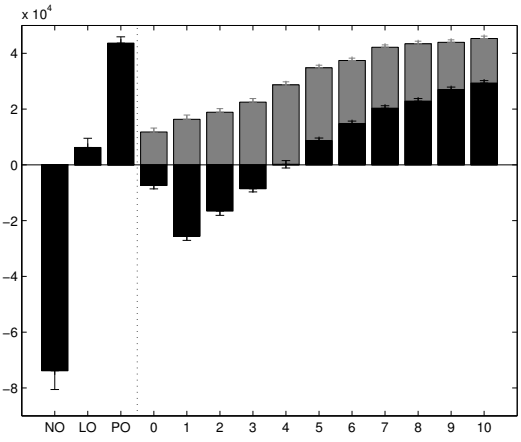
(a) Learning curves for agents with problem-space options.



(b) Learning curves for agents with agent-space options, with varying numbers of training experiences.



(c) Learning curves for agents with both types of options, with varying numbers of training experiences.



(d) Total reward over 70 episodes for agents with no options (NO), learned problem-space options (LO), perfect options (PO), agent-space options with 0-10 training experiences (dark bars), and both option types with 0-10 training experiences (light bars).

Figure 13: Results for the Conveyor Belt Domain.

Figure 13(c) shows that agents with both option types do not experience this initial dip relative to agents with no prior experience and outperform problem-space options immediately, most likely because the agent-space options are able to generalise across belts. Figure 13(d) shows the mean total reward for each type of agent. Agents using agent-space options eventually outperform agents using problem-space options only, even though the agent-space options have a much more limited range; agents using both types of options consistently outperform agents with either option type and eventually approach the performance of agents using pre-learned problem-space options.

In summary, these results demonstrate that when an agent-space is only locally Markov, learning portable options can still result in a significant performance improvement over learning using problem-specific options, but that even with a great deal of experience will not reach the perfor-

mance of perfect pre-learned problem-specific options. Once again, the best approach is to learn using both problem-specific and agent-space options simultaneously.

5.4 Summary

Our results show that options learned in agent-space can be successfully transferred between related tasks, and that this significantly improves performance in sequences of tasks where the agent space cannot be used for learning directly. Our results suggest that when the agent space is large but can support global policies, experience in related tasks can eventually result in options that perform as well as perfect problem-specific options. When the agent space is only locally Markov, learned portable options will improve performance but are unlikely to reach the performance of perfect problem-specific options due to their limited range.

We expect that, in general, learning an option in agent-space will often actually be harder than solving an individual problem-space instance, as was the case in our experiments. In such situations, learning both problem-specific and agent space options simultaneously will likely obtain better performance than either individually. Since intra-option learning methods allow for the update of several options from the same experiences, it may be better in general to simultaneously learn both general portable skills and specific, exact but non-portable skills, and allow them to bootstrap each other.

6. Related Work

Although the majority of research in transfer assumes that the source and target problems have the same state space, some existing research does not make that assumption.

Wilson et al. (2007) consider the case where an agent faces a sequence of environments, each generated by one of a set of environment classes. Each environment class is modeled as a distribution of values of some observed signal given a feature vector, and since the number of classes is unknown, the agent must learn an infinite mixture model of classes. When faced with a new environment, the agent determines which of its existing models it best matches or whether it instead corresponds to a novel class. A model-based planning algorithm is then used to solve the new task. This work explicitly considers environment sequences that do not have the same state space, and thus defines the distributions of each environment class over the output of a function f that generates a feature vector for each state in each environment. Since that feature vector retains its semantics across all of the tasks, it is exactly an agent-space descriptor as defined here. Thus, this work can be seen as using agent-space to learn a set of environment models.

Banerjee and Stone (2007) consider transfer learning for the case of General Game Playing, where knowledge gained from playing one game (e.g., Tic-Tac-Toe) is exploited to improve performance in another (e.g., Connect-4). Here, transfer is affected through the game tree: the authors define generic game-tree features that apply across all games and then use their Q-values to initialize the values of novel states with matching features when playing a subsequent game. This is a very similar mechanism to a portable shaping function, including the use of features—in this case derived from the game tree—that are common across all tasks.

Taylor et al. (2007) use a hand-coded transfer function to seed one task's value function with learned values from another similar task with a potentially different state space. This requires a mapping to be constructed between the weights of the function approximators of each pair of tasks

between which transfer might occur.[5] Our method offers two advantages over this. First, we effectively require the construction of a mapping from each task to an agent-space, so the number of mappings scales linearly with the number of tasks, rather than quadratically. Second, through the use of a shaping function, those mappings can be constructed between state descriptors, rather than between function approximation terms. This allows us to treat the function approximator employed for each task as a black box rather than requiring detailed knowledge of its construction, and it allows us to transfer between potentially very different function approximators where a direct mapping might be difficult to obtain. On the other hand, if performance is critical, then constructing a specialized task-to-task mapping may result in better performance than a more generic agent-space mapping; the results in Taylor et al. (2007) seem slightly better than those given in Section 4.4.2, although a direct comparison is not possible since the benchmarks used (expressing the underlying learning performance) differ (presumably due to implementation differences), even though we used the same parameters.

Another related line of research focuses on effecting representation transfer, where basis functions are learned in one task and applied in another. Representation transfer has so far focused primarily on task sequences where reward function or dynamics differ but the state space remains the same (Ferguson and Mahadevan, 2006; Ferrante et al., 2008). If the state spaces differ significantly, manifold alignment or scaling methods may be employed to transform basis functions from one state space to another (Ferguson and Mahadevan, 2008); however, such transformations require prior knowledge of the topology of the two state spaces to either achieve scaling or to obtain a good alignment.

Lazaric et al. (2008) introduced sample transfer, where sample transitions from a source task may be used as additional data to improve performance in a new task. Transition samples from a set of source tasks are stored, and then used along with a small set of sample transitions in a new task to compute a similarity measure between the new task and the source tasks. The transferred transitions are then sampled according to the similarity measure and added to the new task samples, resulting in a performance boost for batch-learning methods. Reusing such samples requires their state descriptors to (at least) be the same size, although if the reused descriptors were defined in an agent-space, then such a method may be useful for more efficiently learning portable shaping functions.

Konidaris and Hayes (2004) describe a similar method to ours that uses training tasks to learn associations between reward and strong signals at reward states, resulting in a significant improvement in the total reward obtained by a simulated robot learning to find a puck in a novel maze. The research presented in this paper employs a more general mechanism where the agent learns a heuristic from all visited states.

Zhang and Dietterich (1995) use common features to transfer learned value functions across a class of job-shop scheduling problems. The value functions (represented as neural networks) were learned using TD(λ) over a set of features constructed to be common to the entire problem class. Value functions trained using small instances of scheduling problems were then used to obtain solutions to larger problems. This research is a case where an agent-space was sufficient to

5. Construction has been primarily accomplished by hand, but we briefly discuss recent work aimed at automating it in Section 7.1.

represent a solution to each individual problem and the need for a problem-specific state space was avoided.[6]

The X-STAGE algorithm (Boyan and Moore, 2000) uses features common across a class of tasks to transfer learned evaluation functions that predict the performance of a local search algorithm applied to an optimization task. The evaluation functions—which are similar to value functions in that they predict the outcome of the execution of a policy, in this case a search algorithm—serve to identify the most promising restart points for local search. The X-STAGE algorithm learns a distinct evaluation function for each source task and then obtains a "vote" for the next action in the target task from each source evaluation function. Interestingly, while this method of transfer results in an initial performance boost, it eventually obtains solutions inferior to those obtained by learning a problem-specific evaluation function; our use of shaping avoids this dilemma, because it naturally incorporates experience from the current task into the agent's value function and thus avoids permanent bias arising from the use of transferred knowledge.

All of the option creation methods given in Section 2.2 learn options in the same state space in which the agent is performing reinforcement learning, and thus the options can only be reused for the same problem or for a new problem in the same space. The available state abstraction methods (Jonsson and Barto, 2001; Hengst, 2002) only allow for the automatic selection of a subset of this space for option learning, or they require an explicit transformation from one space to another (Ravindran and Barto, 2003a).

There has been some research focusing on extracting options by exploiting commonalities in collections of policies (Thrun and Schwartz, 1995; Bernstein, 1999; Perkins and Precup, 1999; Pickett and Barto, 2002) or analysing the relationships between variables given sample trajectories (Mehta et al., 2008), but in each case the options are learned over a single state space. In contrast, we leave the method used for creating the options unspecified—any option creation method may be used—but create them in a portable space.

Fernández and Veloso (2006) describe a method called Policy Reuse, where an agent given a library of existing policies determines which of them is closest to a new problem it faces, and then incorporates that policy into the agent's exploration strategy. The resulting distance metric is also used to build a library of core policies that can be reused for later tasks in the same state space. Although this method has very attractive attributes (particularly when applied in a hierarchical setting), it is limited to task sequences where only the reward function changes.

Torrey et al. (2006) show that policy fragments learned in a symbolic form using inductive logic programming (ILP) can be transferred to new tasks as constraints on the new value-function. This results in a substantial performance improvement. However, a user must provide a mapping from state variables in the first task to the second, and the use of an ILP implementation introduces significant complexity and overhead.

Croonenborghs et al. (2007) learns relational options and shows that they can be transferred to different state spaces provided the same symbols are still present. This approach is similar to ours, in that we could consider the symbols shared between the tasks to be an agent-space.

6. The agent-space in this case did introduce aliasing, which occasionally caused policies with loops. This was avoided using a loop-detection algorithm.

7. Discussion

The work in the preceding sections has shown that both knowledge and skill transfer can be effected across a sequence of tasks through the use of features common to all tasks in the sequence. Our results have shown significant improvements over learning from scratch, and the framework offers some insight into which problems are amenable to transfer.

However, our framework requires the identification of a suitable agent-space to facilitate transfer, but it does not specify how that space is identified, which creates a design problem similar to that of standard state space design. Researchers in the reinforcement learning community have so far developed significant expertise at designing problem-spaces, but not agent-spaces. Nevertheless, the example domains in this paper offer several examples of related tasks with different types of common feature sets—deictic sensors (the Rod positioning task and the Lightworld), a maximum set (Keepaway), and local sensing (the Conveyor Belt domain)—and we have pointed out the use of similar feature sets in existing work (Zhang and Dietterich, 1995; Boyan and Moore, 2000; Wilson et al., 2007; Snel and Whiteson, 2010). Taken together, these examples suggest that transfer via common features may find wide application.

Additionally, for option learning, an agent-space descriptor should ideally be Markov within the set of states that the option is defined over. The agent-space descriptor form will therefore affect both what options can be learned and their range. In this respect, designing agent-spaces for learning options requires more care than for learning shaping functions.

An important assumption made in our option transfer work is that all tasks have the same set of available actions, even though they have different state spaces. If this is not the case, then learning portable options directly is only possible if the action spaces share a common subset or if we can find a mapping between action spaces. If no such mapping is given, we may be able to construct one from experience using a homomorphism (Ravindran and Barto, 2003b).

When learning portable shaping functions, if the action space differs across tasks then we can simply learn shaping functions defined over states only (as potential-based shaping functions were originally defined by Ng et al., 1999) rather than defining them over state-action pairs. Although we expect that learning using portable state-only shaping functions will not perform as well as learning using portable state-action shaping functions, we nevertheless expect that they will result in substantial performance gains for reward-linked tasks.

The idea of an agent-centric representation is closely related to the notion of deictic or egocentric representations (Agre and Chapman, 1987), where objects are represented from the point of view of the agent rather than in some global frame of reference. We expect that for most problems, especially in robotics, agent-space representations will be egocentric, except in manipulation tasks, where they will likely be object-centric. In problems involving spatial maps, we expect that the difference between problem-space and agent-space will be closely related to the difference between allocentric and egocentric representations of space (Guazzelli et al., 1998)—the utility of such spaces for transfer has been demonstrated by Frommberger (2008).

7.1 Identifying Agent-Spaces

In this work we have assumed that an agent-space is given. However, this may not always be the case; if it is not, then we are faced with the problem of attempting to automatically identify collections of features that retain their semantics across tasks.

This problem may arise in several settings. In the simplest setting, given a pair of corresponding feature sets for two problems, we must determine whether the two feature sets are an agent-space. To do this, we may build approximate transition models for each feature set, and then compare them.

In an intermediate setting, we might be given two sets of corresponding features and asked to identify which subsets of these features form an agent-space. Snel and Whiteson (2010) report very promising results on this problem using a formalization of how task-informative a feature is (and thus how likely it is to be in problem-space) against how domain-informative it is (and thus how likely it is to be in agent-space).

The problem becomes much harder when we are given an arbitrary number of features for each task, and we are required to both identify correspondences between features and determine which subset of features form an agent-space. Taylor et al. (2008) address a similar problem: constructing mappings between two sets of state variables for a pair of given tasks. They propose an algorithm which generates all possible mappings from the first task to the second, then learns a transition model from the first and compares its predictions (using each candidate mapping) to sample data from the second; finally the algorithm selects the mapping with the lowest transition error. This method can be adapted to our setting by selecting a reference task (most likely the first task the agent sees) and then building mappings from each new task back to it. The subset of variables in the reference task that appear in all mappings constitute an agent-space.

Taylor et al. (2008) claim that their algorithm is data-efficient because the same sample transitions can be used for comparing every possible mapping, even though the algorithm's time complexity is exponential in the size of the number of variables in the two tasks. In our setting, once the first mapping (from the reference task to some other task) has been constructed, we may remove the reference variables absent from the mapping from later consideration, which could lead to significant efficiency gains when constructing later mappings. In addition, such a method (mapping to a reference task) would require only $n-1$ mappings to be constructed for arbitrary transfer between pairs of tasks drawn from a sequence of n tasks, whereas a direct mapping methodology requires $O(n^2)$ mappings to be constructed.

In the most difficult setting, we might be given no features at all, and asked to construct an agent-space. This can be considered a problem of discovering latent variables that describe aspects of the state space which can be used for transfer. We expect that this will be a challenging but fruitful avenue of future work.

7.2 Identifying Reward-Linked Tasks

An important distinction raised by this work is the difference between related tasks and tasks that are both related and reward-linked. Tasks that are related (in that they share an agent-space) but are not reward-linked do not allow us to transfer knowledge about the value function, since they do not necessarily have similar reward functions.

This raises an important question for future work: given a solved task T_s and a new related task T_n, how can we determine whether they are reward-linked? More broadly, given a set of previously learned related tasks that are not reward-linked, which one should we use as the source of a portable shaping function for a new related task?

Answering these questions relies upon us finding some method of comparison for the two task reward functions, R_n and R_s. Since one of the important motivations behind learning portable shaping functions is boosting initial task performance, we would prefer to perform this comparison

without requiring much experience. If we are given R_n in some appropriate functional form, then we could obtain its value at sample state-action pairs in T_s and compare the results with the experienced values of R_s. If the method used to compare the two reward functions returns a distance metric, then the agent could use it to cluster portable shaping functions and build libraries of them, drawing on an appropriate one for each new task it encounters.

However, if we are not given R_n, then we must sample R_n with experience. It is easy to construct an adversarial argument showing that an agent with experience only in T_s cannot determine whether T_n and T_s are reward-linked without at the very least one full episode's worth of experience in T_n.

However, we do not believe the complete absence of prior information about a task is representative of applied reinforcement learning settings where the agent must solve multiple tasks sequentially. In most cases, the agent has access to some extra information about a new task before it attempts to solve it. We can formalize this by attaching a descriptor to each task; then the extent to which an agent can recognize a task "type" depends on how much information is contained in its descriptor. If the relationship between task descriptor and task "type" is not known in advance, it can be learned over time using training examples obtained by comparing reward functions after experience.

8. Summary and Conclusions

We have presented a framework for transfer in reinforcement learning based on the idea that related tasks share common features and that transfer can take place through functions defined over those related features. The framework attempts to capture the notion of tasks that are related but distinct, and it provides some insight into when transfer can be usefully applied to a problem sequence and when it cannot.

Most prior work on transfer relies on mappings between pairs of tasks, and therefore implicitly defines transfer as a relationship between problems. This work provides a contrasting viewpoint by relying on a stronger notion of an agent: that there is something common across a series of tasks faced by the same agent, and that that commonality derives from features shared because they are a property of an agent.

We have empirically demonstrated that this framework can be successfully applied to significantly improve transfer using both knowledge transfer and skill transfer. It provides a practical approach to building agents that are capable of improving their own problem-solving capabilities through experience over multiple problems.

Acknowledgments

We would like to thank Ron Parr and our three anonymous reviewers for their tireless efforts to improve this work. We also thank Gillian Hayes, Colin Barringer, Sarah Osentoski, Özgür Şimşek, Michael Littman, Aron Culotta, Ashvin Shah, Chris Vigorito, Kim Ferguson, Andrew Stout, Khashayar Rohanimanesh, Pippin Wolfe and Gene Novark for their comments and assistance. Andrew Barto and George Konidaris were supported in part by the National Science Foundation under Grant No. CCF-0432143, and Andrew Barto was supported in part by a subcontract from Rutgers University, Computer Science Department, under award number HR0011-04-1-0050 from DARPA. George Konidaris was supported in part by the AFOSR under grant AOARD-104135

and the Singapore Ministry of Education under a grant to the Singapore-MIT International Design Center. Any opinions, findings and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation, DARPA, the AFOSR, or the Singapore Ministry of Education.

References

P.E. Agre and D. Chapman. Pengi: An implementation of a theory of activity. In *Proceedings of the Sixth National Conference on Artificial Intelligence*, pages 268–272, 1987.

B. Banerjee and P. Stone. General game learning using knowledge transfer. In *Proceedings of the 20th International Joint Conference on Artificial Intelligence*, pages 672–677, 2007.

A.G. Barto and S. Mahadevan. Recent advances in hierarchical reinforcement learning. *Discrete Event Dynamic Systems*, 13:41–77, 2003. Special Issue on Reinforcement Learning.

D.S. Bernstein. Reusing old policies to accelerate learning on new MDPs. Technical Report UM-CS-1999-026, Department of Computer Science, University of Massachusetts at Amherst, April 1999.

J. Boyan and A.W. Moore. Learning evaluation functions to improve optimization by local search. *Journal of Machine Learning Research*, 1:77–112, 2000.

T. Croonenborghs, K. Driessens, and M. Bruynooghe. Learning relational options for inductive transfer in relational reinforcement learning. In *Proceedings of the Seventeenth International Conference on Inductive Logic Programming*, pages 88–97, 2007.

T.G. Dietterich. Hierarchical reinforcement learning with the MAXQ value function decomposition. *Journal of Artificial Intelligence Research*, 13:227–303, 2000.

B.L. Digney. Learning hierarchical control structures for multiple tasks and changing environments. In R. Pfeifer, B. Blumberg, J. Meyer, and S.W. Wilson, editors, *From Animals to Animats 5: Proceedings of the Fifth International Conference on Simulation of Adaptive Behavior*, Zurich, Switzerland, August 1998. MIT Press.

M. Dorigo and M. Colombetti. *Robot Shaping: An Experiment in Behavior Engineering*. MIT Press/Bradford Books, 1998.

K. Ferguson and S. Mahadevan. Proto-transfer learning in Markov Decision Processes using spectral methods. In *Proceedings of the ICML Workshop on Structural Knowledge Transfer for Machine Learning*, Pittsburgh, June 2006.

K. Ferguson and S. Mahadevan. Proto-transfer learning in Markov Decision Processes using spectral methods. Technical Report TR-08-23, University of Massachusetts Amherst, 2008.

F. Fernández and M. Veloso. Probabilistic policy reuse in a reinforcement learning agent. In *Proceedings of the 5th International Joint Conference on Autonomous Agents and Multiagent Systems*, pages 720–727, 2006.

E. Ferrante, A. Lazaric, and M. Restelli. Transfer of task representation in reinforcement learning using policy-based proto-value functions (short paper). In *Proceedings of the Seventh International Conference on Autonomous Agents and Multiagent Systems*, pages 1329–1332, 2008.

L. Frommberger. Learning to behave in space: A qualitative spatial representation for robot navigation with reinforcement learning. *International Journal on Artificial Intelligence Tools*, 17(3): 465–482, 2008.

A. Guazzelli, F.J. Corbacho, M. Bota, and M.A. Arbib. Affordances, motivations, and the world graph theory. *Adaptive Behavior*, 6(3/4):433–471, 1998.

B. Hengst. Discovering hierarchy in reinforcement learning with HEXQ. In *Proceedings of the Nineteenth International Conference on Machine Learning*, pages 243–250, 2002.

A. Jonsson and A.G. Barto. Automated state abstraction for options using the U-Tree algorithm. In *Advances in Neural Information Processing Systems 13*, pages 1054–1060, 2001.

A. Jonsson and A.G. Barto. A causal approach to hierarchical decomposition of factored MDPs. In *Proceedings of the Twenty Second International Conference on Machine Learning*, 2005.

G.D. Konidaris and G.M. Hayes. Estimating future reward in reinforcement learning animats using associative learning. In *From Animals to Animats 8: Proceedings of the 8th International Conference on the Simulation of Adaptive Behavior*, pages 297–304, July 2004.

A. Laud and G. DeJong. The influence of reward on the speed of reinforcement learning: an analysis of shaping. In *Proceedings of the Twentieth International Conference on Machine Learning*, pages 440–447, 2003.

A. Lazaric, M. Restelli, and A. Bonarini. Transfer of samples in batch reinforcement learning. In *Proceedings of the Twenty-Fifth International Conference on Machine Learning*, pages 544–551, 2008.

S. Mannor, I. Menache, A. Hoze, and U. Klein. Dynamic abstraction in reinforcement learning via clustering. In *Proceedings of the Twenty First International Conference on Machine Learning*, pages 560–567, 2004.

M.J. Matarić. Reinforcement learning in the multi-robot domain. *Autonomous Robots*, 4(1):73–83, 1997.

A. McGovern and A.G. Barto. Automatic discovery of subgoals in reinforcement learning using diverse density. In *Proceedings of the Eighteenth International Conference on Machine Learning*, pages 361–368, 2001.

N. Mehta, S. Ray, P. Tadepalli, and T. Dietterich. Automatic discovery and transfer of MAXQ hierarchies. In *Proceedings of the Twenty Fifth International Conference on Machine Learning*, 2008.

A.W. Moore and C.G. Atkeson. Prioritized sweeping: Reinforcement learning with less data and less time. *Machine Learning*, 13(1):103–130, 1993.

A.Y. Ng, D. Harada, and S. Russell. Policy invariance under reward transformations: theory and application to reward shaping. In *Proceedings of the 16th International Conference on Machine Learning*, pages 278–287, 1999.

R. Parr and S. Russell. Reinforcement learning with hierarchies of machines. In *Advances in Neural Information Processing Systems 10*, pages 1043–1049, 1997.

T.J. Perkins and D. Precup. Using options for knowledge transfer in reinforcement learning. Technical Report UM-CS-1999-034, Department of Computer Science, University of Massachusetts, Amherst, 1999.

M. Pickett and A.G. Barto. Policyblocks: An algorithm for creating useful macro-actions in reinforcement learning. In *Proceedings of the Nineteenth International Conference of Machine Learning*, pages 506–513, 2002.

D. Precup, R.S. Sutton, and S. Singh. Eligibility traces for off-policy policy evaluation. In *Proceedings of the Seventeenth International Conference on Machine Learning*, pages 759–766, 2000.

M.L. Puterman. *Markov Decision Processes*. Wiley, 1994.

J. Randløv and P. Alstrøm. Learning to drive a bicycle using reinforcement learning and shaping. In *Proceedings of the 15th International Conference on Machine Learning*, pages 463–471, 1998.

B. Ravindran and A.G. Barto. Relativized options: Choosing the right transformation. In *Proceedings of the Twentieth International Conference on Machine Learning*, pages 608–615, 2003a.

B. Ravindran and A.G. Barto. SMDP homomorphisms: An algebraic approach to abstraction in semi markov decision processes. In *Proceedings of the Eighteenth International Joint Conference on Artificial Intelligence*, pages 1011–1016, 2003b.

O. Selfridge, R. S. Sutton, and A. G. Barto. Training and tracking in robotics. In *Proceedings of the Ninth International Joint Conference on Artificial Intelligence*, pages 670–672, 1985.

Ö. Şimşek and A. G. Barto. Using relative novelty to identify useful temporal abstractions in reinforcement learning. In *Proceedings of the Twenty-First International Conference on Machine Learning*, pages 751–758, 2004.

Ö. Şimşek, A. P. Wolfe, and A. G. Barto. Identifying useful subgoals in reinforcement learning by local graph partitioning. In *Proceedings of the Twenty-Second International Conference on Machine Learning*, 2005.

S. Singh, A.G. Barto, and N. Chentanez. Intrinsically motivated reinforcement learning. In *Proceedings of the 18th Annual Conference on Neural Information Processing Systems*, 2004.

B. F. Skinner. *The Behavior of Organisms: An Experimental Analysis*. Appleton-Century-Crofts, New York, 1938.

M. Snel and S. Whiteson. Multi-task evolutionary shaping without pre-specified representations. In *Proceedings of the Genetic and Evolutionary Computation Conference*, pages 1031–1038, 2010.

P. Stone, R.S. Sutton, and G. Kuhlmann. Reinforcement learning for robocup soccer keepaway. *Adaptive Behavior*, 13(3):165–188, 2005.

R.S. Sutton and A.G. Barto. *Reinforcement Learning: An Introduction*. MIT Press, Cambridge, MA, 1998.

R.S. Sutton, D. Precup, and S.P. Singh. Intra-option learning about temporally abstract actions. In *Proceedings of the Fifteenth International Conference on Machine Learning*, pages 556–564, 1998.

R.S. Sutton, D. Precup, and S.P. Singh. Between MDPs and semi-MDPs: A framework for temporal abstraction in reinforcement learning. *Artificial Intelligence*, 112(1-2):181–211, 1999.

M.E. Taylor and P. Stone. Value functions for RL-based behavior transfer: a comparative study. In *Proceedings of the Twentieth National Conference on Artificial Intelligence*, 2005.

M.E. Taylor, P. Stone, and Y. Liu. Transfer learning via inter-task mappings for temporal difference learning. *Journal of Machine Learning Research*, 8:2125–2167, 2007.

M.E. Taylor, G. Kuhlmann, and P. Stone. Autonomous transfer for reinforcement learning. In *Proceedings of the Seventh International Conference on Autonomous Agents and Multiagent Systems*, 2008.

S. Thrun and A. Schwartz. Finding structure in reinforcement learning. In *Advances in Neural Information Processing Systems*, volume 7, pages 385–392. The MIT Press, 1995.

L. Torrey, J. Shavlik, T. Walker, and R. Maclin. Skill acquisition via transfer learning and advice taking. In *Proceedings of the Seventeenth European Conference on Machine Learning*, pages 425–436, 2006.

J. Weng, J. McClelland, A. Pentland, O. Sporns, I. Stockman, M. Sur, and E. Thelen. Autonomous mental development by robots and animals. *Science*, 291(5504):599–600, 2000.

E. Wiewiora. Potential-based shaping and Q-value initialization are equivalent. *Journal of Artificial Intelligence Research*, 19:205–208, 2003.

E. Wiewiora, G. Cottrell, and C. Elkan. Principled methods for advising reinforcement learning agents. In *Proceedings of the Twentieth International Conference on Machine Learning*, pages 792–799, 2003.

A. Wilson, A. Fern, S. Ray, and P. Tadepalli. Multi-task reinforcement learning: a hierarchical bayesian approach. In *Proceedings of the 24th International Conference on Machine Learning*, pages 1015–1022, 2007.

W. Zhang and T.G. Dietterich. A reinforcement learning approach to job-shop scheduling. In *Proceedings of the Fourteenth International Conference on Artificial Intelligence*, pages 1114–1120, 1995.

Journal of Machine Learning Research 13 (2012) 1373-1392 Submitted 5/11; Revised 1/12; Published 5/12

On Ranking and Generalization Bounds

Wojciech Rejchel WREJCHEL@GMAIL.COM
Faculty of Mathematics and Computer Science
Nicolaus Copernicus University
Chopina 12/18
87-100 Toruń, Poland

Editor: Nicolas Vayatis

Abstract

The problem of ranking is to predict or to guess the ordering between objects on the basis of their observed features. In this paper we consider ranking estimators that minimize the empirical convex risk. We prove generalization bounds for the excess risk of such estimators with rates that are faster than $\frac{1}{\sqrt{n}}$. We apply our results to commonly used ranking algorithms, for instance boosting or support vector machines. Moreover, we study the performance of considered estimators on real data sets.

Keywords: convex risk minimization, excess risk, support vector machine, empirical process, U-process

1. Introduction

The problem of ranking is to predict or to guess the ordering between objects on the basis of their observed features. This problem has numerous applications in practice. We can mention information retrieval, banking, quality control or survival analysis. The problem is closely related to the classification theory, however it has its own specificity. In recent years many authors have focused their attention on this subject (Freund et al., 2004; Agarwal et al., 2005; Cossock and Zhang, 2006; Rudin, 2006; Clémençon et al., 2008).

In the paper we consider a population of objects equipped with a relation of (linear) ordering. For any two distinct objects o_1 and o_2 it holds either $o_1 \preceq o_2$ or $o_1 \succeq o_2$ (or maybe both), but it is unknown which is true. We lose little generality by assuming that real numbers y_1 and y_2 are assigned to the objects o_1 and o_2 in such a way that $o_1 \preceq o_2$ is equivalent to $y_1 \leq y_2$. Moreover, let d-dimensional vectors x_1 and x_2 describe observed or measured features of the objects and let the observation space $\mathcal{X}$ be a Borel subset of $\mathbb{R}^d$. We are to construct a function $f : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$, called a ranking rule, which predicts the ordering between objects in the following way:

$$\text{if } \ f(x_1,x_2) \leq 0, \text{ then we predict that } \ y_1 \leq y_2.$$

To measure the quality of a ranking rule f we introduce a probabilistic setting. Let us assume that two objects are randomly selected from the population. They are described by a pair of independent and identically distributed (with respect to the measure P) random vectors $Z_1 = (X_1, Y_1)$ and $Z_2 = (X_2, Y_2)$ taking values in $\mathcal{X} \times \mathbb{R}$. Random vectors X_1 and X_2 are regarded as observations, while Y_1 and Y_2 are unknown variables which define the ordering as above.

Most natural approach is to look for a function f which minimizes the risk (the probability of incorrect ranking)

$$L(f) = \mathbb{P}(\text{sign}(Y_1 - Y_2) f(X_1, X_2) < 0) \tag{1}$$

in some family of ranking rules $\mathcal{F}$, where $\text{sign}(t) = 1$ for $t > 0$, $\text{sign}(t) = -1$ for $t < 0$ and $\text{sign}(t) = 0$ for $t = 0$. Since we do not know the distribution P, we cannot solve this problem directly. But if we possess a learning sample $Z_1 = (X_1, Y_1), \ldots, Z_n = (X_n, Y_n)$, then we can consider a sample analog of (1), namely the empirical risk

$$L_n(f) = \frac{1}{n(n-1)} \sum_{i \neq j} \mathbb{I}[\text{sign}(Y_i - Y_j) f(X_i, X_j) < 0], \tag{2}$$

where $\mathbb{I}(\cdot)$ is the indicator function. The ranking rule that minimizes (2) can be used as an estimator of the function that minimizes (1). Notice that $L_n(f)$ is a U-statistic of the order two for a fixed $f \in \mathcal{F}$. The main difficulty in this approach lies in discontinuity of the function (2). It entails that finding its minimizer is computationally difficult and not effective. This fact is probably the main obstacle to wider use of such estimators in practice. To overcome this problem one usually replaces the discontinuous loss function by its convex analog. This trick has been successfully used in the classification theory and has allowed to invent boosting algorithms (Freund and Schapire, 1997) or support vector machines (Vapnik, 1998). Therefore, instead of $0-1$ loss function we consider a convex and nonnegative loss function $\psi : \mathbb{R} \to \mathbb{R}$. Denote the "convex" risk of a ranking rule f by

$$Q(f) = \mathbb{E}\, \psi[\, \text{sign}(Y_1 - Y_2)\, f(X_1, X_2)],$$

and the "convex" empirical risk as

$$Q_n(f) = \frac{1}{n(n-1)} \sum_{i \neq j} \psi_f(Z_i, Z_j),$$

where $\psi_f(z_1, z_2) = \psi[\text{sign}(y_1 - y_2)\, f(x_1, x_2)]$. Notice that $Q_n(f)$ is also a U-statistic of the order two for a fixed function f. Therefore, features of U-process $\{Q_n(f) : f \in \mathcal{F}\}$ are the basis for our consideration on statistical properties of the rule $f_n = \arg\min_{f \in \mathcal{F}} Q_n(f)$ as an estimator of the unknown function $f^* = \arg\min_{f \in \mathcal{F}} Q(f)$. Niemiro and Rejchel (2009) stated theorems about the strong consistency and the asymptotical normality of the estimator f_n in the linear case, that is, when we consider linear ranking rules $f(x_1, x_2) = \theta^T (x_1 - x_2)$, where $\theta \in \mathbb{R}^d$. Similar studies on the asymptotic behaviour of estimators were done in Niemiro (1992) and Bose (1998).

In this paper we are interested in the excess risk of an estimator f_n (in the general model, not necessarily linear). This is the case when one compares the convex risk of f_n with the convex risk of the best rule in the class. Generalization bounds are very popular for such studying in the learning theory. They are probabilistic inequalities of the following form: for every $\alpha \in (0, 1)$

$$\mathbb{P}(\ Q(f_n) - Q(f^*) \leq \eta\ ) \geq 1 - \alpha, \tag{3}$$

where $\eta > 0$ is some small number that depends on the level α, the number n of elements in the sample, a family of ranking rules $\mathcal{F}$ and a loss function ψ, but it is independent of an unknown distribution P. Similar objects were widely studied in the classification theory (Blanchard et al., 2003, 2008; Lugosi and Vayatis, 2004; Bartlett et al., 2006). In ranking one can find them in Clémençon

et al. (2005, 2008). In the latter two papers Authors proved that with some restrictions on the class $\mathcal{F}$ the number η in (3) is equal to $C\sqrt{\frac{\ln(1/\alpha)}{n}}$, where C is some constant. Their inequalities can be applied to ranking analogs of support vector machines or boosting algorithms. Moreover, it was shown in the classification theory that better rates than $\frac{1}{\sqrt{n}}$ are possible to obtain in similar bounds to (3). Noticing the close relation between ranking and the classification theory Clémençon et al. (2008) formulated the question if one can get generalization bounds with "fast rates" for the excess risk in ranking? They gave a positive answer (Clémençon et al., 2008, Corollary 6) but only for estimators that minimize the empirical risk with $0-1$ loss. We have already mentioned about problems with finding such minimizers. Convex loss functions and estimators that minimize the convex empirical risk are used in practice. In this paper we indicate assumptions and methods that allowed us to obtain generalization bounds with better rates than $\frac{1}{\sqrt{n}}$ for the excess convex risk of such estimators. Similar studies were done in Rejchel (2009), but here we strengthen and extend those results. The construction of inequalities of the form (3) is based on the empirical and U-process theory. Empirical processes are well-known and widely described in the literature, while U-processes are not so popular. However, there are very comprehensive monographs about this theory (see de la Peña and Giné, 1999), which originates from Hoeffding (1948).

The paper is organized as follows: Section 2 is devoted to theoretical results. We show that using Hoeffding's decomposition our problem can be divided into two parts. In the first one (Sections 2.1 and 2.2) we are interested in properties of some empirical process. The second part (Section 2.3) is devoted to a U-process that we obtain after Hoeffding's decomposition. We state the main theorem and describe its applications to commonly used ranking algorithms in Section 2.4. In Section 3 we study the practical performance of described estimators on real data sets.

2. Generalization Bounds

First, let us write conditions on a family of ranking rules $\mathcal{F}$ that we need in later work. For simplicity, assume that $f(x_1,x_2) = -f(x_2,x_1)$ for every $f \in \mathcal{F}$ which implies that the kernel of a U-statistic $Q_n(f)$ is symmetric. Moreover, let the class $\mathcal{F}$ be uniformly bounded which means that there exists some constant $A_1 > 0$ such that for every $x_1, x_2 \in \mathcal{X}$ and $f \in \mathcal{F}$ we have $|f(x_1,x_2)| \leq A_1$. We will not repeat these conditions later.

Furthermore, we need some restrictions on the "richness" of a family of ranking rules $\mathcal{F}$. They are bounds for the covering number of $\mathcal{F}$ and are similar to conditions that can be often found in the literature (Pollard, 1984; de la Peña and Giné, 1999; Mendelson, 2002). Thus, let μ be a probability measure on $\mathcal{X} \times \mathcal{X}$ and let ρ_μ be a $\mathbb{L}^2$-pseudometric on $\mathcal{F}$ defined as

$$\rho_\mu(f_1,f_2) = \frac{1}{A_1}\sqrt{\int_{\mathcal{X}\times\mathcal{X}} [f_1(x_1,x_2) - f_2(x_1,x_2)]^2 \, d\mu(x_1,x_2)}. \tag{4}$$

The covering number $N(t,\mathcal{F},\rho_\mu)$ of the class $\mathcal{F}$ with a pseudometric ρ_μ and a radius $t > 0$ is the minimal number of balls (with respect to ρ_μ) with centers in $\mathcal{F}$ and radii t needed to cover $\mathcal{F}$. Thus, $N(t,\mathcal{F},\rho_\mu)$ is the minimal number m with the property

$$\exists_{\mathcal{F}\subset\mathcal{F},\,|\mathcal{F}|=m} \quad \forall_{f\in\mathcal{F}} \quad \exists_{f\in\mathcal{F}} \quad \rho_\mu(f,f) \leq t.$$

Consider the marginal distribution P^X of the vector X and two empirical measures: $P_n^X = \frac{1}{n}\sum_{i=1}^n \delta_{X_i}$ and $\nu_n = \frac{1}{n(n-1)}\sum_{i\neq j}\delta_{(X_i,X_j)}$, where $\delta_{(\cdot)}$ is the counting measure. The family $\mathcal{F}$ that we consider satisfies one of the following conditions:

Assumption A There exist constants $D_i, V_i > 0$, $i = 1,2$ such that for every measures of the form: $\mu_1 = P^X \otimes P_n^X$, $\mu_2 = \nu_n$ and each $t \in (0,1]$ we have

$$N(t, \mathcal{F}, \rho_{\mu_i}) \leq D_i t^{-V_i} \qquad i = 1,2.$$

Assumption B There exist constants $D_i > 0$, $V_i \in (0,1)$, $i = 1,2$ such that for every measures of the form: $\mu_1 = P^X \otimes P_n^X$, $\mu_2 = \nu_n$ and each $t \in (0,1]$ we have

$$\ln N(t, \mathcal{F}, \rho_{\mu_i}) \leq D_i t^{-V_i} \qquad i = 1,2.$$

Families satisfying similar conditions to Assumption A are often called VC-classes or Euclidean (Nolan and Pollard, 1987; Pakes and Pollard, 1989), while classes that fulfill Assumption B are known as satisfying the uniform entropy condition (van der Vaart and Wellner, 1996). As we will see in Section 2.4 more restrictive Assumption A leads to better results.

The first tool that we use is Hoeffding's decomposition (de la Peña and Giné, 1999) of a U-statistic $Q_n(f) - Q_n(f^*)$ that allows to obtain the equality

$$Q(f) - Q(f^*) - [Q_n(f) - Q_n(f^*)] = 2P_n[Q(f) - Q(f^*) - P\psi_f + P\psi_{f^*}] - U_n(h_f - h_{f^*}),$$

where

$$\begin{aligned} P\psi_f(z_1) &= \mathbb{E}[\psi_f(Z_1, Z_2) | Z_1 = z_1], \\ P_n(g) &= \frac{1}{n}\sum_{i=1}^{n} g(Z_i), \\ U_n(h_f - h_{f^*}) &= \frac{1}{n(n-1)}\sum_{i \neq j} [h_f(Z_i, Z_j) - h_{f^*}(Z_i, Z_j)], \\ h_f(z_1, z_2) &= \psi_f(z_1, z_2) - P\psi_f(z_1) - P\psi_f(z_2) + Q(f). \end{aligned}$$

Therefore, Hoeffding's decomposition breaks a difference between a U-statistic $Q_n(f) - Q_n(f^*)$ and its expectation into the sum of iid random variables and a degenerate U-statistic $U_n(h_f - h_{f^*})$. The degeneration of a U-statistic means that the conditional expectation of its kernel is the zero-function, that is, $\mathbb{E}[h_f(Z_1, Z_2) - h_{f^*}(Z_1, Z_2) | Z_1 = z_1] = 0$ for each $z_1 \in \mathcal{X} \times \mathbb{R}$. In what follows, we will separately look for probabilistic inequalities of the appropriate order for the empirical and degenerate term.

2.1 Empirical Term

The empirical process theory is the basis for our consideration concerning the first component in Hoeffding's decomposition of U-statistics. To get better rates in this case one has to be able to uniformly bound second moments of functions from an adequate class by their expectations. This fact combined with some consequence of Talagrand's inequality (Talagrand, 1994) was the key to obtain fast rates in the classification theory. In this subsection we want to apply this method to ranking. First, we need a few preliminaries: let $\mathcal{G}$ be a class of real functions that is uniformly bounded by a constant $G > 0$. Moreover, let us introduce an additional sequence of iid random variables $\varepsilon_1, \ldots, \varepsilon_n$ (the Rademacher sequence). Variables ε_i's take values 1 or -1 with probability $\frac{1}{2}$ and are independent of the sample $Z_1, \ldots, Z_n$. Having the Rademacher sequence let us denote

$$R_n(\mathcal{G}) = \sup_{g \in \mathcal{G}} \frac{1}{n}\sum_{i=1}^{n} \varepsilon_i g(Z_i),$$

and call an expression $\mathbb{E}R_n(\mathcal{G})$ the Rademacher average of the class $\mathcal{G}$. The expectation in the Rademacher average is taken with respect to both samples $Z_1,\ldots,Z_n$ and $\varepsilon_1,\ldots,\varepsilon_n$.

Besides, we should also introduce so called sub-root functions, which are nonnegative and non-decreasing functions $\phi : [0,\infty) \to [0,\infty)$ such that for each $r > 0$ the function $r \longmapsto \phi(r)/\sqrt{r}$ is non-increasing. They have a lot of useful properties, for example they are continuous and have the unique positive fixed point r^* (the positive solution of the equation $\phi(r) = r$). Proofs of these facts can be easily found in the literature (Bartlett et al., 2005). Finally, let the class

$$\mathcal{G}^* = \{\alpha g : g \in \mathcal{G}, \alpha \in [0,1]\}$$

denote a star-hull of $\mathcal{G}$ and $Pg = \mathbb{E}\, g(Z_1)$. Now we can state the aforementioned theorem for empirical processes which can be also found in Massart (2000) and Bartlett et al. (2005).

Theorem 1 *Let the class $\mathcal{G}$ be such that for some constant $B > 0$ and every $g \in \mathcal{G}$ we have $Pg^2 \leq BPg$. Moreover, if there exists a sub-root function ϕ with the fixed point r^*, which satisfies*

$$\phi(r) \geq B\,\mathbb{E}R_n(g \in \mathcal{G}^* : Pg^2 \leq r)$$

for each $r \geq r^*$, then for every $K > 1$ and $\alpha \in (0,1)$

$$\mathbb{P}\left(\forall_{g\in\mathcal{G}} \quad Pg \leq \frac{K}{K-1}P_n(g) + \frac{6K}{B}r^* + [22G + 5BK]\frac{\ln(1/\alpha)}{n}\right) \geq 1-\alpha.$$

The proof of this theorem is based on Talagrand's inequality applied to properly rescaled class $\mathcal{G}$ and can be found in Bartlett et al. (2005). Theorem 1 says that to get better bounds for the empirical term one needs to study properties of the fixed point r^* of a sub-root ϕ. However, it gives no general method for choosing ϕ, but it suggests to relate it to $\mathbb{E}R_n(g \in \mathcal{G}^* : Pg^2 \leq r)$. We will follow this suggestion, similar reasoning was carried out in Bartlett et al. (2005) or Boucheron et al. (2005). Of course, for every n the function

$$r \to \mathbb{E}R_n(g \in \mathcal{G}^* : Pg^2 \leq r)$$

is nonnegative and non-decreasing. Replacing a class $\mathcal{G}$ by its star-hull is needed to prove the last property from the definition of the sub-root function.

Using Theorem 1 we can state the following fact concerning the empirical term in Hoeffding's decomposition. The modulus of convexity of a function ψ that appears in this theorem is described in the next subsection.

Theorem 2 *Let the family of ranking rules $\mathcal{F}$ satisfy Assumption A and be convex. Moreover, if the modulus of convexity of a loss function ψ fulfills on the interval $[-A_1,A_1]$ the condition $\delta(t) \geq Ct^p$ for some constants $C > 0$ and $p \leq 2$, then for every $\alpha \in (0,1)$ and $K > 1$*

$$\mathbb{P}\left(\forall_{f\in\mathcal{F}} \quad Q(f) - Q(f^*) \leq \frac{K}{K-1}P_n(P\psi_f - P\psi_{f^*}) + C_1V_1\frac{\ln n + \ln(1/\alpha)}{n}\right) \geq 1-\alpha,$$

where the constant C_1 depends on K.

If the family $\mathcal{F}$ satisfies Assumption B instead of Assumption A, then for every $\alpha \in (0,1)$ and $K > 1$ with probability at least $1-\alpha$

$$\forall_{f\in\mathcal{F}} \quad Q(f) - Q(f^*) \leq \frac{K}{K-1}P_n(P\psi_f - P\psi_{f^*}) + C_2\max\left(\frac{\ln n}{n}, \frac{1}{n^\beta}\right) + C_3\frac{\ln(1/\alpha)}{n}$$

where $\frac{2}{3} < \beta = \frac{2}{2+V_1} < 1$. Constants C_2, C_3 depend on K.

Remark 3 *Although the constants C_1, C_2, C_3 can be recovered from the proofs we do not write their explicit formulas, because our task is to prove bounds with better rates, that is, which decrease fast with $n \to \infty$. For the same reason we do not attempt to optimize C_1, C_2 and C_3.*

Proof Consider the family of functions

$$P\psi_{\mathcal{F}} - P\psi_{f^*} = \{P\psi_f - P\psi_{f^*} : f \in \mathcal{F}\}.$$

A loss function ψ is convex so it is locally Lipschitz with constant L_ψ. Since $\mathcal{F}$ is uniformly bounded, then $P\psi_{\mathcal{F}} - P\psi_{f^*}$ is also uniformly bounded by $2L_\psi A_1$. Moreover, we show in the next subsection that if $\mathcal{F}$ is convex and the modulus of convexity of ψ satisfies the assumption given in Theorem 2, then one can prove that for some constant B and every function $f \in \mathcal{F}$

$$\mathbb{E}\,[P\psi_f(Z_1) - P\psi_{f^*}(Z_1)]^2 \leq B[Q(f) - Q(f^*)]. \tag{5}$$

The precise value of the constant B is given in Lemma 5 in Section 2.2. Therefore, the relation that is demanded in Theorem 1 between second moments and expectations of functions from the considered class holds. Applying this theorem to the class of functions $\mathcal{G} = \left\{ \frac{P\psi_f - P\psi_{f^*}}{2L_\psi A_1} : f \in \mathcal{F} \right\}$ and the sub-root function

$$\phi(r) = \frac{B}{2L_\psi A_1} \mathbb{E} R_n(g \in \mathcal{G}^* : Pg^2 \leq r)$$

we get the following probabilistic inequality

$$\mathbb{P}\left(\forall_{f \in \mathcal{F}} \quad Q(f) - Q(f^*) \leq \frac{K}{K-1} P_n\left(P\psi_f - P\psi_{f^*}\right) + C_1 r^* + C_2 \frac{\ln(1/\alpha)}{n} \right) \geq 1 - \alpha\,.$$

Constants C_1, C_2 and others that appear in this proof may change from line to line. To finish the proof of the first part of the theorem we have to bound the fixed point of the sub-root ϕ by $\frac{\ln n}{n}$. Described method is similar to consideration contained in Mendelson (2003) or Bartlett et al. (2005).

First we need two additional notations:

$$\mathcal{G}_r^* = \{g \in \mathcal{G}^* : Pg^2 \leq r\}$$

for some $r > 0$ and

$$\xi = \sup_{g \in \mathcal{G}_r^*} \frac{1}{n} \sum_{i=1}^n g^2(Z_i).$$

Using Chaining Lemma for empirical processes (Pollard, 1984) we obtain

$$\mathbb{E} R_n(\mathcal{G}_r^*) \leq \frac{C_1}{\sqrt{n}} \mathbb{E} \int_0^{\sqrt{\xi}/4} \sqrt{\ln N(t, \mathcal{G}_r^*, \rho_{P_n})}\, dt, \tag{6}$$

where

$$\rho_{P_n}(g_1, g_2) = \sqrt{\frac{1}{n} \sum_{i=1}^n [g_1(Z_i) - g_2(Z_i)]^2}\,.$$

Notice that $N(t, \mathcal{G}_r^*, \rho_{P_n}) \leq N(t, \mathcal{G}^*, \rho_{P_n}) \leq N(t/2, \mathcal{G}, \rho_{P_n}) \left\lceil \frac{1}{t} \right\rceil$ since from a cover of a family $\mathcal{G}$ with radius $t/2$ and a cover of the interval $[0,1]$ with radius $t/2$ one can easily construct a cover of a family $\mathcal{G}^*$. Besides, it is not difficult to show that

$$N(t, P\psi_{\mathcal{F}}, \rho_{P_n}) \leq N(t, \psi_{\mathcal{F}}, \rho_{P \otimes P_n}) \leq N\left(\frac{t}{L_\psi}, \mathcal{F}, \rho_{P^X \otimes P_n^X} \right),$$

since the first inequality follows from Nolan and Pollard (1987, Lemma 20) and to prove the second one we use the fact that ψ is locally Lipschitz. Thus, Assumption A and above properties of covering numbers imply that for some positive constants C and C_1

$$\ln N(t, \mathcal{G}_r^*, \rho_{P_n}) \leq C_1 V_1 \ln \frac{C}{t}.$$

So the right side of (6) can be bounded by $C_1 \sqrt{\frac{V_1}{n}}\, \mathbb{E} \int_0^{\sqrt{\xi}/4} \sqrt{\ln \frac{C}{t}}\, dt$. Using Mendelson (2003, Lemma 3.8) and Jensen's inequality we obtain

$$C_1 \sqrt{\frac{V_1}{n}}\, \mathbb{E} \int_0^{\sqrt{\xi}/4} \sqrt{\ln \frac{C}{t}}\, dt \leq C_1 \sqrt{\frac{V_1}{n}} \sqrt{\mathbb{E}\xi} \sqrt{\ln\left(\frac{C}{\mathbb{E}\xi}\right)}.$$

Furthermore, applying Talagrand (1994, Corollary 3.4) to the family $\mathcal{G}_r^*$ we have

$$\mathbb{E}\xi \leq 8\mathbb{E} R_n(\mathcal{G}_r^*) + r.$$

Summarizing we have just shown that

$$\mathbb{E} R_n(\mathcal{G}_r^*) \leq C_1 \sqrt{\frac{V_1}{n}} \sqrt{8\mathbb{E} R_n(\mathcal{G}_r^*) + r} \sqrt{\ln \frac{C}{r}}$$

which for the fixed point r^* implies

$$r^* \leq \frac{C_1 V_1}{n} \ln \frac{C}{r^*},$$

and now it is easy to get that $r^* \leq C V_1 \frac{\ln n}{n}$.

In the second part of the theorem we use less restrictive Assumption B. Reasoning is the same as in the previous case, we need only to notice that

$$\ln N(t, \mathcal{G}_r^*, \rho_{P_n}) \leq C \left[\ln N\left(t, \mathcal{F}, \rho_{P^X \otimes P_n^X}\right) + \ln \frac{C_1}{t} \right] \leq C \left[t^{-V_1} + \ln \frac{C_1}{t} \right].$$

Therefore, the right side of (6) can be bounded by

$$\frac{C}{\sqrt{n}}\, \mathbb{E} \int_0^{\sqrt{\xi}/4} \sqrt{t^{-V_1}}\, dt + \frac{C}{\sqrt{n}}\, \mathbb{E} \int_0^{\sqrt{\xi}/4} \sqrt{\ln \frac{C_1}{t}}\, dt. \tag{7}$$

The second component in (7) has been just considered, so we focus on the first one. Notice that it is equal to $\frac{C}{\sqrt{n}}\, \mathbb{E}\xi^{1/2 - V_1/4}$. Again, using Jensen's inequality and Talagrand (1994, Corollary 3.4) it is less than

$$\frac{C}{\sqrt{n}} \left[8\mathbb{E} R_n(\mathcal{G}_r^*) + r \right]^{1/2 - V_1/4}$$

which implies that

$$\mathbb{E} R_n(\mathcal{G}_r^*) \leq \frac{C}{\sqrt{n}} \left[(8\mathbb{E} R_n(\mathcal{G}_r^*) + r)^{\frac{1}{2} - \frac{V_1}{4}} + \sqrt{(8\mathbb{E} R_n(\mathcal{G}_r^*) + r) \ln \frac{C_1}{r}} \right]. \tag{8}$$

For the fixed point r^* the inequality (8) takes the form

$$r^* \leq \frac{C}{\sqrt{n}} \left[(r^*)^{\frac{1}{2}-\frac{V_1}{4}} + \sqrt{r^* \ln \frac{C_1}{r^*}} \right],$$

so

$$r^* \leq C \max \left(\frac{\ln n}{n}, \frac{1}{n^{\frac{2}{2+V_1}}} \right)$$

and $\frac{2}{3} < \frac{2}{2+V_1} < 1$, since $0 < V_1 < 1$. ■

2.2 On the Inequality (5)

Theorem 2 in the previous subsection shows that better rates can be obtained if we are able to bound second moments of functions from the family $P\psi_{\mathcal{F}} - P\psi_{f^*}$ by their expectations. In this subsection we indicate conditions that are sufficient for even stronger relationship, namely

$$\mathbb{E}\left[\psi_f(Z_1,Z_2) - \psi_{f^*}(Z_1,Z_2)\right]^2 \leq B[Q(f) - Q(f^*)]. \tag{9}$$

The key object in further analysis is the modulus of convexity of the loss ψ. This function was very helpful in proving similar relation in the classification theory (Mendelson, 2002; Bartlett et al., 2006). With minor changes we will use it in our studies.

Definition 4 *The modulus of convexity of* ψ *is the function* $\delta : [0,\infty) \to [0,\infty]$ *defined as*

$$\delta(t) = \inf \left\{ \frac{\psi(x_1)+\psi(x_2)}{2} - \psi\left(\frac{x_1+x_2}{2}\right) : |x_1 - x_2| \geq t \right\}.$$

We illustrate this object with a few examples: for the quadratic function $\psi(x) = x^2$ we obtain $\delta(t) = t^2/4$, the modulus of convexity of the exponential function defined on the interval $[-a,a]$ is equal to $\delta(t) = \frac{t^2}{8\exp(a)} + o(t^2)$, whereas for $\psi(x) = \max[0, 1-x]$ we have $\delta(t) = 0$.

If the class $\mathcal{F}$ is convex, then the risk $Q : \mathcal{F} \to \mathbb{R}$ is the convex functional. It allows to consider the modulus of convexity of Q, that is given by

$$\tilde{\delta}(t) = \inf \left\{ \frac{Q(f_1)+Q(f_2)}{2} - Q\left(\frac{f_1+f_2}{2}\right) : d(f_1,f_2) \geq t \right\},$$

where d is the $\mathbb{L}^2$-pseudometric on $\mathcal{F}$, that is,

$$d(f_1,f_2) = \sqrt{\mathbb{E}\left[f_1(X_1,X_2) - f_2(X_1,X_2)\right]^2}.$$

The important property of the modulus of convexity is the fact that it can be often lower bounded by Ct^p for some $C, p > 0$. This relation is satisfied for many interesting convex functions, for instance e^{-x}, $\log_2(1+e^{-2x})$ or $[\max(0,1-x)]^2$ (the last case needs minor changes in consideration). This property implies the similar one for the modulus of convexity of the functional Q, which is sufficient to prove the relationship (9) between second moments and expectations of functions from the family $\psi_{\mathcal{F}} - \psi_{f^*}$. The following lemma, which is based on Bartlett et al. (2006, Lemma 7 and Lemma 8), can be stated:

Lemma 5 *If the family $\mathcal{F}$ is convex and there exist constants $C, p > 0$ such that the modulus of convexity of ψ satisfies*

$$\delta(t) \geq Ct^p, \tag{10}$$

then

$$\mathbb{E}\left[\psi_f(Z_1,Z_2) - \psi_{f^*}(Z_1,Z_2)\right]^2 \leq L_\psi^2 D_p [Q(f) - Q(f^*)]_,^{\min(1,2/p)} \tag{11}$$

where

$$D_p = \begin{cases} (2C)^{-2/p} & \text{if} \quad p \geq 2, \\ 2^{1-p} A_1^{2-p} C^{-1} & \text{if} \quad p < 2. \end{cases}$$

Proof Using Lipschitz property of ψ we can obtain

$$\begin{aligned} & \mathbb{E}\left[\psi_f(Z_1,Z_2) - \psi_{f^*}(Z_1,Z_2)\right]^2 \\ \leq \;& L_\psi^2 \mathbb{E}\left[\operatorname{sign}(Y_1 - Y_2) f(X_1,X_2) - \operatorname{sign}(Y_1 - Y_2) f^*(X_1,X_2)\right]^2 \\ = \;& L_\psi^2 d^2(f, f^*). \end{aligned} \tag{12}$$

The second step of the proof is based on showing that if the modulus δ satisfies (10), then the modulus $\tilde{\delta}$ also fulfills a similar condition. Namely, let $f_1, f_2 \in \mathcal{F}$ satisfy $d(f,g) \geq t$. Then from the definition of the modulus of convexity δ and (10)

$$\begin{aligned} & \frac{Q(f_1) + Q(f_2)}{2} - Q\left(\frac{f_1 + f_2}{2}\right) = \mathbb{E}\left[\frac{\psi_{f_1}(Z_1,Z_2) + \psi_{f_2}(Z_1,Z_2)}{2} - \psi_{\frac{f_1+f_2}{2}}(Z_1,Z_2)\right] \\ \geq \;& \mathbb{E}\,\delta(|\operatorname{sign}(Y_1 - Y_2) f_1(X_1,X_2) - \operatorname{sign}(Y_1 - Y_2) f_2(X_1,X_2)|) \\ = \;& \mathbb{E}\,\delta(|f_1(X_1,X_2) - f_2(X_1,X_2)|) \geq C\,\mathbb{E}\,|f_1(X_1,X_2) - f_2(X_1,X_2)|^p. \end{aligned}$$

Easy calculation (see Bartlett et al., 2006, the proof of Lemma 8) indicates that the modulus $\tilde{\delta}$ fulfills

$$\tilde{\delta}(t) \geq C_p t^{\max(2,p)}, \tag{13}$$

where $C_p = C$ for $p \geq 2$ and $C_p = C(2A_1)^{p-2}$, otherwise. Moreover, from the definition of the modulus $\tilde{\delta}$ and the fact that f^* is the minimizer of $Q(f)$ in the convex class $\mathcal{F}$ we have

$$\frac{Q(f) + Q(f^*)}{2} \geq Q\left(\frac{f + f^*}{2}\right) + \tilde{\delta}(d(f,f^*)) \geq Q(f^*) + \tilde{\delta}(d(f,f^*)).$$

Combining this fact with the inequality (12) and the property (13) of the modulus $\tilde{\delta}$ we get

$$\begin{aligned} Q(f) - Q(f^*) &\geq 2\tilde{\delta}\left(\frac{\sqrt{\mathbb{E}\left[\psi_f(Z_1,Z_2) - \psi_{f^*}(Z_1,Z_2)\right]^2}}{L_\psi}\right) \\ &\geq 2C_p\left(\frac{\sqrt{\mathbb{E}\left[\psi_f(Z_1,Z_2) - \psi_{f^*}(Z_1,Z_2)\right]^2}}{L_\psi}\right)^{\max(2,p)} \end{aligned}$$

which is equivalent to the inequality (11). ■

Thus, for convex functions that were mentioned before Lemma 5 we obtain in the inequality (11) the exponent equal to 1, because their modulus of convexity can be easily bounded from below with $p = 2$. However, if $p > 2$, then the exponent belongs to the interval $(0,1)$, but we can still bound the considered empirical process by an expression of the order better than $\frac{1}{\sqrt{n}}$ (Mendelson, 2002; Bartlett et al., 2006). Of course, we get better bounds if the exponent is closer to 1.

2.3 Degenerate Component

In this subsection we obtain exponential inequalities for degenerate U-processes. We bound the second term in Hoeffding's decomposition by $\frac{1}{n}$ that is sufficient to get better rates for the excess risk of ranking estimators. Let us recall that considered object has the following form

$$\left\{ U_n(h_f - h_{f^*}) = \frac{1}{n(n-1)} \sum_{i \neq j} [h_f(Z_i, Z_j) - h_{f^*}(Z_i, Z_j)] : f \in \mathcal{F} \right\}, \tag{14}$$

where

$$h_f(z_1, z_2) = \psi_f(z_1, z_2) - P\psi_f(z_1) - P\psi_f(z_2) + Q(f).$$

Moreover, kernels of the U-process (14) are symmetric, uniformly bounded and degenerate.

Similar problems were also considered in Arcones and Giné (1994); de la Peña and Giné (1999), Major (2006) and Adamczak (2007).

Theorem 6 *If a family of ranking rules $\mathcal{F}$ satisfies Assumption A, then for every $\alpha \in (0,1)$*

$$\mathbb{P}\left(\quad \forall_{f \in \mathcal{F}} \quad |U_n(h_f - h_{f^*})| \leq C_1 \max(V_1, V_2) \frac{\ln(C_2/\alpha)}{n} \right) \geq 1 - \alpha$$

for some constants $C_1, C_2 > 0$.

If a family of ranking rules $\mathcal{F}$ satisfies Assumption B, then for every $\alpha \in (0,1)$

$$\mathbb{P}\left(\quad \forall_{f \in \mathcal{F}} \quad |U_n(h_f - h_{f^*})| \leq \frac{C_3}{1 - \max(V_1, V_2)} \frac{\ln(C_4/\alpha)}{n} \right) \geq 1 - \alpha$$

for some constants $C_3, C_4 > 0$.

Remark 7 *In Assumption B we restrict to $V_1, V_2 < 1$, whereas in the empirical process theory these exponents usually belong to $(0,2)$. This restriction is needed to prove Theorem 6, namely to calculate the integral (19) in the proof of this theorem.*

Proof Our aim is to bound the expression

$$\mathbb{E} \exp\left(\lambda \sqrt{\sup_{f \in \mathcal{F}} |(n-1) U_n(h_f - h_{f^*})|} \right) \tag{15}$$

for every $\lambda > 0$. Combining it with Markov's inequality finishes the proof.

Introduce the Rademacher sequence $\varepsilon_1, \ldots, \varepsilon_n$ and the symmetrized U-process defined as

$$S_n(h_f - h_{f^*}) = \frac{1}{n(n-1)} \sum_{i \neq j} \varepsilon_i \varepsilon_j [h_f(Z_i, Z_j) - h_{f^*}(Z_i, Z_j)].$$

Using Symmetrization for U-processes (de la Peña and Giné, 1999) we can bound (15) by

$$C_2 \mathbb{E} \exp\left(C_1 \lambda \sqrt{\sup_{f \in \mathcal{F}} |(n-1) S_n(h_f - h_{f^*})|} \right). \tag{16}$$

Constants C_1, C_2 that appear in this proof may differ from line to line. If we fix $Z_1, \ldots, Z_n$, then we work with the Rademacher chaos process whose properties are well studied. By Arcones and Giné (1994, Formula 3.4 and 3.5) and Hölder's inequality we bound (16) by

$$C_2 \mathbb{E} \exp\left(C_1 \lambda^2 \mathbb{E}_\varepsilon \sup_{f \in \mathcal{F}} |(n-1) S_n (h_f - h_{f^*})| \right),$$

where $\mathbb{E}_\varepsilon$ is conditional expectation with respect to the Rademacher sequence. To finish the proof we study the expression $\mathbb{E}_\varepsilon \sup_{f \in \mathcal{F}} |(n-1) S_n (h_f - h_{f^*})|$. This step relies on Chaining Lemma for U-processes (Nolan and Pollard, 1987, Lemma 5), similar reasoning can be found in Arcones and Giné (1993) and Sherman (1993). For convenience let us denote $h_f - h_{f^*}$ by h. Furthermore, for fixed $Z_1, \ldots, Z_n$ consider a stochastic process

$$\left\{ J_n(h) = \frac{1}{En} \sum_{i \neq j} \varepsilon_i \varepsilon_j h(Z_i, Z_j) : h \in \mathcal{H} \right\}, \tag{17}$$

where E is the uniform bound on elements of $\mathcal{H}$. Define a pseudometric ρ on $\mathcal{H} = \{h_f - h_{f^*} : f \in \mathcal{F}\}$ as

$$\rho(h_1, h_2) = \frac{1}{E} \sqrt{\frac{1}{n(n-1)} \sum_{i \neq j} [h_1(Z_i, Z_j) - h_2(Z_i, Z_j)]^2}.$$

The process (17) satisfies assumptions of Chaining Lemma for U-processes with the function $\phi(x) = \exp\left(\frac{x}{\kappa} - 1\right)$, where κ is some positive constant. Indeed, $J_n(h_1 - h_2)$ is the Rademacher chaos of the order two, so from de la Peña and Giné (1999, Corollary 3.2.6) there exists $\kappa > 0$ such that

$$\mathbb{E}_\varepsilon \exp\left(\frac{|J_n(h_1 - h_2)|}{\kappa \sqrt{\mathbb{E}_\varepsilon [J_n(h_1 - h_2)]^2}} \right) \leq e.$$

Moreover, it is easy to calculate that $\mathbb{E}_\varepsilon [J_n(h_1 - h_2)]^2 \leq \rho^2(h_1, h_2)$, which implies that $\mathbb{E}_\varepsilon \phi\left(\frac{|J_n(h_1 - h_2)|}{\rho(h_1, h_2)} \right) \leq 1$. Therefore, we obtain the inequality

$$\mathbb{E}_\varepsilon \sup_{f \in \mathcal{F}} |(n-1) S_n (h_f - h_{f^*})| \leq C_1 \int_0^{1/4} \ln N(t, \mathcal{H}, \rho)\, dt. \tag{18}$$

Besides, the covering number of the family $\mathcal{H}_1 + \mathcal{H}_2 = \{h_1 + h_2 : h_1 \in \mathcal{H}_1, h_2 \in \mathcal{H}_2\}$ clearly satisfies the inequality

$$N(2t, \mathcal{H}_1 + \mathcal{H}_2, \rho) \leq N(t, \mathcal{H}_1, \rho) N(t, \mathcal{H}_2, \rho).$$

If the family $\mathcal{F}$ fulfills Assumption A, then similarly to the proof of Theorem 2 we have $N(t, P\psi_{\mathcal{F}}, \rho_{P_n}) \leq C_1 t^{-V_1}$ and $N(t, \psi_{\mathcal{F}}, \rho) \leq C_2 t^{-V_2}$. Therefore, for some constants $C, C_1 > 0$

$$N(t, \mathcal{H}, \rho) \leq C t^{-C_1 \max(V_1, V_2)}$$

and the right-hand side of (18) is bounded (for some constants $C, C_1, C_2 > 0$) by

$$C_1 \max(V_1, V_2) \int_0^{1/4} \ln \frac{C}{t}\, dt \leq C_2 \max(V_1, V_2).$$

If the family satisfies Assumption B, then the right-hand side of (18) is bounded (for some constants $C, C_1 > 0$) by

$$C\int_0^{1/4} t^{-\max(V_1,V_2)}\,dt \le \frac{C_1}{1-\max(V_1,V_2)}\,. \tag{19}$$

Summarizing we obtain for every $\lambda > 0$

$$\mathbb{E}\exp\left(\lambda\sqrt{\sup_{f\in\mathcal{F}}|(n-1)U_n(h_f - h_{f^*})|}\right) \le C_2\exp(C_1\lambda^2)$$

and the form of the constant C_1 depends on the assumption (A or B) that is satisfied by the family $\mathcal{F}$. Finally, we take $\lambda = \sqrt{\frac{\ln(C_2/\alpha)}{C_1}}$ and use Markov's inequality. ■

2.4 Main Result and Examples

Our task relied on showing that in ranking, similarly to the classification theory, the convex excess risk can be bounded with better rates than $\frac{1}{\sqrt{n}}$ which were proved in Clémençon et al. (2008). By Hoeffding's decomposition the effort was divided into the empirical term (Sections 2.1 and 2.2) and the degenerate U-process (Section 2.3). Taking results of these three parts together we can state the main theorem.

Theorem 8 *Let the family of ranking rules $\mathcal{F}$ satisfy Assumption A and be convex. Moreover, if the modulus of convexity of a function ψ fulfills on the interval $[-A_1, A_1]$ the condition $\delta(t) \ge Ct^p$ for some constants $C > 0$ and $p \le 2$, then for every $\alpha \in (0,1)$*

$$\mathbb{P}\left(Q(f_n) - Q(f^*) \le C_1\max(V_1,V_2)\frac{\ln n + \ln(C_2/\alpha)}{n}\right) \ge 1-\alpha \tag{20}$$

for some constants C_1, C_2.

If the family $\mathcal{F}$ satisfies Assumption B instead of Assumption A, then for every $\alpha \in (0,1)$

$$\mathbb{P}\left(Q(f_n) - Q(f^*) \le C_3\max\left(\frac{\ln n}{n}, \frac{1}{n^\beta}\right) + \frac{C_4}{1-\max(V_1,V_2)}\frac{\ln(C_5/\alpha)}{n}\right) \ge 1-\alpha$$

for some constants C_3, C_4, C_5 and $\beta = \frac{2}{2+V_1} \in \left(\frac{2}{3}, 1\right)$.

Remark 9 *The dependence on exponents V_1, V_2 in the inequality (20) is the same as in Clémençon et al. (2008, Corollary 6), where one considered minimizers of the empirical risk with $0-1$ loss and the family $\mathcal{F}$ with finite Vapnik-Chervonenkis dimension.*

Proof Let us slightly modify Hoeffding's decomposition of the U-statistic $Q_n(f) - Q_n(f^*)$, namely for each $K > 2$

$$\begin{aligned}&(K-2)[Q(f) - Q(f^*)] - K[Q_n(f) - Q_n(f^*)]\\ &= 2P_n\left\{(K-1)[Q(f) - Q(f^*)] - K(P\psi_f - P\psi_{f^*})\right\} - K[U_n(h_f) - U_n(h_{f^*})].\end{aligned}$$

Therefore, the first part of Theorem 2, Lemma 5 and Theorem 6 are sufficient to prove that for every $\alpha \in (0,1)$ and $K > 2$ with probability at least $1-\alpha$

$$\forall_{f\in\mathcal{F}} \quad Q(f) - Q(f^*) \leq \frac{K}{K-2}[Q_n(f) - Q_n(f^*)] + C_1 \max(V_1, V_2)\left[\frac{\ln n + \ln(C_2/\alpha)}{n}\right]$$

for some constants C_1, C_2. Moreover, for the rule f_n that minimizes the empirical convex risk we have $Q_n(f_n) - Q_n(f^*) \leq 0$. If the family $\mathcal{F}$ satisfies Assumption B we use the second thesis of Theorem 2 and further reasoning is the same. ■

Now we give three examples of ranking procedures that we can apply Theorem 8 to.

Example 1 *Consider the family $\mathcal{F}$ containing linear ranking rules*

$$\mathcal{F} = \{f(x_1,x_2) = \theta^T(x_1 - x_2) : \theta, x_1, x_2 \in \mathbb{R}^d\}$$

In this case our prediction of the ordering between objects depends on the hyperplane that the vector $x_1 - x_2$ belongs to. The family $\mathcal{F}$ is convex. Moreover, the class $\{subgraph(f) : f \in \mathcal{F}\}$, where

$$subgraph(f) = \{(x_1,x_2,t) \in \mathcal{X}^2 \times \mathbb{R} : 0 < t < f(x_1,x_2) \text{ or } f(x_1,x_2) < t < 0\},$$

is by Pakes and Pollard (1989, Lemma 2.4 and 2.5) a VC-class of sets. Thus, Pakes and Pollard (1989, Lemma 2.12) implies that the family $\mathcal{F}$ satisfies Assumption A. If we take a "good" function ψ (for example one of functions mentioned in Section 2.2), then we obtain generalization bounds for the excess risk of the estimator f_n of the order $\frac{\ln n}{n}$.

Theorem 8 can be also applied to a popular ranking procedure called "boosting". Here we are interested in a ranking version of AdaBoost that uses the exponential loss function.

Example 2 *Let $\mathcal{R} = \{r : \mathcal{X} \times \mathcal{X} \to \{-1,1\}\}$ be a family of "base" ranking rules with finite Vapnik-Chervonenkis dimension. The output of the algorithm is an element of a convex T-hull of $\mathcal{R}$, where T is the number of iterations of the procedure. Namely, it belongs to the family*

$$conv_T(\mathcal{R}) = \{f(x_1,x_2) = \sum_{j=1}^{T} w_j r_j(x_1,x_2) : \sum_{j=1}^{T} w_j = A_1,$$
$$w_j \geq 0, r_j \in \mathcal{R} \text{ for } j = 1,\ldots,T\}.$$

This class is obviously convex. The family $\mathcal{R}$ has finite VC dimension, so a class

$$\{A_r = \{(x_1,x_2) : r(x_1,x_2) = 1\} : r \in \mathcal{R}\}$$

is a VC-class of sets. The subgraph of each $r \in \mathcal{R}$ has the following form

$$\{(x_1,x_2) \in A_r \text{ and } t \in (0,1)\} \cup \{(x_1,x_2) \in A_r^c \text{ and } t \in (-1,0)\}.$$

Again using Pakes and Pollard (1989, Lemma 2.5 and 2.12) we obtain that $N(t,\mathcal{R},\rho_\mu) \leq Ct^{-V}$ for some constants $C, V > 0$ and every probability measure μ on $\mathcal{X} \times \mathcal{X}$. Quick calculation shows that

$$N(t, conv_T(\mathcal{R}), \rho_\mu) \leq C_1 t^{-T(V+1)},$$

so $\mathcal{F}$ satisfies Assumption A. Furthermore, the modulus of convexity of $\psi(x) = \exp(-x)$ fulfills on the interval $[-A_1, A_1]$ the condition $\delta(t) > \frac{t^2}{8\exp(A_1)}$. Thus, in this example we also obtain generalization bounds for the excess convex risk of f_n of the order $\frac{\ln n}{n}$.

The last example is a ranking version of support vector machines.

Example 3 *Let $K : \mathcal{X}^2 \times \mathcal{X}^2 \to \mathbb{R}$ be a kernel that is symmetric, continuous and nonnegative definite function. The last property means that for every natural number m, vectors $x_1, \ldots, x_m \in \mathcal{X}^2$ and $\alpha_1, \ldots, \alpha_m \in \mathbb{R}$*

$$\sum_{i,j=1}^{m} \alpha_i \alpha_j K(x_i, x_j) \geq 0.$$

One can show (Cucker and Smale, 2002) that for every kernel K there exists the unique Hilbert space H_K (called reproducing kernel Hilbert space) of real functions on $\mathcal{X}^2$ that the inner product of its elements is defined by K. Namely, H_K is the completion of

$$\text{span}\{K(x, \cdot) : x \in \mathcal{X}^2\},$$

and the inner product is defined by

$$\langle f_1, f_2 \rangle = \sum_{i=1}^{k} \sum_{j=1}^{m} \alpha_i \beta_j K(x_i, x_j)$$

for $f_1(\cdot) = \sum_{i=1}^{k} \alpha_i K(x_i, \cdot)$ and $f_2(\cdot) = \sum_{j=1}^{m} \beta_j K(x_j, \cdot)$.

Similarly to SVM in the classification theory our task is to linearly separate (with possibly wide "margin") two sets: $\{(X_i, X_j) : Y_i > Y_j, 1 \leq i \neq j \leq n\}$ and $\{(X_i, X_j) : Y_i < Y_j, 1 \leq i \neq j \leq n\}$, which can be solved using Lagrange multipliers. This primary problem is "transposed" from the $\mathcal{X}^2 \subset \mathbb{R}^{2d}$ to H_K by the function $x \longmapsto K(x, \cdot)$ and by the "kernel trick" we obtain the nonlinear procedure - comprehensive descriptions are in Cortes and Vapnik (1995), Burges (1998), Vapnik (1998) and Blanchard et al. (2008). Finally, we are to minimize the empirical convex risk of the form

$$Q_n(f) = \frac{1}{n(n-1)} \sum_{i \neq j} \max\left[0, 1 - \text{sign}(Y_i - Y_j) f(X_i, X_j)\right] + \lambda ||f||^2$$

in some ball with radius R in the Hilbert space H_K, that is

$$\mathcal{F} = \{f \in H_K : ||f|| \leq R\}$$

and $\lambda > 0$ is a parameter. Consider a Gaussian kernel of the form

$$K(x, x') = \exp\left(-\sigma^2 ||x - x'||_2^2\right),$$

where $x, x' \in \mathcal{X}^2$, $||x||_2 = \sqrt{\sum_{i=1}^{2d} x_i^2}$ and $\sigma > 0$ is a scale parameter. Using Scovel and Steinwart (2007, Theorem 3.1) we obtain that for every compact set $\mathcal{X}$, $\sigma \geq 1$ and $0 < V < 1$

$$\ln N(t, \mathcal{F}, C(\mathcal{X}^2)) \leq C t^{-V} \tag{21}$$

for some constant C dependent on V, d, σ and R. The covering number $N(t, \mathcal{F}, C(\mathcal{X}^2))$ denotes the minimal number of balls with centers in the space of continuous functions on $\mathcal{X}^2$ with the metric $d(f_1, f_2) = \max_{x \in \mathcal{X}^2} |f_1(x) - f_2(x)|$ needed to cover $\mathcal{F}$. This definition differs from ours given in the beginning of Section 2. But Steinwart (2001) proved that H_K corresponding to the Gaussian kernel is dense in $C(\mathcal{X}^2)$, so we can use the property (21) in our studies. Moreover, for every probability

measure μ on $\mathcal{X}^2$ we have $\rho_\mu(f_1,f_2) \le d(f_1,f_2)$, where ρ_μ is defined by (4). Thus, the family $\mathcal{F}$ satisfies Assumption B and is convex. The inequality (5) with $B = \frac{2(2R\lambda+1)^2}{\lambda}$ can be obtained using almost the same arguments as Scovel and Steinwart (2005, Section 6.1). Therefore, we get

$$\mathbb{P}\left(Q(f_n) - Q(f^*) \le C_1 \max\left(\frac{\ln n}{n}, \frac{1}{n^\beta}\right) + C_2 \frac{\ln(C_3/\alpha)}{n}\right) \ge 1-\alpha$$

with $\frac{2}{3} < \beta < 1$.

In the paper we consider ranking estimators that minimize the convex empirical risk. The natural question is: are these estimators also "good" in the case of the primary $0-1$ loss function? Is there any relation between the excess risk and the convex excess risk? Let us introduce, similarly to Clémençon et al. (2008), two notations

$$\rho_+(X_1,X_2) = \mathbb{P}(Y_1 > Y_2 \,|\, X_1, X_2)$$

and

$$\rho_-(X_1,X_2) = \mathbb{P}(Y_1 < Y_2 \,|\, X_1, X_2).$$

It is easy to see that the ranking rule

$$f(x_1,x_2) = 2\,\mathbb{I}_{[\rho_+(x_1,x_2) \ge \rho_-(x_1,x_2)]} - 1$$

minimizes the risk (1) in the class of all measurable functions. Denote $L^* = L(f)$. Let Q^* be the minimal value of $Q(f)$ for every measurable functions $f : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$. Bartlett et al. (2006) proved the relation between the excess risks and the convex excess risk for the classification theory. However, Clémençon et al. (2008) noticed that those results can be applied to ranking. They obtained that for every ranking rule f

$$\gamma(L(f) - L^*) \le Q(f) - Q^*$$

for some invertible function γ that depends on ψ. Moreover, γ can be computed in most interesting cases, for instance: $\gamma(x) = 1 - \sqrt{1-x^2}$ for $\psi(x) = \exp(-x)$.

Divide the difference $Q(f) - Q^*$ into the sum of two terms

$$[Q(f) - Q(f^*)] + [Q(f^*) - Q^*]. \tag{22}$$

The first component in (22), so called "estimation error", tells us how close the risk of f is to the risk of the best element in the class $\mathcal{F}$. The second term ("approximation error") describes how much we lose using the family $\mathcal{F}$. In the paper we study the estimation error, however approximation properties of the family $\mathcal{F}$ are also important problems. For instance they were considered in Cucker and Smale (2002), Lugosi and Vayatis (2004) and Scovel and Steinwart (2007).

3. Experiments

This section is devoted to results of our experiments on real data sets (Frank and Asuncion, 2010). We compare the performance of different SVM's for ranking problems. In Section 2.4 we describe a general method to obtain such procedures, but one can propose some simplification of this idea

that is useful in practice. Consider linearly separable case which means that there exists a vector $\theta \in \mathbb{R}^d$ such that

$$\theta^T X_i > \theta^T X_j \quad \text{for} \quad Y_i > Y_j, \quad 1 \leq i \neq j \leq n.$$

Thus, our task is to assign differences $X_i - X_j$ to classes defined by $\text{sign}(Y_i - Y_j)$. We assume that the distribution of the variable Y is continuous, so $\mathbb{P}(Y_1 = Y_2) = 0$. Therefore, we can use SVM for the classification theory to solve ranking problems if we consider differences of observations in place of observations. Thus, instead of a kernel $\mathcal{K} : \mathcal{X}^2 \times \mathcal{X}^2 \to \mathbb{R}$ we can use a kernel $K : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ if we take

$$\mathcal{K}((x_1, x_2), (x_3, x_4)) = K(x_1 - x_2, x_3 - x_4).$$

The kernel $\mathcal{K}$ is symmetric, continuous and nonnegative definite by the same properties of the kernel K. Therefore, all calculations done by a procedure are made in $\mathbb{R}^d$ instead of $\mathbb{R}^{2d}$. Similar considerations can be found in Herbrich et al. (2000) and Joachims (2006).

To our experiments we use "e1071" package in "R" (R Development Core Team, 2009; Dimitriadou et al., 2010). We choose three types of kernels:

a) linear – $K(x_1, x_2) = <x_1, x_2>_{\mathbb{R}^d}$,

b) polynomial – $K(x_1, x_2) = <x_1, x_2>^3_{\mathbb{R}^d}$,

c) Gaussian – $K(x_1, x_2) = \exp\left(-\frac{1}{2}||x_1 - x_2||^2_{\mathbb{R}^d}\right)$

and two values of the parameter λ: 1 and $\frac{1}{10}$. Less value of λ corresponds to the case when the algorithm should be more adjusted to the sample. Greater value of λ has an effect in wider margin.

We divide every considered data sets into two subsets. The first one is used as a learning sample and we determine an estimator on it. On the second subset we test the estimator, that is, we take two objects and check if the ordering indicated by the estimator is the same as the true one. We repeat the experiment for every data set thirty times and average proportions of wrong decisions are presented in tables below. We denote SVM with the linear kernel and the parameter λ equal to 1 and $\frac{1}{10}$ by L(1) and L(10), respectively. Similarly, W(1) and W(10) stand for polynomial kernels, and G(1) and G(10) for Gaussian kernels.

The first data set concerns experiments that the concrete compressive strength was measured (Yeh, 1998). There are more than 1000 observations, 9 features are considered such that the age of material, contents of water, cement and other ingredients, and finally the concrete compressive strength. In Table 1 we compare errors in predicting the ordering between objects by six algorithms. Notice that in both cases (a learning sample with 100 and 300 elements) SVM with Gaussian kernels

Error	L(1)	L(10)	W(1)	W(10)	G(1)	G(10)
n=100	0,198	0,196	0,199	0,196	0,179	0,185
n=300	0,191	0,189	-	-	0,165	0,179

Table 1: Concrete compressive strength

have least errors, and among them G(1) is better. Proportions of wrong decisions of remaining four algorithms are similar. Besides, for linear and polynomial kernels greater adjustment to the sample has an effect in slightly better effectiveness, contrary to G(1) and G(10). The mark "-" in the

table means that the algorithm did not calculate an estimate for 100 minutes. Comparing to three following data sets it usually happens for polynomial SVM and n=300. Such numerical problems occur, since the number of pairs of instances, that algorithms work with, increases with the square of the sample size n. It makes these procedures inefficient for large n. Some improvements can be found in Joachims (2006).

In the second data set values of houses in the area of Boston are compared (Frank and Asuncion, 2010). Thirteen features were measured, for instance the crime rate, the distance to five Boston employment centres or pupil-teacher ratio by town. Our results are contained in Table 2. We notice

Error	L(1)	L(10)	W(1)	W(10)	G(1)	G(10)
n=100	0,153	0,157	0,148	0,153	0,133	0,132
n=300	0,132	0,133	-	-	0,107	0,123

Table 2: Boston housing data

an improvement of every procedure in recognizing the ordering. Again G(1) and G(10) have least errors. In this case estimators obtained for greater value of the parameter λ (except for G(1)) are better.

Last two experiments are carried out on data sets concerning the quality of red and white wine (Cortez et al., 2009). In both cases one measured 11 features such that the content of alcohol, citric acid, the density and pH. The quality of a wine was determined by wine experts. Results in Table

Red	L(1)	L(10)	W(1)	W(10)	G(1)	G(10)
n=100	0,226	0,227	0,281	0,271	0,257	0,285
n=300	0,214	0,216	-	-	0,232	0,270
White						
n=100	0,265	0,266	0,292	-	0,282	0,305
n=300	0,253	0,249	-	-	0,268	0,303

Table 3: Wine quality

3 indicate lower efficiency of procedures than in previous examples. For red wine as well as white one we can notice the advantage of SVM with linear kernels, whose errors are very similar. The worst algorithm is G(10) which in previous experiments has one of the least error.

Acknowledgments

The research for this paper was partially supported by the grant of Ministry of Science and Higher Education no. N N201 391237.

References

R. Adamczak. Moment inequalities for U-statistics. *Ann. Probab.*, 34:2288–2314, 2007.

S. Agarwal, T. Graepel, R. Herbrich, S. Har-Peled, and D. Roth. Generalization bounds for the area under the ROC curve. *J. Machine Learning Research*, 6:393–425, 2005.

M. A. Arcones and E. Giné. Limit theorems for U-processes. *Ann. Probab.*, 21:1494–1542, 1993.

M. A. Arcones and E. Giné. U-processes indexed by Vapnik-Cervonenkis classes of functions with applications to asymptotics and bootstrap of U-statistics with estimated parameters. *Stochastic Process. Appl.*, 52:17–38, 1994.

P. L. Bartlett, O. Bousquet, and S. Mendelson. Local rademacher complexities. *Ann. Statist.*, 33: 1497–1537, 2005.

P. L. Bartlett, M. I. Jordan, and J. D. McAuliffe. Convexity, classification and risk bounds. *Journal of the American Statistical Association*, 101:138–156, 2006.

G. Blanchard, G. Lugosi, and N. Vayatis. On the rates of convergence of regularized boosting classifiers. *J. Machine Learning Research*, 4:861–894, 2003.

G. Blanchard, O. Bousquet, and P. Massart. Statistical performance of support vector machines. *Ann. Statist.*, 36:489–531, 2008.

A. Bose. Bahadur representation of M_m estimates. *Ann. Statist.*, 26:771–777, 1998.

S. Boucheron, O. Bousquet, and G. Lugosi. Theory of classification: a survey of some recent advances. *ESAIM: P&S*, 9:323–375, 2005.

C. J. C. Burges. A tutorial on support vector machines for pattern recognition. *Data Mining and Knowledge Discovery*, 2:121–167, 1998.

S. Clémençon, G. Lugosi, and N. Vayatis. Ranking and scoring using empirical risk minimization. In *Proceedings of COLT*, pages 1–15, 2005.

S. Clémençon, G. Lugosi, and N. Vayatis. Ranking and empirical minimization of U-statistics. *Ann. Statist.*, 36:844–874, 2008.

C. Cortes and V. N. Vapnik. Support vector networks. *Machine Learning*, 20:273–297, 1995.

P. Cortez, A. Cerdeira, F. Almeida, F. Matos, and J. Reis. Modeling wine preferences by data mining from physicochemical properties. *Decision Support Systems*, 47:547–553, 2009.

D. Cossock and T. Zhang. Subset ranking using regression. In *Proceedings of the 19th Annual Conference on Learning Theory*, 2006.

F. Cucker and S. Smale. On the mathematical foundations of learning. *Bulletin of the American Mathematical Society*, 39:1–49, 2002.

V. H. de la Peña and E. Giné. *Decoupling: From Dependence to Independence*. Springer-Verlag, New York, 1999.

E. Dimitriadou, K. Hornik, F. Leisch, D. Meyer, and A. Weingessel. *e1071: Misc Functions of the Department of Statistics (e1071)*. TU Wien, 2010. URL `http://CRAN.R-project.org/package=e1071`.

A. Frank and A. Asuncion. *UCI Machine Learning Repository*. University of California, Irvine, School of Information and Computer Sciences, 2010. URL `http://archive.ics.uci.edu/ml`.

Y. Freund and R. E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *J. of Computer and System Sciences*, 55:119–139, 1997.

Y. Freund, R. Iyer, R. E. Schapire, and Y. Singer. An efficient boosting algorithm for combining preferences. *J. Machine Learning Research*, 4:933–969, 2004.

R. Herbrich, T. Graepel, and K. Obermayer. *Large Margin Rank Boundaries for Ordinal Regression*, chapter 7. Advanced Large Margin Classifiers. MIT Press, Cambridge, 2000.

W. Hoeffding. A class of statistics with asymptotically normal distribution. *Ann. Math. Statist.*, 19: 293–325, 1948.

T. Joachims. Training linear svms in linear time. In *Proceedings of the ACM KDD*, pages 217–226, 2006.

G. Lugosi and N. Vayatis. On the bayes-risk consistency of regularized boosting methods. *Ann. Statist.*, 32:30–55, 2004.

P. Major. An estimate of the supremum of a nice class of stochastic integrals and U-statistics. *Probab. Theory Related Fields*, 134:489–537, 2006.

P. Massart. Some applications of concentration inequalities to statistics. *Probability theory. Ann. Fac. Sci. Toulouse Math.*, 9:245–303, 2000.

S. Mendelson. Improving the sample complexity using global data. *IEEE Trans. Inform. Theory*, 48:1977–1991, 2002.

S. Mendelson. *A Few Notes on Statistical Learning Theory*, chapter 1. Advanced Lectures in Machine Learning. Springer, 2003.

W. Niemiro. Asymptotics for M-estimators defined by convex minimization. *Ann. Statist.*, 20: 1514–1533, 1992.

W. Niemiro and W. Rejchel. Rank correlation estimators and their limiting distributions. *Statistical Papers*, 50:887–893, 2009.

D. Nolan and D. Pollard. U-processes: rates of convergence. *Ann. Statist.*, 15:780–799, 1987.

A. Pakes and D. Pollard. Simulation and asymptotics of optimization estimators. *Econometrica*, 57:1027–1057, 1989.

D. Pollard. *Convergence of Stochastic Processes*. Springer, New York, 1984.

R Development Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2009. URL `http://www.R-project.org`.

W. Rejchel. Ranking - convex risk minimization. In *Proceedings of WASET*, pages 172–178, 2009.

C. Rudin. Ranking with a p-norm push. In *In Proceedings of the 19th Annual Conference on Learning Theory*, 2006.

C. Scovel and I. Steinwart. Fast rates for support vector machines using Gaussian kernels, 2005. Preprint.

C. Scovel and I. Steinwart. Fast rates for support vector machines using Gaussian kernels. *Ann. Statist.*, 35:575–607, 2007.

R. P. Sherman. The limiting distributions of the maximum rank correlation estimator. *Econometrica*, 61:123–137, 1993.

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

M. Talagrand. Sharper bounds for Gaussian and empirical processes. *Ann. Probab.*, 22:28–76, 1994.

A. W. van der Vaart and J. A. Wellner. *Weak Convergence and Empirical Processes: With Applications to Statistics*. Springer Verlag, New York, 1996.

V. N. Vapnik. *Statistical Learning Theory*. Wiley, New York, 1998.

I. C. Yeh. Modeling of strength of high performance concrete using artificial neural networks. *Cement and Concrete Research*, 28:1797–1808, 1998.

Journal of Machine Learning Research 13 (2012) 1393-1434 Submitted 5/07; Revised 6/11; Published 5/12

Feature Selection via Dependence Maximization

Le Song LSONG@CC.GATECH.EDU
Computational Science and Engineering
Georgia Institute of Technology
266 Ferst Drive
Atlanta, GA 30332, USA

Alex Smola ALEX@SMOLA.ORG
Yahoo! Research
4301 Great America Pky
Santa Clara, CA 95053, USA

Arthur Gretton* ARTHUR.GRETTON@GMAIL.COM
Gatsby Computational Neuroscience Unit
17 Queen Square
London WC1N 3AR, UK

Justin Bedo† JUSTIN.BEDO@NICTA.COM.AU
Statistical Machine Learning Program
National ICT Australia
Canberra, ACT 0200, Australia

Karsten Borgwardt KARSTEN.BORGWARDT@TUEBINGEN.MPG.DE
Machine Learning and Computational Biology Research Group
Max Planck Institutes
Spemannstr. 38
72076 Tübingen, Germany

Editor: Aapo Hyvärinen

Abstract

We introduce a framework for feature selection based on dependence maximization between the selected features and the labels of an estimation problem, using the Hilbert-Schmidt Independence Criterion. The key idea is that good features should be highly dependent on the labels. Our approach leads to a greedy procedure for feature selection. We show that a number of existing feature selectors are special cases of this framework. Experiments on both artificial and real-world data show that our feature selector works well in practice.

Keywords: kernel methods, feature selection, independence measure, Hilbert-Schmidt independence criterion, Hilbert space embedding of distribution

1. Introduction

In data analysis we are typically given a set of observations $X = \{x_1, \ldots, x_m\} \subseteq \mathcal{X}$ which can be used for a number of tasks, such as novelty detection, low-dimensional representation, or a range of

*. Also at Intelligent Systems Group, Max Planck Institutes, Spemannstr. 38, 72076 Tübingen, Germany.
†. Also at the the Australian National University, Canberra, ACT 0200, Australia.

supervised learning problems. In the latter case we also have a set of labels $Y = \{y_1, \ldots, y_m\} \subseteq \mathcal{Y}$ at our disposition. Tasks include ranking, classification, regression, or sequence annotation. While not always true in practice, we assume in the following that the data X and Y are drawn independently and identically distributed (i.i.d.) from some underlying distribution $\Pr(x, y)$.

We often want to reduce the dimension of the data (the number of features) before the actual learning (Guyon and Elisseeff, 2003); a larger number of features can be associated with higher data collection cost, more difficulty in model interpretation, higher computational cost for the classifier, and *sometimes* decreased generalization ability. In other words, there often exist motives in addition to finding a well performing estimator. It is therefore important to select an informative feature subset.

The problem of supervised feature selection can be cast as a combinatorial optimization problem. We have a full set of features, denoted by $\mathcal{S}$ (each element in $\mathcal{S}$ corresponds to one dimension of the data). It is our aim to select a subset $\mathcal{T} \subseteq \mathcal{S}$ such that this subset retains the relevant information contained in X. Suppose the relevance of a feature subset (to the outcome) is quantified by $Q(\mathcal{T})$, and is computed by restricting the data to the dimensions in $\mathcal{T}$. Feature selection can then be formulated as

$$\mathcal{T}_0 = \arg\max_{\mathcal{T} \subseteq \mathcal{S}} Q(\mathcal{T}) \text{ subject to } |\mathcal{T}| \leq t, \tag{1}$$

where $|\cdot|$ computes the cardinality of a set and t is an upper bound on the number of selected features. Two important aspects of problem (1) are the choice of the criterion $Q(\mathcal{T})$ and the selection algorithm.

1.1 Criteria for Feature Selection

A number of quality functionals $Q(\mathcal{T})$ are potential candidates for feature selection. For instance, we could use a mutual information-related quantity or a Hilbert Space-based estimator. In any case, the choice of $Q(\mathcal{T})$ should respect the underlying task. In the case of supervised learning, the goal is to estimate a functional dependence f from training data such that f predicts well on test data. Therefore, a good feature selection criterion should satisfy two conditions:

I: $Q(\mathcal{T})$ is capable of detecting desired (linear or nonlinear) functional dependence between the data and the labels.

II: $Q(\mathcal{T})$ is concentrated with respect to the underlying measure. This guarantees with high probability that detected functional dependence is preserved in test data.

While many feature selection criteria have been explored, not all of them take these two conditions explicitly into account. Examples of criteria that satisfy both conditions include the leave-one-out error bound of SVM (Weston et al., 2000) and the mutual information (Zaffalon and Hutter, 2002). Although the latter has good theoretical justification, it requires density estimation, which is problematic for high dimensional and continuous variables. We sidestep these problems by employing the Hilbert-Schmidt Independence Criterion (HSIC) (Gretton et al., 2005a). Like the mutual information, HSIC is a nonparametric dependence measure, which takes into account all modes of dependence between the variables (not just linear correlation). Unlike some popular mutual information estimates, however, HSIC does not require density estimation as an intermediate step, being based on the covariance between variables mapped to reproducing kernel Hilbert spaces (RKHS).

HSIC has good uniform convergence guarantees, and an unbiased empirical estimate. As we show in Section 2, HSIC satisfies conditions I and II required for $Q(\mathcal{T})$.

1.2 Feature Selection Algorithms

Finding a global optimum for (1) is typically NP-hard (Weston et al., 2003), unless the criterion is easily decomposable or has properties which make approximate optimization easier, for example, submodularity (Nemhauser et al., 1978; Guestrin et al., 2005). Many algorithms transform (1) into a continuous problem by introducing weights on the dimensions (Weston et al., 2000; Bradley and Mangasarian, 1998; Weston et al., 2003; Neal, 1998). These methods perform well for linearly separable problems. For nonlinear problems, however, the optimisation usually becomes non-convex and a local optimum does not necessarily provide good features. Greedy approaches, such as forward selection and backward elimination, are often used to tackle problem (1) directly. Forward selection tries to increase $Q(\mathcal{T})$ as much as possible for each inclusion of features, and backward elimination tries to achieve this for each deletion of features (Guyon et al., 2002). Although forward selection is computationally more efficient, backward elimination provides better features in general since the features are assessed within the context of all others present. See Section 7 for experimental details.

In principle, the Hilbert-Schmidt independence criterion can be employed for feature selection using either a weighting scheme, forward selection or backward selection, or even a mix of several strategies. While the main focus of this paper is on the backward elimination strategy, we also discuss the other selection strategies. As we shall see, several specific choices of kernel function will lead to well known feature selection and feature rating methods. Note that backward elimination using HSIC (BAHSIC) is a filter method for feature selection. It selects features independent of a particular classifier. Such decoupling not only facilitates subsequent feature interpretation but also speeds up the computation over wrapper and embedded methods.

We will see that BAHSIC is directly applicable to binary, multiclass, and regression problems. Most other feature selection methods are only formulated either for binary classification or regression. Multiclass extensions of these methods are usually achieved using a one-versus-the-rest strategy. Still fewer methods handle classification and regression cases at the same time. BAHSIC, on the other hand, accommodates all these cases *and* unsupervised feature selection in a principled way: by choosing different kernels, BAHSIC not only subsumes many existing methods as special cases, but also allows us to define new feature selectors. This versatility is due to the generality of HSIC. The current work is built on earlier presentations by Song et al. (2007b,a). Compared with this earlier work, the present study contains more detailed proofs of the main theorems, proofs of secondary theorems omitted due to space constraints, and a number of additional experiments.

Our paper is structured as follows. In Section 2, we introduce the Hilbert-Schmidt Independence criterion. We provide both biased and unbiased empirical estimates, as well as more efficient approximate empirical estimates. In addition, we prove the empirical estimate converges in probability, and provide its asymptotic distribution. Section 3 contains a brief description of notation for the remainder of the paper. Section 4 presents our two feature selection algorithms, based respectively on forward selection and backwards elimination. Section 5 presents a number of variants of BAHSIC obtained via different kernel choices, with a focus on using the appropriate kernel for the underlying task (e.g., two-class classification, multiclass classification, and regression). Section 6 gives an overview of a variety of feature selection approaches, which can be shown to employ

particular variants of HSIC as their feature relevance criterion. Finally, Sections 7–9 contain our experiments, where we apply HSIC to a number of domains, including real and artificial benchmarks, brain computer interface data, and microarray data.

2. Measures of Dependence

We begin with the simple example of linear dependence detection, and then generalize to the detection of more general kinds of dependence. Consider spaces $\mathcal{X} \subset \mathbb{R}^d$ and $\mathcal{Y} \subset \mathbb{R}^l$, on which we jointly sample observations (x,y) from a distribution $\Pr(x,y)$. Denote by $\mathcal{C}_{xy}$ the covariance matrix

$$\mathcal{C}_{xy} = \mathbb{E}_{xy}\left[xy^\top\right] - \mathbb{E}_x[x]\,\mathbb{E}_y\left[y^\top\right], \tag{2}$$

which contains all second order dependence between the random variables. A statistic that efficiently summarizes the degree of *linear correlation* between x and y is the Frobenius norm of $\mathcal{C}_{xy}$. Given the singular values σ_i of $\mathcal{C}_{xy}$ the norm is defined as

$$\|\mathcal{C}_{xy}\|^2_{\text{Frob}} := \sum_i \sigma_i^2 = \operatorname{tr} \mathcal{C}_{xy}\mathcal{C}_{xy}^\top.$$

This quantity is zero if and only if there exists no *linear dependence* between x and y. This statistic is limited in several respects, however, of which we mention two: first, dependence can exist in forms other than that detectable via covariance (and even when a second order relation exists, the full extent of the dependence between x and y may only be apparent when nonlinear effects are included). Second, the restriction to subsets of $\mathbb{R}^d$ excludes many interesting kinds of variables, such as strings and class labels. In the next section, we generalize the notion of covariance to nonlinear relationships, and to a wider range of data types.

2.1 Hilbert-Schmidt Independence Criterion (HSIC)

In general $\mathcal{X}$ and $\mathcal{Y}$ will be two domains from which we draw samples (x,y): these may be real valued, vector valued, class labels, strings (Lodhi et al., 2002), graphs (Gärtner et al., 2003), dynamical systems (Vishwanathan et al., 2007), parse trees (Collins and Duffy, 2001), images (Schölkopf, 1997), and any other domain on which kernels can be defined. See Schölkopf et al. (2004) and Schölkopf and Smola (2002) for further references.

We define a (possibly nonlinear) mapping $\phi : \mathcal{X} \to \mathcal{F}$ from each $x \in \mathcal{X}$ to a feature space $\mathcal{F}$ (and an analogous map $\psi : \mathcal{Y} \to \mathcal{G}$ wherever needed). In this case we may write the inner product between the features via the positive definite kernel functions

$$k(x,x') := \left\langle \phi(x), \phi(x') \right\rangle \text{ and } l(y,y') := \left\langle \psi(y), \psi(y') \right\rangle.$$

The kernels k and l are associated uniquely with respective reproducing kernel Hilbert spaces $\mathcal{F}$ and $\mathcal{G}$ (although the feature maps ϕ and ψ may not be unique). For instance, if $\mathcal{X} = \mathbb{R}^d$, then this could be as simple as a set of polynomials of order up to b in the components of x, with kernel $k(\mathbf{x},\mathbf{x}') = (\langle \mathbf{x},\mathbf{x}' \rangle + a)^b$. Other kernels, like the Gaussian RBF kernel correspond to infinitely large feature spaces. We need never evaluate these feature representations explicitly, however.

We may now define a cross-covariance operator[1] between these feature maps, in accordance with Baker (1973) and Fukumizu et al. (2004): this is a linear operator $\mathcal{C}_{xy} : \mathcal{G} \longmapsto \mathcal{F}$ such that

$$\mathcal{C}_{xy} := \mathbb{E}_{xy}\left[(\phi(x)-\mu_x)\otimes(\psi(y)-\mu_y)\right] \text{ where } \mu_x = \mathbb{E}_x[\phi(x)] \text{ and } \mu_y = \mathbb{E}_y[\psi(y)].$$

Here $\otimes$ denotes the tensor product. We need to extend the notion of a Frobenius norm to operators. This leads us to the Hilbert-Schmidt norm, which is given by the trace of $\mathcal{C}_{xy}\mathcal{C}_{xy}^\top$. For operators with discrete spectrum this amounts to computing the ℓ_2 norm of the singular values. We use the square of the Hilbert-Schmidt norm of the cross-covariance operator (HSIC), $\|\mathcal{C}_{xy}\|_{\mathrm{HS}}^2$ as our feature selection criterion $Q(\mathcal{T})$. Gretton et al. (2005a) show that HSIC can be expressed in terms of kernels as

$$\begin{aligned}\mathrm{HSIC}(\mathcal{F},\mathcal{G},\Pr_{xy}) &:= \|\mathcal{C}_{xy}\|_{\mathrm{HS}}^2 \qquad (3)\\ =&\mathbb{E}_{xx'yy'}[k(x,x')l(y,y')] + \mathbb{E}_{xx'}[k(x,x')]\mathbb{E}_{yy'}[l(y,y')] - 2\mathbb{E}_{xy}[\mathbb{E}_{x'}[k(x,x')]\mathbb{E}_{y'}[l(y,y')]].\end{aligned}$$

This allows us to compute a measure of dependence between x and y simply by taking expectations over a set of kernel functions k and l with respect to the joint and marginal distributions in x and y *without* the need to perform density estimation (as may be needed for entropy based methods).

2.2 Estimating the Hilbert-Schmidt Independence Criterion

We denote by $Z = (X,Y)$ the set of observations $\{(x_1,y_1),\ldots,(x_m,y_m)\}$ which are drawn *iid* from the joint distribution $\Pr_{xy}$. We denote by $\mathbb{E}_Z$ the expectation with respect Z as drawn from $\Pr_{xy}$. Moreover, $\mathbf{K},\mathbf{L} \in \mathbb{R}^{m\times m}$ are kernel matrices containing entries $\mathbf{K}_{ij} = k(x_i,x_j)$ and $\mathbf{L}_{ij} = l(y_i,y_j)$. Finally, $\mathbf{H} = \mathbf{I} - m^{-1}\mathbf{1}\mathbf{1} \in \mathbb{R}^{m\times m}$ is a centering matrix which projects onto the space orthogonal to the vector $\mathbf{1}$.

Gretton et al. (2005a) derive estimators of $\mathrm{HSIC}(\mathcal{F},\mathcal{G},\Pr_{xy})$ which have $O(m^{-1})$ *bias* and they show that this estimator is well concentrated by means of appropriate tail bounds. For completeness we briefly restate this estimator and its properties below.

Theorem 1 (Biased estimator of HSIC **Gretton et al., 2005a)** *The estimator*

$$\mathrm{HSIC}_0(\mathcal{F},\mathcal{G},Z) := (m-1)^{-2}\,\mathrm{tr}\mathbf{K}\mathbf{H}\mathbf{L}\mathbf{H} \qquad (4)$$

has bias $O(m^{-1})$, that is, $\mathrm{HSIC}(\mathcal{F},\mathcal{G},\Pr_{xy}) - \mathbb{E}_Z\left[\mathrm{HSIC}_0(\mathcal{F},\mathcal{G},Z)\right] = O(m^{-1})$.

This bias arises from the self-interaction terms which are present in HSIC_0, that is, we still have $O(m)$ terms of the form $\mathbf{K}_{ij}\mathbf{L}_{il}$ and $\mathbf{K}_{ji}\mathbf{L}_{li}$ present in the sum, which leads to the $O(m^{-1})$ bias. To address this, we now devise an unbiased estimator which removes those additional terms while ensuring proper normalization. Our proposed estimator has the form

$$\mathrm{HSIC}_1(\mathcal{F},\mathcal{G},Z) := \frac{1}{m(m-3)}\left[\mathrm{tr}(\tilde{\mathbf{K}}\tilde{\mathbf{L}}) + \frac{\mathbf{1}^\top\tilde{\mathbf{K}}\mathbf{1}\mathbf{1}^\top\tilde{\mathbf{L}}\mathbf{1}}{(m-1)(m-2)} - \frac{2}{m-2}\mathbf{1}^\top\tilde{\mathbf{K}}\tilde{\mathbf{L}}\mathbf{1}\right], \qquad (5)$$

where $\tilde{\mathbf{K}}$ and $\tilde{\mathbf{L}}$ are related to $\mathbf{K}$ and $\mathbf{L}$ by $\tilde{\mathbf{K}}_{ij} = (1-\delta_{ij})\mathbf{K}_{ij}$ and $\tilde{\mathbf{L}}_{ij} = (1-\delta_{ij})\mathbf{L}_{ij}$ (i.e., the diagonal entries of $\tilde{\mathbf{K}}$ and $\tilde{\mathbf{L}}$ are set to zero).

1. We abuse the notation here by using the same subscript in the operator $\mathcal{C}_{xy}$ as in the covariance matrix of (2), even though we now refer to the covariance between feature maps.

Theorem 2 (Unbiased estimator of HSIC) *The estimator* HSIC_1 *is unbiased, that is, we have* $\mathbb{E}_Z[\mathrm{HSIC}_1(\mathcal{F},\mathcal{G},Z)] = \mathrm{HSIC}(\mathcal{F},\mathcal{G},\Pr_{xy})$.

Proof We prove the claim by constructing unbiased estimators for each term in (3). Note that we have three types of expectations, namely $\mathbb{E}_{xy}\mathbb{E}_{x'y'}$, a partially decoupled expectation $\mathbb{E}_{xy}\mathbb{E}_{x'}\mathbb{E}_{y'}$, and $\mathbb{E}_x\mathbb{E}_y\mathbb{E}_{x'}\mathbb{E}_{y'}$, which takes all four expectations independently.

If we want to replace the expectations by empirical averages, we need to take care to avoid using the same discrete indices more than once for independent random variables. In other words, when taking expectations over n independent random variables, we need n-tuples of indices where each index occurs exactly once. We define the sets $\mathbf{i}_n^m$ to be the collections of indices satisfying this property. By simple combinatorics one can see that their cardinalities are given by the Pochhammer symbols $(m)_n = \frac{m!}{(m-n)!}$. Jointly drawn random variables, on the other hand, share the same index.

For the joint expectation over pairs we have

$$\mathbb{E}_{xy}\mathbb{E}_{x'y'}\left[k(x,x')l(y,y')\right] = (m)_2^{-1}\mathbb{E}_Z\Big[\sum_{(i,j)\in\mathbf{i}_2^m}\mathbf{K}_{ij}\mathbf{L}_{ij}\Big] = (m)_2^{-1}\mathbb{E}_Z\left[\mathrm{tr}\,\tilde{\mathbf{K}}\tilde{\mathbf{L}}\right]. \tag{6}$$

Recall that we set $\tilde{\mathbf{K}}_{ii} = \tilde{\mathbf{L}}_{ii} = 0$. In the case of the expectation over three independent terms $\mathbb{E}_{xy}\mathbb{E}_{x'}\mathbb{E}_{y'}[k(x,x')l(y,y')]$ we obtain

$$(m)_3^{-1}\mathbb{E}_Z\Big[\sum_{(i,j,q)\in\mathbf{i}_3^m}\mathbf{K}_{ij}\mathbf{L}_{iq}\Big] = (m)_3^{-1}\mathbb{E}_Z\left[\mathbf{1}^\top\tilde{\mathbf{K}}\tilde{\mathbf{L}}\mathbf{1} - \mathrm{tr}\,\tilde{\mathbf{K}}\tilde{\mathbf{L}}\right]. \tag{7}$$

For four independent random variables $\mathbb{E}_x\mathbb{E}_y\mathbb{E}_{x'}\mathbb{E}_{y'}[k(x,x')l(y,y')]$,

$$(m)_4^{-1}\mathbb{E}_Z\Big[\sum_{(i,j,q,r)\in\mathbf{i}_4^m}\mathbf{K}_{ij}\mathbf{L}_{qr}\Big] = (m)_4^{-1}\mathbb{E}_Z\left[\mathbf{1}^\top\tilde{\mathbf{K}}\mathbf{1}\mathbf{1}^\top\tilde{\mathbf{L}}\mathbf{1} - 4\mathbf{1}^\top\tilde{\mathbf{K}}\tilde{\mathbf{L}}\mathbf{1} + 2\,\mathrm{tr}\,\tilde{\mathbf{K}}\tilde{\mathbf{L}}\right]. \tag{8}$$

To obtain an expression for HSIC we only need to take linear combinations using (3). Collecting terms related to $\mathrm{tr}\,\tilde{\mathbf{K}}\tilde{\mathbf{L}}$, $\mathbf{1}^\top\tilde{\mathbf{K}}\tilde{\mathbf{L}}\mathbf{1}$, and $\mathbf{1}^\top\tilde{\mathbf{K}}\mathbf{1}\mathbf{1}^\top\tilde{\mathbf{L}}\mathbf{1}$ yields

$$\mathrm{HSIC}(\mathcal{F},\mathcal{G},\Pr_{xy}) = \frac{1}{m(m-3)}\mathbb{E}_Z\left[\mathrm{tr}\,\tilde{\mathbf{K}}\tilde{\mathbf{L}} + \frac{\mathbf{1}^\top\tilde{\mathbf{K}}\mathbf{1}\mathbf{1}^\top\tilde{\mathbf{L}}\mathbf{1}}{(m-1)(m-2)} - \frac{2}{m-2}\mathbf{1}^\top\tilde{\mathbf{K}}\tilde{\mathbf{L}}\mathbf{1}\right]. \tag{9}$$

This is the expected value of $\mathrm{HSIC}_1[\mathcal{F},\mathcal{G},Z]$. ■

Note that neither HSIC_0 nor HSIC_1 require any explicit regularization parameters, unlike earlier work on kernel dependence estimation. Rather, the regularization is implicit in the choice of the kernels. While in general the biased HSIC is acceptable for estimating dependence, bias becomes a significant problem for diagonally dominant kernels. These occur mainly in the context of sequence analysis such as texts and biological data. Experiments on such data (Quadrianto et al., 2009) show that bias removal is essential to obtain good results.

For suitable kernels $\mathrm{HSIC}(\mathcal{F},\mathcal{G},\Pr_{xy}) = 0$ if and only if x and y are independent. Hence the empirical estimate HSIC_1 can be used to design nonparametric tests of independence. A key feature is that HSIC_1 itself is *unbiased* and its computation is simple. Compare this to quantities based on the mutual information, which requires sophisticated bias correction strategies (e.g., Nemenman et al., 2002).

Previous work used HSIC to *measure* independence between two sets of random variables (Feuerverger, 1993; Gretton et al., 2005a). Here we use it to *select* a subset $\mathcal{T}$ from the first full set of random variables $\mathcal{S}$. We next describe properties of HSIC which support its use as a feature selection criterion.

2.3 HSIC Detects Arbitrary Dependence (Property I)

Whenever $\mathcal{F}, \mathcal{G}$ are RKHSs with characteristic kernels k, l (in the sense of Fukumizu et al., 2008; Sriperumbudur et al., 2008, 2010), then $\mathrm{HSIC}(\mathcal{F}, \mathcal{G}, \Pr_{xy}) = 0$ if and only if x and y are independent.[2] In terms of feature selection, a characteristic kernel such as the Gaussian RBF kernel or the Laplace kernel permits HSIC to detect any dependence between $\mathcal{X}$ and $\mathcal{Y}$. HSIC is zero only if features and labels are independent. Clearly we want to reach the opposite result, namely strong dependence between features and labels. Hence we try to select features that maximize HSIC. Likewise, whenever we want to select a subset of features from X we will try to retain maximal dependence between X and its reduced version.

Note that non-characteristic and non-universal kernels can also be used for HSIC, although they may not guarantee that all dependence is detected. Different kernels incorporate distinctive prior knowledge into the dependence estimation, and they focus HSIC on dependence of a certain type. For instance, a linear kernel requires HSIC to seek only second order dependence, whereas a polynomial kernel of degree b restricts HSIC to test for dependences of degree (up to) b. Clearly HSIC is capable of finding and exploiting dependence of a much more general nature by kernels on graphs, strings, or other discrete domains. We return to this issue in Section 5, where we describe the different kernels that are suited to different underlying classification tasks.

2.4 HSIC is Concentrated (Property II)

HSIC_1, the estimator in (5), can be alternatively formulated using U-statistics (Hoeffding, 1948). This reformulation allows us to derive a uniform convergence bound for HSIC_1. Thus for a given set of features, the feature quality evaluated using HSIC_1 closely reflects its population counterpart HSIC.

Theorem 3 (U-statistic of HSIC) HSIC_1 *can be rewritten in terms of a U-statistic*

$$\mathrm{HSIC}_1(\mathcal{F}, \mathcal{G}, Z) = (m)_4^{-1} \sum_{(i,j,q,r)\in \mathbf{i}_4^m} h(i,j,q,r), \tag{10}$$

where the kernel h of the U-statistic is defined by

$$h(i,j,q,r) = \frac{1}{24} \sum_{(s,t,u,v)}^{(i,j,q,r)} \mathbf{K}_{st}[\mathbf{L}_{st} + \mathbf{L}_{uv} - 2\mathbf{L}_{su}] \tag{11}$$

$$= \frac{1}{6} \sum_{(s\prec t),(u\prec v)}^{(i,j,q,r)} \mathbf{K}_{st}[\mathbf{L}_{st} + \mathbf{L}_{uv}] - \frac{1}{12} \sum_{(s,t,u)}^{(i,j,q,r)} \mathbf{K}_{st}\mathbf{L}_{su}. \tag{12}$$

Here the first sum represents all $4! = 24$ quadruples (s,t,u,v) which can be selected without replacement from (i,j,q,r). Likewise the sum over (s,t,u) is the sum over all triples chosen without replacement. Finally, the sum over $(s \prec t),(u \prec v)$ has the additional condition that the order imposed by (i,j,q,r) is preserved. That is (i,q) and (j,r) are valid pairs, whereas (q,i) or (r,q) are not.

2. This result is more general than the earlier result of Gretton et al. (2005a, Theorem 4), which states that when $\mathcal{F}, \mathcal{G}$ are RKHSs with universal kernels k, l in the sense of Steinwart (2001), on respective *compact* domains $\mathcal{X}$ and $\mathcal{Y}$, then $\mathrm{HSIC}(\mathcal{F}, \mathcal{G}, \Pr_{xy}) = 0$ if and only if x and y are independent. Universal kernels are characteristic on compact domains, however characteristic kernels also exist on non-compact domains.

Proof Combining the three unbiased estimators in (6-8) we obtain a single U-statistic

$$\mathrm{HSIC}_1(\mathcal{F},\mathcal{G},Z) = (m)_4^{-1} \sum_{(i,j,q,r)\in \mathbf{i}_4^m} (\mathbf{K}_{ij}\mathbf{L}_{ij} + \mathbf{K}_{ij}\mathbf{L}_{qr} - 2\mathbf{K}_{ij}\mathbf{L}_{iq}). \tag{13}$$

In this form, however, the kernel $h(i,j,q,r) = \mathbf{K}_{ij}\mathbf{L}_{ij} + \mathbf{K}_{ij}\mathbf{L}_{qr} - 2\mathbf{K}_{ij}\mathbf{L}_{iq}$ is not symmetric in its arguments. For instance $h(i,j,q,r) \neq h(q,j,r,i)$. The same holds for other permutations of the indices. Thus, we replace the kernel with a symmetrized version, which yields

$$h(i,j,q,r) := \frac{1}{4!} \sum_{(s,t,u,v)}^{(i,j,q,r)} (\mathbf{K}_{st}\mathbf{L}_{st} + \mathbf{K}_{st}\mathbf{L}_{uv} - 2\mathbf{K}_{st}\mathbf{L}_{su}) \tag{14}$$

where the sum in (14) represents all ordered quadruples (s,t,u,v) selected without replacement from (i,j,q,r).

This kernel can be simplified, since $\mathbf{K}_{st} = \mathbf{K}_{ts}$ and $\mathbf{L}_{st} = \mathbf{L}_{ts}$. The first one only contains terms $\mathbf{L}_{st}\mathbf{K}_{st}$, hence the indices (u,v) are irrelevant. Exploiting symmetry we may impose $(s \prec t)$ without loss of generality. The same holds for the second term. The third term remains unchanged, which completes the proof. ■

We now show that $\mathrm{HSIC}_1(\mathcal{F},\mathcal{G},Z)$ is concentrated and that it converges to $\mathrm{HSIC}(\mathcal{F},\mathcal{G},\mathrm{Pr}_{xy})$ with rate $1/\sqrt{m}$. The latter is a slight improvement over the convergence of the biased estimator $\mathrm{HSIC}_0(\mathcal{F},\mathcal{G},Z)$, proposed by Gretton et al. (2005a).

Theorem 4 (HSIC is Concentrated) *Assume k,l are bounded almost everywhere by 1, and are non-negative. Then for $m > 1$ and all $\delta > 0$, with probability at least $1-\delta$ for all Pr_{xy}*

$$\left|\mathrm{HSIC}_1(\mathcal{F},\mathcal{G},Z) - \mathrm{HSIC}(\mathcal{F},\mathcal{G},\underset{xy}{\mathrm{Pr}})\right| \leq 8\sqrt{\log(2/\delta)/m}.$$

Proof [Sketch] By virtue of (10) we see immediately that HSIC_1 is a U-statistic of order 4, where each term is contained in $[-2,2]$. Applying Hoeffding's bound for U-statistics as in Gretton et al. (2005a) proves the result. ■

If k and l were just bounded by 1 in terms of absolute value the bound of Theorem 4 would be worse by a factor of 2.

2.5 Asymptotic Normality

Theorem 4 gives *worst case* bounds on the deviation between HSIC and HSIC_1. In many instances, however, an indication of this difference in *typical* cases is needed. In particular, we would like to know the limiting distribution of HSIC_1 for large sample sizes. We now show that HSIC_1 is asymptotically normal, and we derive its variance. These results are also useful since they allow us to formulate statistics for a significance test.

Theorem 5 (Asymptotic Normality) *If $\mathbb{E}[h^2] < \infty$, and data and labels are not independent,*[3] *then as $m \to \infty$, HSIC_1 converges in distribution to a Gaussian random variable with mean*

3. This is a subtle but important point: if the data and labels are independent, then the U-statistic is degenerate, and the null distribution takes a different form. See Gretton et al. (2008) and (Serfling, 1980, Section 5.5).

$\mathrm{HSIC}(\mathcal{F},\mathcal{G},\Pr_{xy})$ *and estimated variance*

$$\sigma^2_{\mathrm{HSIC}_1} = \frac{16}{m}\left(R - \mathrm{HSIC}_1^2\right) \quad \textit{where} \quad R = \frac{1}{m}\sum_{i=1}^{m}\Big((m-1)_3^{-1}\sum_{(j,q,r)\in \mathbf{i}_3^m\setminus\{i\}} h(i,j,q,r)\Big)^2, \tag{15}$$

where $\mathbf{i}_n^m \setminus \{i\}$ *denotes the set of all n-tuples drawn without replacement from* $\{1,\ldots,m\}\setminus\{i\}$.

Proof [Sketch] This follows directly from Serfling (1980, Theorem B, p. 193), which shows asymptotic normality of U-statistics. ■

Unfortunately (15) is expensive to compute by means of an explicit summation: even computing the kernel h of the U-statistic itself is a nontrivial task. For practical purposes we need an expression which can exploit fast matrix operations. As we shall see, $\sigma^2_{\mathrm{HSIC}_1}$ can be computed in $O(m^2)$, given the matrices $\tilde{\mathbf{K}}$ and $\tilde{\mathbf{L}}$. To do so, we first form a vector $\mathbf{h}$ with its ith entry corresponding to $\sum_{(j,q,r)\in\mathbf{i}_3^m\setminus\{i\}} h(i,j,q,r)$. Collecting terms in (11) related to matrices $\tilde{\mathbf{K}}$ and $\tilde{\mathbf{L}}$, $\mathbf{h}$ can be written as

$$\begin{aligned}\mathbf{h} =& (m-2)^2(\tilde{\mathbf{K}}\circ\tilde{\mathbf{L}})\mathbf{1} + (m-2)\Big((\mathrm{tr}\tilde{\mathbf{K}}\tilde{\mathbf{L}})\mathbf{1} - \tilde{\mathbf{K}}\tilde{\mathbf{L}}\mathbf{1} - \tilde{\mathbf{L}}\tilde{\mathbf{K}}\mathbf{1}\Big) - m(\tilde{\mathbf{K}}\mathbf{1})\circ(\tilde{\mathbf{L}}\mathbf{1}) \\ &+ (\mathbf{1}^\top\tilde{\mathbf{L}}\mathbf{1})\tilde{\mathbf{K}}\mathbf{1} + (\mathbf{1}^\top\tilde{\mathbf{K}}\mathbf{1})\tilde{\mathbf{L}}\mathbf{1} - (\mathbf{1}^\top\tilde{\mathbf{K}}\tilde{\mathbf{L}}\mathbf{1})\mathbf{1}\end{aligned}$$

where $\circ$ denotes elementwise matrix multiplication. Then R in (15) can be computed as $R = (4m)^{-1}(m-1)_3^{-2}\mathbf{h}^\top\mathbf{h}$. Combining this with the the unbiased estimator in (5) leads to the matrix computation of $\sigma^2_{\mathrm{HSIC}_1}$.

2.6 Computation

In this section, we first analyze the complexity of computing estimators for Hilbert-Schmidt Independence Criterion. We then propose efficient methods for approximately computing these estimators which are linear in the number of examples.

2.6.1 EXACT COMPUTATION OF HSIC_0 AND HSIC_1

Note that both HSIC_0 and HSIC_1 are simple to compute, since only the kernel matrices $\mathbf{K}$ and $\mathbf{L}$ are needed, and no density estimation is involved. Assume that computing an entry in $\mathbf{K}$ and $\mathbf{L}$ takes constant time, then computing the full matrix takes $O(m^2)$ time. In term of the sample size m, we have the following analysis of the time complexity of HSIC_0 and HSIC_1 (by considering summation and multiplication as atomic operations):

HSIC_0 Centering $\mathbf{L}$ takes $O(m^2)$ time. Since $\mathrm{tr}(\mathbf{KHLH})$ is equivalent to $\mathbf{1}^\top(\mathbf{K}\circ\mathbf{HLH})\mathbf{1}$, it also takes $O(m^2)$ time. Overall, computing HSIC_0 takes $O(m^2)$ time.

HSIC_1 Each of the three terms in HSIC_1, namely $\mathrm{tr}(\tilde{\mathbf{K}}\tilde{\mathbf{L}})$, $\mathbf{1}^\top\tilde{\mathbf{K}}\mathbf{1}\mathbf{1}^\top\tilde{\mathbf{L}}\mathbf{1}$ and $\mathbf{1}^\top\tilde{\mathbf{K}}\tilde{\mathbf{L}}\mathbf{1}$, takes $O(m^2)$ time. Overall, computing HSIC_1 also takes $O(m^2)$ time.

2.6.2 APPROXIMATE COMPUTATION OF HSIC_0 AND HSIC_1

Further speedup is also possible via a low rank approximation of the kernel matrices. Particularly, using incomplete Cholesky decomposition, Gretton et al. (2005a) derive an efficient approximation of HSIC_0. Formally, it can be summarized as the following lemma:

Lemma 6 (Efficient Approximation to HSIC$_0$) *Let* $\mathbf{K} \approx \mathbf{A}\mathbf{A}^\top$ *and* $\mathbf{L} \approx \mathbf{B}\mathbf{B}^\top$, *where* $\mathbf{A} \in \mathbb{R}^{m \times d_f}$ *and* $\mathbf{B} \in \mathbb{R}^{m \times d_g}$. *Then* HSIC$_0$ *can be approximated in* $O(m(d_f^2 + d_g^2))$ *time.*

Note that in this case the dominant computation comes from the incomplete Cholesky decomposition, which can be carried out in $O(md_f^2)$ and $O(md_g^2)$ time respectively (Fine and Scheinberg, 2000).

The three terms in HSIC$_1$ can be computed analogously. Denote by $\mathbf{D_K} = \operatorname{diag}(\mathbf{A}\mathbf{A}^\top)$ and $\mathbf{D_L} = \operatorname{diag}(\mathbf{B}\mathbf{B}^\top)$ the diagonal matrices of the approximating terms. The latter can be computed in $O(md_f)$ and $O(md_g)$ time respectively. We have

$$\mathbf{1}^\top \tilde{\mathbf{K}} \mathbf{1} = \mathbf{1}^\top (\mathbf{A}\mathbf{A}^\top - \mathbf{D_K})\mathbf{1} = \|\mathbf{1}^\top \mathbf{A}\|^2 + \mathbf{1}^\top \mathbf{D_K} \mathbf{1}.$$

Computation requires $O(md_f)$ time. The same holds when computing $\mathbf{1}^\top \tilde{\mathbf{L}} \mathbf{1}$. To obtain the second term we exploit that

$$\mathbf{1}^\top \tilde{\mathbf{K}}\tilde{\mathbf{L}} \mathbf{1} = \mathbf{1}^\top (\mathbf{A}\mathbf{A}^\top - \mathbf{D_K})(\mathbf{B}\mathbf{B}^\top - \mathbf{D_K})\mathbf{1} = ((\mathbf{A}(\mathbf{A}^\top \mathbf{1})) - \mathbf{D_K}\mathbf{1})^\top ((\mathbf{B}(\mathbf{B}^\top \mathbf{1})) - \mathbf{D_L}\mathbf{1}).$$

This can be computed in $O(m(d_f + d_g))$. Finally, to compute the third term we use

$$\begin{aligned} \operatorname{tr} \tilde{\mathbf{K}}\tilde{\mathbf{L}} &= \operatorname{tr}(\mathbf{A}\mathbf{A}^\top - \mathbf{D_K})(\mathbf{B}\mathbf{B}^\top - \mathbf{D_L}) \\ &= \|\mathbf{A}^\top \mathbf{B}\|_{\text{Frob}}^2 - \operatorname{tr} \mathbf{B}^\top \mathbf{D_K} \mathbf{B} - \operatorname{tr} \mathbf{A}^\top \mathbf{D_L} \mathbf{A} + \operatorname{tr} \mathbf{D_K} \mathbf{D_L}. \end{aligned}$$

This can be computed in $O(md_f d_g)$ time. It is the most costly of all operations, since it takes all interactions between the reduced factorizations of $\mathbf{K}$ and $\mathbf{L}$ into account. Hence we may compute HSIC$_1$ efficiently (note again that dominant computation comes from the incomplete Cholesky decomposition):

Lemma 7 (Efficient Approximation of HSIC$_1$**)** *Let* $\mathbf{K} \approx \mathbf{A}\mathbf{A}^\top$ *and* $\mathbf{L} \approx \mathbf{B}\mathbf{B}^\top$, *where* $\mathbf{A} \in \mathbb{R}^{m \times d_f}$ *and* $\mathbf{B} \in \mathbb{R}^{m \times d_g}$. *Then* HSIC$_1$ *can be approximated in* $O(m(d_f^2 + d_g^2))$ *time.*

2.6.3 VARIANCE OF HSIC$_1$

To compute the variance of HSIC$_1$ we also need to deal with $(\tilde{\mathbf{K}} \circ \tilde{\mathbf{L}})\mathbf{1}$. For the latter, no immediate linear algebra expansion is available. However, we may use of the following decomposition. Assume that $\mathbf{a}$ and $\mathbf{b}$ are vectors in $\mathbb{R}^m$. In this case

$$((\mathbf{a}\mathbf{a}^\top) \circ (\mathbf{b}\mathbf{b}^\top))\mathbf{1} = (\mathbf{a} \circ \mathbf{b})(\mathbf{a} \circ \mathbf{b})^\top \mathbf{1}$$

which can be computed in $O(m)$ time. Hence we may compute

$$((\mathbf{A}\mathbf{A}^\top) \circ (\mathbf{B}\mathbf{B}^\top))\mathbf{1} = \sum_{i=1}^{d_f} \sum_{j=1}^{d_g} ((\mathbf{A}_i \circ \mathbf{B}_j)(\mathbf{A}_i \circ \mathbf{B}_j)^\top)\mathbf{1}$$

which can be carried out in $O(md_f d_g)$ time. To take care of the diagonal corrections note that $(\mathbf{A}\mathbf{A}^\top - \mathbf{D_K}) \circ \mathbf{D_L} = \mathbf{0}$. The same holds for $\mathbf{B}$ and $\mathbf{D_K}$. The remaining term $\mathbf{D_K}\mathbf{D_L}\mathbf{1}$ is obviously also computable in $O(m)$ time.

3. Notation

In the following sections, we will deal mainly with vectorial data. Whenever we have vectorial data, we use $\mathbf{X}$ as a shorthand to denote the matrix of all vectorial observations $\mathbf{x}_i \in \mathbb{R}^d$ (the ith row of $\mathbf{X}$ corresponds to $\mathbf{x}_i^\top$). Likewise, whenever the labels can be bundled into a matrix $\mathbf{Y}$ or a vector $\mathbf{y}$ (for binary classification), we will use the latter for a more concise notation. Also, we will refer to the jth column of $\mathbf{X}$ and $\mathbf{Y}$ as $\mathbf{x}_{*j}$ and $\mathbf{y}_{*j}$ respectively as needed.

Furthermore, we denote the mean and standard deviation of the jth feature (dimension) by $x_j = \frac{1}{m}\sum_i^m x_{ij}$ and $s_j = (\frac{1}{m}\sum_i^m (x_{ij}-x_j)^2)^{1/2}$ respectively (x_{ij} is the value of the jth feature of data $\mathbf{x}_i$). For binary classification problems we denote by m_+ and m_- the numbers of positive and negative observations. Moreover, x_{j+} and x_{j-} correspond respectively to the means of the positive and negative classes at the jth feature (the corresponding standard deviations are s_{j+} and s_{j-}). More generally, let m_y be the number of samples with class label equal to y (this notation is also applicable to multiclass problems). Finally, let $\mathbf{1}_n$ be a vector of all ones with length n and $\mathbf{0}_n$ be a vector of all zeros.

For non-vectorial or scalar data, we will use lower case letters to denote them. Very often the labels are scalars, we use y to denote them. The mean and standard deviation of the labels are y and s_y respectively.

4. Feature Selection via HSIC

Having defined our feature selection *criterion*, we now describe *algorithms* that conduct feature selection on the basis of this dependence measure. Denote by $\mathcal{S}$ the full set of features, $\mathcal{T}$ a subset of features ($\mathcal{T} \subseteq \mathcal{S}$). We want to find $\mathcal{T}$ such that the dependence between features in $\mathcal{T}$ and the labels is maximized. Moreover, we may choose between different feature selection strategies, that is, whether we would like to build up a catalog of features in an incremental fashion (forward selection) or whether we would like to remove irrelevant features from a catalog (backward selection). For certain kernels, such as a linear kernel, both selection methods are equivalent: the objective function decomposes into individual coordinates, and thus feature selection can be done without recursion in one go. Although forward selection is computationally more efficient, backward elimination in general yields better features (especially for nonlinear features), since the quality of the features is assessed within the context of all other features (Guyon and Elisseeff, 2003).

4.1 Backward Elimination Using HSIC (BAHSIC)

BAHSIC works by generating a list $\mathcal{S}^\dagger$ which contains the features in increasing degree of relevance. At each step $\mathcal{S}^\dagger$ is appended by a feature from $\mathcal{S}$ which is not contained in $\mathcal{S}^\dagger$ yet by selecting the features which are least dependent on the reference set (i.e., Y or the full set X).

Once we perform this operation, the feature selection problem in (1) can be solved by simply taking the last t elements from $\mathcal{S}^\dagger$. Our algorithm produces $\mathcal{S}^\dagger$ recursively, eliminating the least relevant features from $\mathcal{S}$ and adding them to the end of $\mathcal{S}^\dagger$ at each iteration. For convenience, we also denote HSIC as $\mathrm{HSIC}(\sigma, \mathcal{S})$, where $\mathcal{S}$ are the features used in computing the data kernel matrix $\mathbf{K}$, and σ is the parameter for the data kernel (for instance, this might be the size of a Gaussian kernel $k(\mathbf{x},\mathbf{x}') = \exp(-\sigma\|\mathbf{x}-\mathbf{x}'\|^2)$).

Step 3 of the algorithm denotes a policy for adapting the kernel parameters. Depending on the availability of prior knowledge and the type of preprocessing, we explored three types of policies

1. If we have prior knowledge about the nature of the nonlinearity in the data, we can use a fixed kernel parameter throughout the iterations. For instance, we can use a polynomial kernel of fixed degree, for example, $(\langle \mathbf{x}, \mathbf{x}' \rangle + 1)^2$, to select the features for the XOR data set in Figure 2(a).

2. If we have no prior knowledge, we can optimize HSIC over a set of kernel parameters. In this case, the policy corresponds to $\arg\max_{\sigma \in \Theta} \mathrm{HSIC}(\sigma, \mathcal{S})$, where Θ is a set of parameters that ensure the kernel is bounded. For instance, σ can be the scale parameter of a Gaussian kernel, $k(\mathbf{x}, \mathbf{x}') = \exp(-\sigma \|\mathbf{x} - \mathbf{x}'\|^2)$. Optimizing over the scaling parameter allows us to adapt to the scale of the nonlinearity present in the (feature-reduced) data.

3. Adapting kernel parameters via optimization is computational intensive. Alternatively we can use a policy that produces approximate parameters in each iteration. For instance, if we normalize each feature separately to zero mean and unit variance, we know that the expected value of the distance between data points, $\mathbb{E}\left[(\mathbf{x} - \mathbf{x}')^2\right]$, is $2d$ (d is the dimension of the data). When using a Gaussian kernel, we can then use a policy that assigns σ to $1/(2d)$ as the dimension of the data is reduced.

We now consider in more detail what it means to optimize the kernel. In the case of a radial basis kernel on the observations and a linear kernel on binary labels, the example in Section 5.2 is instructive: optimizing the bandwidth of the kernel k on the observations corresponds to finding the optimum lengthscale for which smooth functions may be found to maximize the *linear* covariance with the labels. This optimum lengthscale will change as the dimensionality of the observation feature space changes (as feature selection progresses). For a related discussion, see (Sriperumbudur et al., 2009, Section 5): in this case, the kernel bandwidth which maximizes a kernel distance measure between two distributions P and Q corresponds to the lengthscale at which P and Q differ. When P is the joint distirbution $P = \Pr(x, y)$, and Q the product of the marginals $Q = \Pr(x)\Pr(y)$, the kernel distance measure in Sriperumbudur et al. (2009) corresponds to HSIC (see Gretton et al., 2007b, Section 7.3). Note further that when a radial basis kernel (such as the Gaussian) is used, the unbiased HSIC_1 is zero both for bandwidth zero, and as the bandwidth approaches infinity (in the former case, the off-diagonal kernel values are zero; in the latter, the off-diagonal kernel values are all equal). Thus HSIC_1 must have a maximum between these two extremes in bandwidth, and this maximum is bounded since the kernel is bounded. Again, see Sriperumbudur et al. (2009) for a related discussion when comparing arbitrary distributions P and Q.

Algorithm 1 BAHSIC

Input: The full set of features $\mathcal{S}$
Output: An ordered set of features $\mathcal{S}^\dagger$

1: $\mathcal{S}^\dagger \leftarrow \varnothing$
2: **repeat**
3: $\quad \sigma \leftarrow \Xi$
4: $\quad I \leftarrow \arg\max_I \ \sum_{j \in I} \mathrm{HSIC}(\sigma, \mathcal{S} \setminus \{j\}), \ \ I \subset \mathcal{S}$
5: $\quad \mathcal{S} \leftarrow \mathcal{S} \setminus I$
6: $\quad \mathcal{S}^\dagger \leftarrow (\mathcal{S}^\dagger, I)$
7: **until** $\mathcal{S} = \varnothing$

Step 4 of the algorithm is concerned with the selection of a set I of features to eliminate. While one could choose a single element of $\mathcal{S}$, this would be inefficient when there are a large number of irrelevant features. On the other hand, removing too many features at once risks the loss of relevant features. In our experiments, we found a good compromise between speed and feature quality was to remove 10% of the current features at each iteration.

In BAHSIC, the kernel matrix $\mathbf{L}$ for the labels is fixed through the whole process. It can be precomputed and stored for speedup if needed. Therefore, the major computation comes from repeated calculation of the kernel matrix $\mathbf{K}$ for the dimension-reduced data. If we remove $1-\beta$ of the data at every step and under the assumption that beyond computing the dot product the actual evaluation of an entry in $\mathbf{K}$ requires only constant time irrespective of the dimension of the data, then the ith iteration of BAHSIC takes $O(\beta^{i-1}dm^2)$ time: d is the total number of features, hence $\beta^{i-1}d$ features remain after $i-1$ iterations and we have m^2 elements in the kernel matrix in total. If we want to reduce the number of features to t we need at most $\tau = \log_\beta(t/d)$ iterations. This brings the total time complexity to $O\left(\frac{1-\beta^\tau}{1-\beta}dm^2\right) = O\left(\frac{d-t}{1-\beta}m^2\right)$ operations. When using incomplete Cholesky factorization we may reduce computational complexity somewhat further to $O\left(\frac{d-t}{1-\beta}m(d_f^2+d_g^2)\right)$ time. This saving is significant as long as $d_f d_g < m$, which may happen, for instance whenever $\mathbf{Y}$ is a binary label matrix. In this case $d_g = 1$, hence incomplete factorizations may yield significant computational gains.

4.2 Forward Selection Using HSIC (FOHSIC)

FOHSIC uses the converse approach to backward selection: it builds a list of features in *decreasing* degree of relevance. This is achieved by adding one feature at a time to the set of features $\mathcal{S}^\dagger$ obtained so far using HSIC as a criterion for the quality of the so-added features. For faster selection of features, we can choose a group of features (for instance, a fixed proportion γ) at step 4 and add them in one shot at step 6. The adaptation of kernel parameters in step 3 follows the same policies as those for BAHSIC. The feature selection problem in (1) can be solved by simply taking the *first* t elements from $\mathcal{S}^\dagger$.

Algorithm 2 FOHSIC

Input: The full set of features $\mathcal{S}$
Output: An ordered set of features $\mathcal{S}^\dagger$

1: $\mathcal{S}^\dagger \leftarrow \varnothing$
2: **repeat**
3: $\quad\sigma \leftarrow \Xi$
4: $\quad I \leftarrow \arg\max_I \ \sum_{j\in I} \mathrm{HSIC}(\sigma, \mathcal{S}^\dagger \cup \{j\}), \ \ I \subset \mathcal{S}$
5: $\quad\mathcal{S} \leftarrow \mathcal{S} \setminus I$
6: $\quad\mathcal{S}^\dagger \leftarrow (\mathcal{S}^\dagger, I)$
7: **until** $\mathcal{S} = \varnothing$

4.2.1 TIME COMPLEXITY

Under the same assumption as BAHSIC, the ith iteration of FOHSIC takes $O((1-\gamma)^{i-1}dm^2)$ time. The total number of iterations τ to obtain t features is $t = [1-(1-\gamma)^\tau]d$, that is $\tau = \frac{\log(d-t)-\log d}{\log(1-\gamma)}$

iterations. Performing τ steps will therefore take $\sum_{i=0}^{\tau-1} d(1-\gamma)^i = d(1-(1-\gamma)^\tau)/\gamma = t/\gamma$ operations. This means that FOHSIC takes $O(tm^2/\gamma)$ time to extract t features.

5. Variants of BAHSIC

So far we discussed a set of algorithms to select features *once* we decided to choose a certain family of kernels k, l to measure dependence between two sets of observations. We now proceed to discussing a number of design choices for k and l. This will happen in two parts: in the current section we discuss generic choices of kernels on data and labels. Various combinations of such kernels will then lead to new algorithms that aim to discover different types of dependence between features and labels (or between a full and a restricted data set we are interested in unsupervised feature selection). After that (in Section 6) we will study specific choices of kernels which correspond to existing feature selection methods.

5.1 Kernels on Data

There exists a great number of kernels on data. Obviously, different kernels will correspond to a range of different assumptions on the type of dependence between the random variables x and y. Hence different kernels induce distinctive similarity measure on the data.

5.1.1 LINEAR KERNEL

The simplest choice for k is to take a linear kernel $k(\mathbf{x}, \mathbf{x}') = \langle \mathbf{x}, \mathbf{x}' \rangle$. This means that we are just using the underlying Euclidean space to define the similarity measure. Whenever the dimensionality d of $\mathbf{x}$ is very high, this may allow for more complexity in the function class than what we could measure and assess otherwise. An additional advantage of this setting is that the kernel decomposes into the sum of products between individual coordinates. This means that any expression of the type $\operatorname{tr}\mathbf{KM}$ can be maximized with respect to the subset of available features via

$$\sum_{j=1}^{d} \mathbf{x}_{*j}^\top \mathbf{M} \mathbf{x}_{*j}.$$

This means that the optimality criterion decomposes into a sum over the scores of individual coordinates. Hence maximization with respect to a subset of size t is trivial, since it just involves finding the t largest contributors. Using (9) we can see that for HSIC_1 the matrix $\mathbf{M}$ is given by

$$\mathbf{M} = \frac{1}{m(m-3)} \left[\tilde{\mathbf{L}} + \left(\mathbf{1}\mathbf{1}^\top - \mathbf{I} \right) \frac{\mathbf{1}^\top \tilde{\mathbf{L}} \mathbf{1}}{(m-1)(m-2)} - \frac{2}{m-2} \left(\tilde{\mathbf{L}} \mathbf{1}\mathbf{1}^\top - \operatorname{diag}\left(\tilde{\mathbf{L}} \mathbf{1} \right) \right) \right].$$

These terms are essentially rank-1 and diagonal updates on $\tilde{\mathbf{L}}$, which means that they can be computed very efficiently. Note also that in this case FOHSIC and BAHSIC generate the *optimal* feature selection with respect to the criterion applied.

5.1.2 POLYNOMIAL KERNEL

Clearly in some cases the use of linear features can be quite limiting. It is possible, though, to use higher order correlations between data for the purpose of feature selection. This is achieved by

using a polynomial kernel

$$k(\mathbf{x},\mathbf{x}') = \left(\langle \mathbf{x},\mathbf{x}'\rangle + a\right)^b \text{ for some } a \geq 0 \text{ and } b \in \mathbb{N}.$$

This kernel incorporates all polynomial interactions up to degree b (provided that $a > 0$). For instance, if we wanted to take only mean and variance into account, we would only need to consider $b = 2$ and $a = 1$. Placing a higher emphasis on means is achieved by increasing the constant offset a.

5.1.3 RADIAL BASIS FUNCTION KERNEL

Note that polynomial kernels only map data into a *finite* dimensional space: while potentially huge, the dimensionality of polynomials of bounded degree is finite, hence criteria arising from such kernels will not provide us with guarantees for a good dependence measure. On the other hand, many radial basis function kernels, such as the Gaussian RBF kernel map $\mathbf{x}$ into an *infinite* dimensional space. One may show that these kernels are in fact characteristic (Fukumizu et al., 2008; Sriperumbudur et al., 2008, 2010). That is, we use kernels of the form

$$k(\mathbf{x},\mathbf{x}') = \kappa(\|\mathbf{x}-\mathbf{x}'\|) \text{ where } \kappa(\xi) = \exp(-\xi) \text{ or } \kappa(\xi) = \exp(-\xi^2)$$

to obtain Laplace and Gaussian kernels respectively. Since the spectrum of the corresponding matrices decays rapidly (Bach and Jordan, 2002, Appendix C), it is easy to compute incomplete Cholesky factorizations of the kernel matrix efficiently.

5.1.4 STRING AND GRAPH KERNEL

One of the key advantages of our approach is that it is not limited to vectorial data. For instance, we can perform feature selection on documents or graphs. For many such situations we have

$$k(x,x') = \sum_{a \sqsubseteq x} w_a \#_a(x) \#_a(x'),$$

where $a \sqsubseteq x$ is a substring of x (Vishwanathan and Smola, 2003; Leslie et al., 2002). Similar decompositions can be made for graphs, where kernels on random walks and paths can be defined. As before, we could use BAHSIC to remove or FOHSIC to generate a list of features such that only relevant ones remain. That said, given that such kernels are additive in their features, we can use the same argument as made above for linear kernels to determine meaningful features in one go.

5.2 Kernels on Labels

The kernels on the data described our inherent assumptions on which properties of x (e.g., linear, polynomial, or nonparametric) are relevant for estimation. We now describe the complementary part, namely a set of possible choices for kernels on labels. Note that these kernels can be just as general as those defined on the data. This means that we may apply our algorithms to classification, regression, Poisson models, ranking, etc., in the same fashion. This is a significant difference to previous approaches which only apply to specialized settings such as binary classification. For completeness we begin with the latter.

5.2.1 BINARY CLASSIFICATION

The simplest kernel we may choose is

$$l(y,y') = yy' \text{ where } y,y' \in \{\pm 1\}. \tag{16}$$

In this case the label kernel matrix $\mathbf{L} = \mathbf{y}\mathbf{y}^\top$ has rank 1 and it is simply the outer product of the vector of labels. Note that we could transform l by adding a positive constant c, such as to obtain $l(y,y') = yy' + c$ which yields $l(y,y') = 2\delta_{y,y'}$ for $c = 1$. This transformation, however, is immaterial: once $\mathbf{K}$ has been centered it is orthogonal to constant matrices.

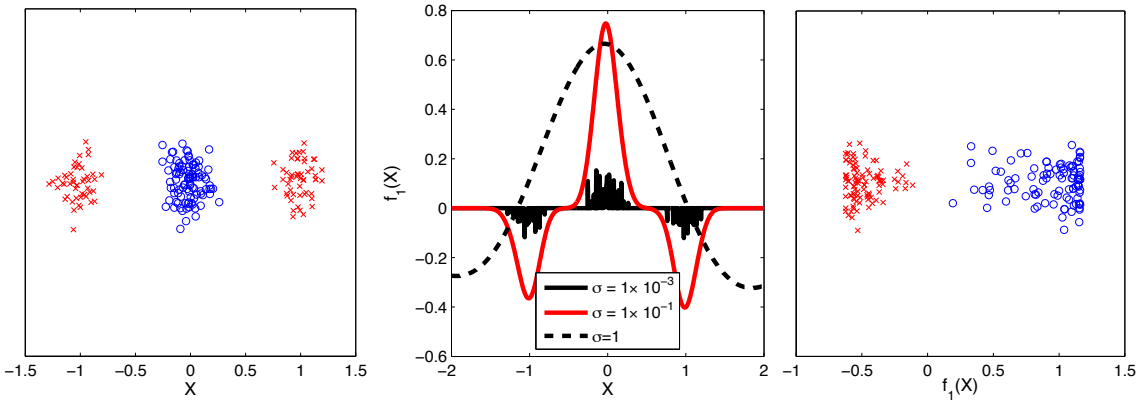
A second transformation, however, leads to nontrivial changes: we may change the relative weights of positive and negative classes. This is achieved by transforming $y \to c_y y$. For instance, we may pick $c_+ = m_+^{-1}$ and $c_- = m_-^{-1}$. That is, we choose

$$\mathbf{y} = \left(m_+^{-1}\mathbf{1}_{m_+}^\top, m_-^{-1}\mathbf{1}_{m_-}^\top\right)^\top \text{ which leads to } l(y,y') = m_y^{-1} m_{y'}^{-1} yy'. \tag{17}$$

That is, we give different weight to positive and negative class according to their sample size. As we shall see in the next section, this corresponds to making the feature selection independent of the class size and it will lead to criteria derived from Maximum Mean Discrepancy estimators (Gretton et al., 2007a).

At this point, it is worth examining in more detail what it means to maximize HSIC in binary classification, as required in Step 3 of Algorithms 1 and 2 (see Section 4). When a linear kernel is used on the observations, HSIC is related to a number of well-established dependence measures, as we will establish in Section 6. Hence, we focus for the moment on the case where the feature space $\mathcal{F}$ for the observations is nonlinear (eg, an RBF kernel), and we use the linear kernel (16) on the labels. HSIC being the squared Hilbert-Schmidt norm of the covariance operator between the feature spaces $\mathcal{F}$ and $\mathcal{G}$, it corresponds to the sum of the squared singular values of this operator. The maximum singular value (COCO; see Gretton et al., 2005b) corresponds to the largest covariance between the mappings $f_1(X)$ and $g_1(Y)$ of X and Y. Given a linear kernel is used on the labels, $g_1(Y)$ will be a linear function on the label space. The nature of $f_1(X)$ will depend on the choice of observation kernel k. For a Gaussian kernel, $f_1(X)$ will be a smooth mapping.

We illustrate this property with a simple toy example in Figure 1. Figure 1(a) plots our observations, where one class has a bimodal distribution in feature X, with cluster centres at ± 1. The second class has a single peak at the origin. The maximum singular vector $f_1(X)$ is shown in Figure 1(b), and is computed using a Gaussian kernel on the observations in accordance with Gretton et al. (2005b). The resulting mapped points in Figure 1(c) have a strong linear relation with the labels (which can only be linearly transformed). Thus, when a nonlinear kernel is used on the observations, the features that maximize HSIC are those that can be smoothly mapped to have a strong linear correlation with the labels. The family of smooth mappings is determined by the choice of kernel on the observations: as we see from Figure 1(b), too large or small a kernel can result in a mapping that does not reflect the lengthscale of the underlying difference in features. This demonstrates the need for the kernel bandwidth selection step described in Section 4.



(a) (b) (c)

Figure 1: Maximum eigenfunction of the covrariance operator. Figure 1(a) contains the original data, where blue points have the label $+1$ and red points are labeled -1. The feature of interest is plotted along the x-axis, and an irrelevant feature on the y-axis. Figure 1(b) contains the largest eigefunction of the covariance operator on the relevant feature alone, for three different kernel sizes: the smallest kernel shows overfitting, and the largest is too smooth. Figure 1(c) contains the mapped points for a "good" kernel choice $\sigma = 0.1$, illustrating a strong linear relation between the mapped points and the labels for this choice of σ.

5.2.2 MULTICLASS CLASSIFICATION

Here we have a somewhat larger choice of options to contend with. Clearly the simplest kernel would be

$$l(y,y') = c_y \delta_{y,y'} \text{ where } c_y > 0. \tag{18}$$

For $c_y = m_y^{-1}$ we obtain a per-class normalization. Clearly, for n classes, the kernel matrix $\mathbf{L}$ can be represented by the outer product of a rank-n matrix, where each row is given by $c_{y_i}\mathbf{e}_{y_i}^\top$, where $\mathbf{e}_y$ denotes the y-th unit vector in $\mathbb{R}^n$. Alternatively, we may adjust the inner product between classes to obtain

$$\begin{aligned} l(y,y') &= \langle \psi(y), \psi(y') \rangle \\ \text{where } \psi(y) &= \mathbf{e}_y \frac{m}{m_y(m-m_y)} - \mathbf{z} \text{ and } \mathbf{z} = ((m-m_1)^{-1}, \ldots, (m-m_n)^{-1})^\top. \end{aligned} \tag{19}$$

This corresponds to assigning a "one versus the rest" feature to each class and taking the inner product between them. As before in the binary case, note that we may drop $\mathbf{z}$ from the expansion, since constant offsets do not change the relative values of HSIC for feature selection. In this case we recover (18) with $c_y = m^2 m_y^{-2}(m-m_y)^{-2}$.

5.2.3 REGRESSION

This is one of the situations where the advantages of using HSIC are clearly apparent: we are able to adjust our method to such situations simply by choosing appropriate kernels. Clearly, we could just use a linear kernel $l(y,y') = yy'$ which would select simple correlations between data and labels.

Another choice is to use an RBF kernel on the labels, such as

$$l(y,y') = \exp\left(-\sigma \left\|y - y'\right\|^2\right). \tag{20}$$

This will ensure that we capture arbitrary nonlinear dependence between $\mathbf{x}$ and y. The price is that in this case $\mathbf{L}$ will have full rank, hence computation of BAHSIC and FOHSIC are correspondingly more expensive.

6. Connections to Other Approaches

We now show that several feature selection criteria are special cases of BAHSIC by choosing appropriate preprocessing of data and kernels. We will directly relate these criteria to the biased estimator HSIC_0 in (4). Given the fact that HSIC_0 converges to HSIC_1 with rate $O(m^{-1})$ it follows that the criteria are well related. Additionally we can infer from this that by using HSIC_1 these other criteria could also be improved by correcting their bias. In summary BAHSIC is capable of finding and exploiting dependence of a much more general nature (for instance, dependence between data and labels with graph and string values).

6.1 Pearson Correlation

Pearson's correlation is commonly used in microarray analysis (van't Veer et al., 2002; Ein-Dor et al., 2006). It is defined as

$$R_j := \frac{1}{m}\sum_{i=1}^{m}\left(\frac{x_{ij}-x_j}{s_{x_j}}\right)\left(\frac{y_i-y}{s_y}\right) \text{ where} \tag{21}$$
$$x_j = \frac{1}{m}\sum_{i=1}^{m} x_{ij} \text{ and } y = \frac{1}{m}\sum_{i=1}^{m} y_i \text{ and } s_{x_j}^2 = \frac{1}{m}\sum_{i=1}^{m}(x_{ij}-x_j)^2 \text{ and } s_y^2 = \frac{1}{m}\sum_{i=1}^{m}(y_i-y)^2.$$

This means that all features are individually centered by x_j and scaled by their coordinate-wise variance s_{x_j} as a preprocessing step. Performing those operations before applying a linear kernel yields the equivalent HSIC_0 formulation:

$$\mathrm{tr}\,\mathbf{KHLH} = \mathrm{tr}\left(\mathbf{X}\mathbf{X}^\top\mathbf{H}\mathbf{y}\mathbf{y}^\top\mathbf{H}\right) = \left\|\mathbf{H}\mathbf{X}^\top\mathbf{H}\mathbf{y}\right\|^2 \tag{22}$$
$$= \sum_{j=1}^{d}\left(\sum_{i=1}^{m}\left(\frac{x_{ij}-x_j}{s_{x_j}}\right)\left(\frac{y_i-y}{s_y}\right)\right)^2 = \sum_{j=1}^{d} R_j^2. \tag{23}$$

Hence HSIC_1 computes the sum of the squares of the Pearson Correlation (pc) coefficients. Since the terms are additive, feature selection is straightforward by picking the list of best performing features.

6.2 Mean Difference and Its Variants

The difference between the means of the positive and negative classes at the jth feature, $(x_{j+} - x_{j-})$, is useful for scoring individual features. With different normalization of the data and the labels, many variants can be derived. In our experiments we compare a number of these variants. For example, the centroid (lin) (Bedo et al., 2006), t-statistic (t), signal-to-noise ratio (snr), moderated t-score (m-t) and B-statistics (lods) (Smyth, 2004) all belong to this family. In the following we make those connections more explicit.

Centroid Bedo et al. (2006) use $v_j := \lambda x_{j+} - (1-\lambda)x_{j-}$ for $\lambda \in (0,1)$ as the score for feature j.[4] Features are subsequently selected according to the absolute value $|v_j|$. In experiments the authors typically choose $\lambda = \frac{1}{2}$.

For $\lambda = \frac{1}{2}$ we can achieve the same goal by choosing $\mathbf{L}_{ii'} = \frac{y_i y_{i'}}{m_{y_i} m_{y_{i'}}}$ ($y_i, y_{i'} \in \{\pm 1\}$), in which case $\mathbf{HLH} = \mathbf{L}$, since the label kernel matrix is already centered. Hence we have

$$\operatorname{tr}\mathbf{KHLH} = \sum_{i,i'=1}^{m} \frac{y_i y_{i'}}{m_{y_i} m_{y_{i'}}} \mathbf{x}_i^\top \mathbf{x}_{i'} = \sum_{j=1}^{d} \left(\sum_{i,i'=1}^{m} \frac{y_i y_{i'} x_{ij} x_{i'j}}{m_{y_i} m_{y_{i'}}} \right) = \sum_{j=1}^{d} (x_{j+} - x_{j-})^2.$$

This proves that the centroid feature selector can be viewed as a special case of BAHSIC in the case of $\lambda = \frac{1}{2}$. From our analysis we see that other values of λ amount to effectively rescaling the patterns $\mathbf{x}_i$ *differently* for different classes, which may lead to undesirable features being selected.

t-Statistic The normalization for the jth feature is computed as

$$s_j = \left[\frac{s_{j+}^2}{m_+} + \frac{s_{j-}^2}{m_-} \right]^{\frac{1}{2}}. \tag{24}$$

In this case we define the t-statistic for the jth feature via $t_j = (x_{j+} - x_{j-})/s_j$.

Compared to the Pearson correlation, the key difference is that now we normalize each feature not by the overall sample standard deviation but rather by a value which takes each of the two classes separately into account.

Signal to noise ratio is yet another criterion to use in feature selection. The key idea is to normalize each feature by $s_j = s_{j+} + s_{j-}$ instead. Subsequently the $(x_{j+} - x_{j-})/s_j$ are used to score features.

Moderated t-score is similar to t-statistic and is used for microarray analysis (Smyth, 2004). Its normalization for the jth feature is derived via a Bayes approach as

$$\tilde{s}_j = \frac{m s_j^2 + m_0 s_0^2}{m + m_0}$$

where s_j is from (24), and s_0 and m_0 are hyperparameters for the prior distribution on s_j (all s_j are assumed to be *iid*). s_0 and m_0 are estimated using information from all feature dimensions.

4. The parameterization in Bedo et al. (2006) is different but it can be shown to be equivalent.

This effectively borrows information from the ensemble of features to aid with the scoring of an individual feature. More specifically, s_0 and m_0 can be computed as (Smyth, 2004)

$$m_0 = 2\Gamma'^{-1}\left(\frac{1}{d}\sum_{j=1}^{d}(z_j - z)^2 - \Gamma'\left(\frac{m}{2}\right)\right), \tag{25}$$
$$s_0^2 = \exp\left(z - \Gamma\left(\frac{m}{2}\right) + \Gamma\left(\frac{m_0}{2}\right) - \ln\left(\frac{m_0}{m}\right)\right)$$

where $\Gamma(\cdot)$ is the gamma function, $'$ denotes derivative, $z_j = \ln(s_j^2)$ and $z = \frac{1}{d}\sum_{j=1}^{d} z_j$.

***B*-statistic** is the logarithm of the posterior odds (lods) that a feature is differentially expressed. Lönnstedt and Speed (2002) and Smyth (2004) show that, for large number of features, B-statistic is given by

$$B_j = a + b\tilde{t}_j^2$$

where both a and b are constant ($b > 0$), and $\tilde{t}_j$ is the moderated-t statistic for the jth feature. Here we see that B_j is monotonic increasing in $\tilde{t}_j$, and thus results in the same gene ranking as the moderated-t statistic.

The reason why these connections work is that the signal-to-noise ratio, moderated t-statistic, and B-statistic are three variants of the t-test. They differ only in their respective denominators, and are thus special cases of HSIC_0 if we normalize the data accordingly.

6.3 Maximum Mean Discrepancy

For binary classification, an alternative criterion for selecting features is to check whether the distributions $\Pr(x|y=1)$ and $\Pr(x|y=-1)$ differ and subsequently pick those coordinates of the data which primarily contribute to the difference between the two distributions.

More specifically, we could use Maximum Mean Discrepancy (MMD) (Gretton et al., 2007a), which is a generalization of mean difference for Reproducing Kernel Hilbert Spaces, given by

$$\mathrm{MMD} = \|\mathbb{E}_x[\phi(x)|y=1] - \mathbb{E}_x[\phi(x)|y=-1]\|_{\mathcal{H}}^2.$$

A biased estimator of the above quantity can be obtained simply by replacing expectations by averages over a finite sample. We relate a biased estimator of MMD to HSIC_0 again by setting m_+^{-1} as the labels for positive samples and $-m_-^{-1}$ for negative samples. If we apply a linear kernel on labels, $\mathbf{L}$ is automatically centered, that is, $\mathbf{L1} = \mathbf{0}$ and $\mathbf{HLH} = \mathbf{L}$. This yields

$$\begin{aligned}\operatorname{tr}\mathbf{KHLH} &= \operatorname{tr}\mathbf{KL} \\ &= \frac{1}{m_+^2}\sum_{i,j}^{m_+} k(x_i,x_j) + \frac{1}{m_-^2}\sum_{i,j}^{m_-} k(x_i,x_j) - \frac{2}{m_+ m_-}\sum_{i}^{m_+}\sum_{j}^{m_-} k(x_i,x_j) \\ &= \left\|\frac{1}{m_+}\sum_{i}^{m_+}\phi(x_i) - \frac{1}{m_-}\sum_{j}^{m_-}\phi(x_j)\right\|_{\mathcal{H}}^2.\end{aligned} \tag{26}$$

The quantity in the last line is an estimator of MMD with bias $O(m^{-1})$ (Gretton et al., 2007a). This implies that HSIC_0 and the biased estimator of MMD are identical up to a constant factor. Since the bias of HSIC_0 is also $O(m^{-1})$, this effectively show that scaled MMD and HSIC_1 converges to each other with rate $O(m^{-1})$.

6.4 Kernel Target Alignment

Alternatively, one could use Kernel Target Alignment (KTA) (Cristianini et al., 2003) to test directly whether there exists any correlation between data and labels. KTA has been used for feature selection in this context. Formally it is defined as $\mathrm{tr}(\mathbf{KL})/\|\mathbf{K}\|\|\mathbf{L}\|$, that is, as the normalized cosine between the kernel matrix and the label matrix.

The nonlinear dependence on $\mathbf{K}$ makes it somewhat hard to optimize for. Indeed, for computational convenience the normalization is often omitted in practice (Neumann et al., 2005), which leaves us with $\mathrm{tr}\,\mathbf{KL}$, the corresponding estimator of MMD.[5] Note the key difference, though, that normalization of $\mathbf{L}$ according to label size does not occur. Nor does KTA take centering into account. Both normalizations are rather important, in particular when dealing with data with very uneven distribution of classes and when using data that is highly collinear in feature space. On the other hand, whenever the sample sizes for both classes are approximately matched, such lack of normalization is negligible and we see that both criteria are similar.

Hence in some cases in binary classification, selecting features that maximizes HSIC also maximizes MMD and KTA. Note that in general (multiclass, regression, or generic binary classification) this connection does not hold. Moreover, the use of HSIC offers uniform convergence bounds on the tails of the distribution of the estimators.

6.5 Shrunken Centroid

The shrunken centroid (pam) method (Tibshirani et al., 2002, 2003) performs feature ranking using the differences from the class centroids to the centroid of all the data, that is

$$(x_{j+} - x_j)^2 + (x_{j-} - x_j)^2,$$

as a criterion to determine the relevance of a given feature. It also scores each feature separately.

To show that this criterion is related to HSIC we need to devise an appropriate map for the labels y. Consider the feature map $\psi(y)$ with $\psi(1) = (m_+^{-1}, 0)^\top$ and $\psi(-1) = (0, m_-^{-1})^\top$. Clearly, when applying $\mathbf{H}$ to $\mathbf{Y}$ we obtain the following centered effective feature maps

$$\psi(1) = (m_+^{-1} - m^{-1}, -m^{-1}) \text{ and } \psi(-1) = (-m^{-1}, m_-^{-1} - m^{-1}).$$

Consequently we may express $\mathrm{tr}\,\mathbf{KHLH}$ via

$$\begin{aligned}
\mathrm{tr}\,\mathbf{KHLH} &= \left\| \frac{1}{m_+} \sum_{i=1}^{m_+} \mathbf{x}_i - \frac{1}{m} \sum_{i=1}^{m} \mathbf{x}_i \right\|^2 + \left\| \frac{1}{m_-} \sum_{i=1}^{m_-} \mathbf{x}_i - \frac{1}{m} \sum_{i=1}^{m} \mathbf{x}_i \right\|^2 && (27) \\
&= \sum_{j=1}^{d} \left(\left(\frac{1}{m_+} \sum_{i=1}^{m_+} x_{ij} - \frac{1}{m} \sum_{i=1}^{m} x_{ij} \right)^2 + \left(\frac{1}{m_-} \sum_{i=1}^{m_-} x_{ij} - \frac{1}{m} \sum_{i=1}^{m} x_{ij} \right)^2 \right) && (28) \\
&= \sum_{j=1}^{d} \left((x_{j+} - x_j)^2 + (x_{j-} - x_j)^2 \right).
\end{aligned}$$

5. The denominator provides a trivial constraint in the case where the *features* are individually normalized to unit norm for a linear kernel, since in this case $\|\mathbf{K}\| = d$: that is, the norm of the kernel matrix scales with the dimensionality d of remaining features in X. The normalization in the denominator can have a more meaningful effect, however, for instance in the taxonomy fitting work of Blaschko and Gretton (2009), where the quality-of-fit score could otherwise be made arbitrarily large independent of the data.

This is the information used by the shrunken centroid method, hence we see that it can be seen to be a special case of HSIC when using a linear kernel on the data and a specific feature map on the labels. Note that we could assign different weights to the two classes, which would lead to a weighted linear combination of distances from the centroid. Finally, it is straightforward how this definition can be extended to multiclass settings, simply by considering the map $\psi : y \to m_y^{-1}\mathbf{e}_y$.

6.6 Ridge Regression

BAHSIC can also be used to select features for regression problems, except that in this case the labels are continuous variables. We could, in principle, use an RBF kernel or similar on the labels to address the feature selection issue. What we show now is that even for a simple linear kernel, interesting results can be obtained. More to the point, we show that feature selection using ridge regression can also be seen to arise as a special case of HSIC feature selection. We assume here that $\mathbf{y}$ is centered.

In ridge regression (Hastie et al., 2001), we estimate the outputs $\mathbf{y}$ using the design matrix $\mathbf{V}$ and a parameter vector $\mathbf{w}$ by minimizing the following regularized risk functional

$$J = \|\mathbf{y} - \mathbf{V}\mathbf{w}\|^2 + \lambda\|\mathbf{w}\|^2.$$

Here the second term is known as the regularizer. If we choose $\mathbf{V} = \mathbf{X}$ we obtain the family of *linear* models. In the general (nonlinear) case $\mathbf{V}$ may be an arbitrary matrix, where each row consists of a set of basis functions, for example, a feature map $\phi(\mathbf{x})$. One might conclude that small values of J correspond to good sets of features, since there a $\mathbf{w}$ with small norm would still lead to a small approximation error. It turns out that J is minimized for $\mathbf{w} = (\mathbf{V}^\top\mathbf{V} + \lambda\mathbf{I})^{-1}\mathbf{y}$. Hence the minimum is given by

$$\begin{aligned} J^* &= \mathbf{y}^\top\mathbf{y} - \mathbf{y}^\top\mathbf{V}(\mathbf{V}^\top\mathbf{V} + \lambda\mathbf{I})^{-1}\mathbf{V}^\top\mathbf{y} \\ &= \text{constant} - \text{tr}\left[\mathbf{V}(\mathbf{V}^\top\mathbf{V} + \lambda\mathbf{I})^{-1}\mathbf{V}^\top\right]\mathbf{y}\mathbf{y}^\top. \end{aligned} \tag{29}$$

Whenever we are only given $\mathbf{K} = \mathbf{V}^\top\mathbf{V}$ we have the following equality

$$J^* = \text{constant} - \text{tr}\left[\mathbf{K}(\mathbf{K} + \lambda\mathbf{I})^{-1}\right]\mathbf{y}\mathbf{y}^\top.$$

This means that the matrices

$$\mathbf{K} := \mathbf{V}(\mathbf{V}^\top\mathbf{V} + \lambda\mathbf{I})^{-1}\mathbf{V}^\top \text{ and } \mathbf{K} := \mathbf{K}(\mathbf{K} + \lambda\mathbf{I})^{-1}$$

are equivalent kernel matrices to be used in BAHSIC. Note that obviously instead of using $\mathbf{y}\mathbf{y}^\top$ as a kernel on the labels $\mathbf{L}$ we could use a nonlinear kernel *in conjunction* with the matrices arrived at from feature selection by ridge regression. It also generalizes the setting of Hastie et al. (2001) to situations other than regression.

6.7 Quadratic Mutual Information

Torr (2003) introduces the quadratic mutual information for feature selection. That is, he uses the L_2 distance between the joint and the marginal distributions on x and y as a criterion for how dependent the two distributions are:

$$I(x,y) = \int\int (\Pr(x,y) - \Pr(x)\Pr(y))^2 \mathrm{d}x\mathrm{d}y. \tag{30}$$

In general, (30) is not efficiently computable. That said, when using a Parzen windows estimate of the joint and the marginals, it is possible to evaluate $I(x,y)$ explicitly. Since we only have a finite number of observations, one uses the estimates

$$\hat{p}(x) = \frac{1}{m}\sum_{i=1}^{m} \kappa_x(x_i - x),$$
$$\hat{p}(y) = \frac{1}{m}\sum_{i=1}^{m} \kappa_y(y_i - y),$$
$$\hat{p}(x,y) = \frac{1}{m}\sum_{i=1}^{m} \kappa_x(x_i - x)\kappa_y(y_i - y).$$

Here κ_x and κ_y are appropriate kernels of the Parzen windows density estimator. Denote by

$$\kappa_{ij} = \int \kappa_x(x_i - x)\kappa_x(x_j - x)dx \;\text{ and }\; \nu_{ij} = \int \kappa_y(y_i - y)\kappa_y(y_j - y)dy$$

inner products between Parzen windows kernels. In this case we have

$$\|\hat{p}(x,y) - \hat{p}(x)\cdot\hat{p}(y)\|^2 = m^{-2}\left[\mathrm{tr}\,\kappa\nu - 2\mathbf{1}^\top\kappa\nu\mathbf{1} + \mathbf{1}^\top\kappa\mathbf{1}\mathbf{1}^\top\nu\mathbf{1}\right] = m^{-2}\kappa\mathbf{H}\nu\mathbf{H}.$$

In other words, we obtain the same criterion as what can be derived from a biased estimator of HSIC. The key difference, though, is that this analogy only works whenever κ and ν can be seen to be arising from an inner product between Parzen windows kernel estimates. This is not universally true: for instance, for graphs, trees, or strings no simple density estimates can be found. This is a serious limitation. Moreover, since we are using a plug-in estimate of the densities, we inherit an innate slow-down of convergence due to the convergence of the density estimators. This issue is discussed in detail in Anderson et al. (1994).

6.8 Recursive Feature Elimination with Support Vectors

Another popular feature selection algorithm is to use Support Vector Machines and to determine the relevance of features by the size of the induced margin as a solution of the dual optimization problem (Guyon et al., 2002). While the connection to BAHSIC is somewhat more tenuous in this context, it is still possible to recast this algorithm in our framework. Before we do so, we describe the basic idea of the method, using ν-SVM instead of plain C-SVMs: for ν-SVM without a constant offset b we have the following dual optimization problem (Schölkopf et al., 1999).

$$\underset{\alpha}{\text{minimize}}\ \frac{1}{2}\alpha^\top(\mathbf{K}\circ\mathbf{L})\alpha \;\text{ subject to }\; \alpha^\top\mathbf{1} = \nu m \text{ and } \alpha_i \in [0,1]. \tag{31}$$

This problem is first solved with respect to α for the full set of features. Features are then selected from (31) by removing coordinates such that the objective function decreases least (if at all). For computational convenience, α is not recomputed for a number of feature removals, since repeated solving of a quadratic program tends to be computationally expensive.

We now show that this procedure can be viewed as a special case of BAHSIC, where now the class of kernels, parameterized by σ is the one of *conformal* kernels. Given a base kernel $k(\mathbf{x},\mathbf{x}')$ Amari and Wu (1999) propose the following kernel:

$$k(\mathbf{x},\mathbf{x}') = \alpha(\mathbf{x})\alpha(\mathbf{x}')k(\mathbf{x},\mathbf{x}') \;\text{ where }\; \alpha(\mathbf{x}) \geq 0.$$

It is easy to see that

$$\alpha^\top (\mathbf{K} \circ \mathbf{L})\alpha = \mathbf{y}^\top [\operatorname{diag}\alpha]\,\mathbf{K}\,[\operatorname{diag}\alpha]\,\mathbf{y} = \mathbf{y}^\top \mathbf{K}\mathbf{y},$$

where $\mathbf{K}$ is the kernel matrix arising from the conformal kernel $k(\mathbf{x},\mathbf{x}')$. Hence for fixed α the objective function is given by a quantity which can be interpreted as a biased version of HSIC. Re-optimization with respect to α is consistent with the kernel adjustment step in Algorithm 1. The only difference being that here the kernel parameters are given by α rather than a kernel width σ. That said, it is also clear from the optimization problem that this style of feature selection may not be as desirable, since the choice of kernel parameters emphasizes only points close to the decision boundary.

7. Experiments

We analyze BAHSIC and related algorithms in an extensive set of experiments. The current section contains results on synthetic and real benchmark data, that is, data from Statlib, the UCI repository, and data from the NIPS feature selection challenge. Sections 8 and 9 then discusses applications to biological data, namely brain signal analysis and feature selection for microarrays.

Since the number of possible choices for feature selection within the BAHSIC family is huge, it is clearly impossible to investigate and compare all of them to all possible other feature selectors. In the present section we pick the following three feature selectors as representative examples. A wider range of kernels and choices is investigated in Section 8 and 9 in the context of biomedical applications.

In this section, we presents three concrete examples of BAHSIC which are used for our later experiments. We apply a Gaussian kernel $k(\mathbf{x},\mathbf{x}') = \exp(-\sigma\|\mathbf{x}-\mathbf{x}'\|^2)$ on data, while varying the kernels on labels. These BAHSIC variants are dedicated respectively to the following settings:

Binary classification (BIN) Use the feature map in (17) and apply a linear kernel.

Multiclass classification (MUL) Use the feature map in (18) and apply a linear kernel.

Regression problem (REG) Use the kernel in (20), that is, a Gaussian RBF kernel on $\mathbf{Y}$.

For the above variants a further speedup of BAHSIC is possible by updating entries in the data kernel matrix incrementally. We use the fact that distance computation of a RBF kernel decomposes into individual coordinates, that is, we use that $\|\mathbf{x}_i - \mathbf{x}_{i'}\|^2 = \sum_{j=1}^{d} \|x_{ij} - x_{i'j}\|^2$. Hence $\|\mathbf{x}_i - \mathbf{x}_{i'}\|^2$ needs to be computed only once, and subsequent updates are effected by subtracting $\|x_{ij} - x_{i'j}\|^2$.

We will use BIN, MUL and REG as the particular instances of BAHSIC in our experiments. We will refer to them commonly as BAHSIC since the exact meaning will be clear depending on the data sets encountered. Furthermore, we also instantiate FOHSIC using the same kernels as BIN, MUL and REG, and we adopt the same convention when we refer to it in our experiments.

7.1 Artificial Data

We constructed 3 artificial data sets, as illustrated in Figure 2, to illustrate the difference between BAHSIC variants with linear and nonlinear kernels. Each data set has 22 dimensions—only the first two dimensions are related to the prediction task and the rest are just Gaussian noise. These data sets are (*i*) **Binary XOR data**: samples belonging to the same class have multimodal distributions;

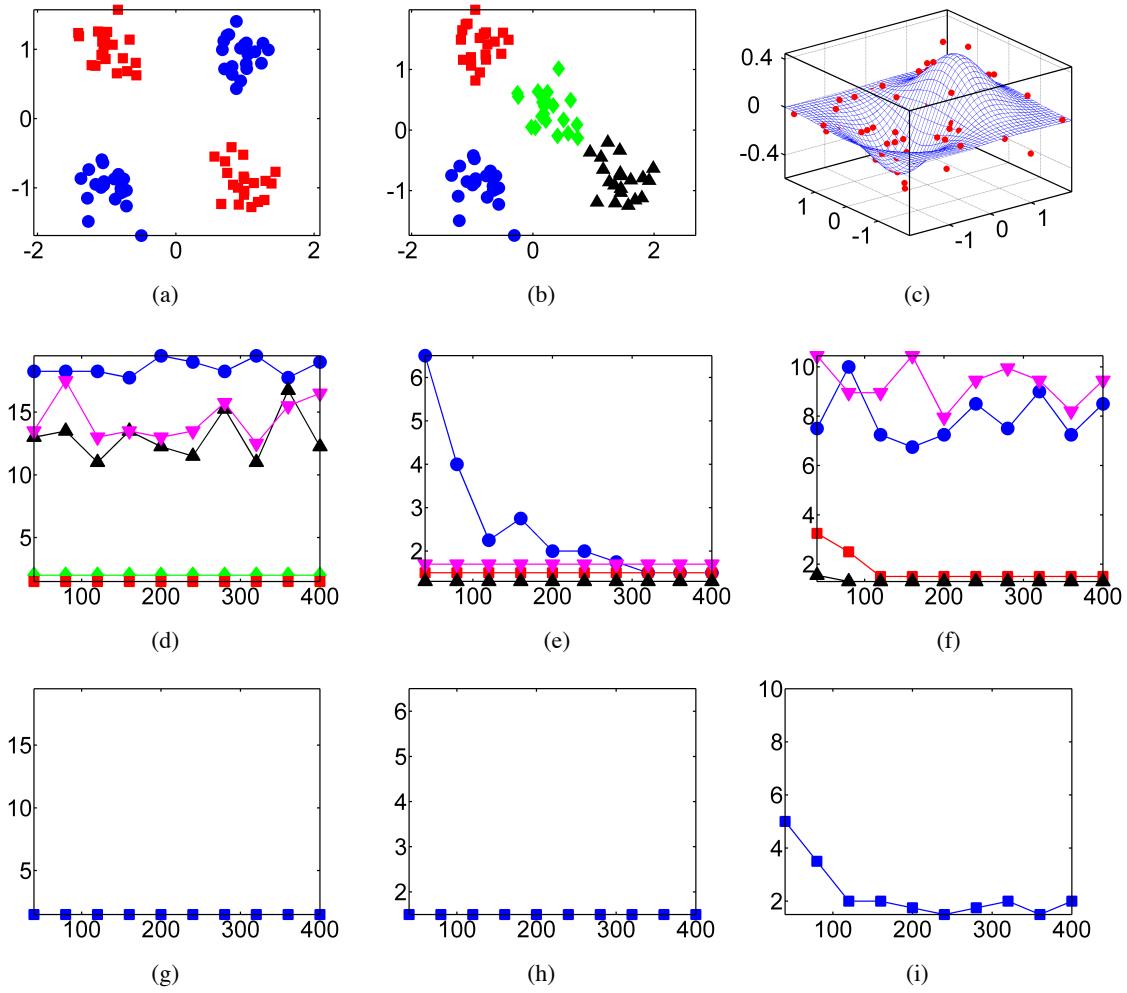


Figure 2: Artificial data sets and the performance of different methods when varying the number of observations. **The first row** contains plots for the first 2 dimension of the (a) binary (b) multiclass and (c) regression data. Different classes are encoded with different colours. **The second row** plots the median rank (y-axis) of the two relevant features as a function of sample size (x-axis) for the corresponding data sets in the first row. **The third row** plots median rank (y-axis) of the two relevant features produced in the first iteration of BAHSIC as a function of the sample size. (Blue circle: Pearson's correlation; Green triangle: RELIEF; Magenta downward triangle: mutual information; Black triangle: FOHSIC; Red square: BAHSIC. Note that RELIEF only works for binary classification.)

(*ii*) **Multiclass data**: there are 4 classes but 3 of them are collinear; (*iii*) **Nonlinear regression data**: labels are related to the first two dimension of the data by $y = x_1 \exp(-x_1^2 - x_2^2) + \varepsilon$, where ε denotes additive Gaussian noise. We compare BAHSIC to FOHSIC, Pearson's correlation, mutual information (Zaffalon and Hutter, 2002), and RELIEF (RELIEF works only for binary problems).

We aim to show that when nonlinear dependencies exist in the data, BAHSIC with nonlinear kernels is very competent in finding them.

We instantiate the artificial data sets over a range of sample sizes (from 40 to 400), and plot the median rank, produced by various methods, for the first two dimensions of the data. All numbers in Figure 2 are averaged over 10 runs. In all cases, BAHSIC shows good performance. More specifically, we observe:

Binary XOR Both BAHSIC and RELIEF correctly select the first two dimensions of the data even for small sample sizes; while FOHSIC, Pearson's correlation, and mutual information fail. This is because the latter three evaluate the goodness of each feature independently. Hence they are unable to capture nonlinear interaction between features.

Multiclass Data BAHSIC, FOHSIC and mutual information select the correct features irrespective of the size of the sample. Pearson's correlation only works for large sample size. The collinearity of 3 classes provides linear correlation between the data and the labels, but due to the interference of the fourth class such correlation is picked up by Pearson's correlation only for a large sample size.

Nonlinear Regression Data The performance of Pearson's correlation and mutual information is slightly better than random. BAHSIC and FOHSIC quickly converge to the correct answer as the sample size increases.

In fact, we observe that as the sample size increases, BAHSIC is able to rank the relevant features (the first two dimensions) almost correctly in the first iteration. In the third row of Figure 2, we show the median rank of the relevant features produced in the first iteration as a function of the sample size. It is clear from the pictures that BAHSIC effectively selects features in a single iteration when the sample size is large enough. For the regression case, we also see that BAHSIC with several iterations, indicated by the red square in Figure 2(f), slightly improves the correct ranking over BAHSIC with a single iteration, given by the blue square in Figure 2(i).

While this does not prove BAHSIC with nonlinear kernels is always better than that with a linear kernel, it illustrates the competence of BAHSIC in detecting nonlinear features. This is obviously useful in a real-world situations. The second advantage of BAHSIC is that it is readily applicable to both classification and regression problems, by simply choosing a different kernel on the labels.

7.2 Public Benchmark Data

In this section, we compare our method, BAHSIC, to several state-of-the-art feature selectors on a large collection of public benchmark datasets. BAHSIC achieves the overall best performance in three experimental settings, *i.e.*, feature selection for binary, multiclass and regression problems.

7.2.1 ALGORITHMS

In this experiment, we show that the performance of BAHSIC can be comparable to other state-of-the-art feature selectors, namely SVM Recursive Feature Elimination (RFE) (Guyon et al., 2002), RELIEF (Kira and Rendell, 1992), L_0-norm SVM (L_0) (Weston et al., 2003), and R2W2 (Weston et al., 2000). We used the implementation of these algorithms as given in the Spider machine learning toolbox, since those were the only publicly available implementations.[6] Furthermore, we

6. The Spider toolbox can be found at `http://www.kyb.tuebingen.mpg.de/bs/people/spider`.

also include filter methods, namely FOHSIC, Pearson's correlation (PC), and mutual information (MI), in our comparisons.

7.2.2 DATA SETS

We used various real world data sets taken from the UCI repository,[7] the Statlib repository,[8] the LibSVM website,[9] and the NIPS feature selection challenge [10] for comparison. Due to scalability issues in Spider, we produced a balanced random sample of size less than 2000 for data sets with more than 2000 samples.

7.2.3 EXPERIMENTAL PROTOCOL

We report the performance of an SVM using a Gaussian kernel on a feature subset of size 5 and 10-fold cross-validation. These 5 features were selected per fold using different methods. Since we are comparing the selected features, we used the same family of classifiers for all methods: an SVM with a Gaussian kernel. To address issues of automatic bandwidth selection (after all, we are interested in adjusting the function class to the data at hand) we chose σ to be the median distance between points in the sample (Schölkopf and Smola, 2002) and we fixed the regularization parameter to $C = 100$. On classification data sets, we measured the performance using the error rate, and on regression data sets we used the percentage of variance *not*-explained (also known as $1-r^2$). The results for binary data sets are summarized in the first part of Table 1. Those for multiclass and regression data sets are reported respectively in the second and the third parts of Table 1.

To provide a concise summary of the performance of various methods on binary data sets, we measured how the methods compare with the best performing one in each data set in Table 1. We recorded the best absolute performance of *all* feature selectors as the baseline, and computed the distance of each algorithm to the best possible result. In this context it makes sense to penalize catastrophic failures more than small deviations. In other words, we would like to have a method which is at least almost always very close to the best performing one. Taking the ℓ_2 distance achieves this effect, by penalizing larger differences more heavily. It is also our goal to choose an algorithm that performs homogeneously well across all data sets. The ℓ_2 distance scores are listed for the binary data sets in Table 1. In general, the smaller the ℓ_2 distance, the better the method. In this respect, BAHSIC and FOHSIC have the best performance. We did not produce the ℓ_2 distance for multiclass and regression data sets, since the limited number of such data sets did not allow us to draw statistically significant conclusions.

Besides using 5 features, we also plot the performance of the learners as a function of the number of selected features for 9 data sets (covertype, ionosphere, sonar, satimage, segment, vehicle, housing, bodyfat and abalone) in Figure 3. Generally speaking, the smaller the plotted number the better the performance of the corresponding learner. For multiclass and regression data sets, it is clear that the curves for BAHSIC very often lie along the lower bound of all methods. For binary classification, however, SVM-RFE as a member of our framework performs the best in general. The advantage of BAHSIC becomes apparent when a small percentage of features is selected. For instance, BAHSIC is the best when only 5 features are selected from data set 1 and 2. Note that

7. UCI repository can be found at `http://www.ics.uci.edu/~mlearn/MLSummary.html`.
8. Statlib repository can be found at `http://lib.stat.cmu.edu/datasets/`.
9. LibSVM can be found at `http://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/`.
10. NIPS feature selection challenge can be found at `http://clopinet.com/isabelle/Projects/NIPS2003/`.

Data	BAHSIC	FOHSIC	PC	MI	RFE	RELIEF	L_0	R2W2
covertype	**26.3±1.5**	37.9±1.7	40.3±1.3	**26.7±1.1**	33.0±1.9	42.7±0.7	43.4±0.7	44.2±1.7
ionosphere	12.3±1.7	12.8±1.6	12.3±1.5	13.1±1.7	20.2±2.2	**11.7±2.0**	35.9±0.4	13.7±2.7
sonar	27.9±3.1	25.0±2.3	25.5±2.4	26.9±1.9	**21.6±3.4**	24.0±2.4	36.5±3.3	32.3±1.8
heart	**14.8±2.4**	**14.4±2.4**	16.7±2.4	15.2±2.5	21.9±3.0	21.9±3.4	30.7±2.8	19.3±2.6
breastcancer	3.8±0.4	3.8±0.4	4.0±0.4	3.5±0.5	**3.4±0.6**	**3.1±0.3**	32.7±2.3	**3.4±0.4**
australian	**14.3±1.3**	**14.3±1.3**	**14.5±1.3**	**14.5±1.3**	**14.8±1.2**	**14.5±1.3**	35.9±1.0	**14.5±1.3**
splice	22.6±1.1	22.6±1.1	22.8±0.9	21.9±1.0	**20.7±1.0**	22.3±1.0	45.2±1.2	24.0±1.0
svmguide3	**20.8±0.6**	**20.9±0.6**	21.2±0.6	**20.4±0.7**	21.0±0.7	21.6±0.4	23.3±0.3	23.9±0.2
adult	24.8±0.2	24.4±0.6	**18.3±1.1**	21.6±1.1	21.3±0.9	24.4±0.2	24.7±0.1	100.0±0.0*
cleveland	**19.0±2.1**	20.5±1.9	21.9±1.7	**19.5±2.2**	20.9±2.1	22.4±2.5	25.2±0.6	21.5±1.3
derm	**0.3±0.3**	**0.3±0.3**	**0.3±0.3**	**0.3±0.3**	**0.3±0.3**	**0.3±0.3**	24.3±2.6	**0.3±0.3**
hepatitis	**13.8±3.5**	15.0±2.5	15.0±4.1	15.0±4.1	15.0±2.5	17.5±2.0	16.3±1.9	17.5±2.0
musk	29.9±2.5	29.6±1.8	**26.9±2.0**	31.9±2.0	34.7±2.5	**27.7±1.6**	42.6±2.2	36.4±2.4
optdigits	**0.5±0.2**	**0.5±0.2**	**0.5±0.2**	3.4±0.6	3.0±1.6	0.9±0.3	12.5±1.7	0.8±0.3
specft	20.0±2.8	20.0±2.8	**18.8±3.4**	**18.8±3.4**	37.5±6.7	26.3±3.5	36.3±4.4	31.3±3.4
wdbc	**5.3±0.6**	**5.3±0.6**	**5.3±0.7**	6.7±0.5	7.7±1.8	7.2±1.0	16.7±2.7	6.8±1.2
wine	**1.7±1.1**	**1.7±1.1**	**1.7±1.1**	**1.7±1.1**	3.4±1.4	4.2±1.9	25.1±7.2	**1.7±1.1**
german	29.2±1.9	29.2±1.8	26.2±1.5	26.2±1.7	27.2±2.4	33.2±1.1	32.0±0.0	**24.8±1.4**
gisette	**12.4±1.0**	**13.0±0.9**	16.0±0.7	50.0±0.0	42.8±1.3	16.7±0.6	42.7±0.7	100.0±0.0*
arcene	**22.0±5.1**	**19.0±3.1**	31.0±3.5	45.0±2.7	34.0±4.5	30.0±3.9	46.0±6.2	32.0±5.5
madelon	**37.9±0.8**	**38.0±0.7**	38.4±0.6	51.6±1.0	41.5±0.8	38.6±0.7	51.3±1.1	100.0±0.0*
ℓ_2	**11.2**	14.8	19.7	48.6	42.2	25.9	85.0	138.3
satimage	**15.8±1.0**	17.9±0.8	52.6±1.7	22.7±0.9	18.7±1.3	-	22.1±1.8	-
segment	28.6±1.3	33.9±0.9	**22.9±0.5**	27.1±1.3	24.5±0.8	-	68.7±7.1	-
vehicle	36.4±1.5	48.7±2.2	42.8±1.4	45.8±2.5	**35.7±1.3**	-	40.7±1.4	-
svmguide2	**22.8±2.7**	**22.2±2.8**	26.4±2.5	27.4±1.6	35.6±1.3	-	34.5±1.7	-
vowel	**44.7±2.0**	**44.7±2.0**	48.1±2.0	45.4±2.2	51.9±2.0	-	85.6±1.0	-
usps	**43.4±1.3**	**43.4±1.3**	73.7±2.2	67.8±1.8	55.8±2.6	-	67.0±2.2	-
housing	**18.5±2.6**	**18.9±3.6**	25.3±2.5	**18.9±2.7**	-	-	-	-
bodyfat	**3.5±2.5**	**3.5±2.5**	**3.4±2.5**	**3.4±2.5**	-	-	-	-
abalone	55.1±2.7	55.9±2.9	**54.2±3.3**	56.5±2.6	-	-	-	-

Table 1: Classification error (%) or percentage of variance *not*-explained (%). The best result, and those results not significantly worse than it, are highlighted in bold (Matlab signrank test with 0.05 significance level). 100.0±0.0*: program is not finished in a week or crashed. -: not applicable.

in these cases, the performance produced by BAHSIC is very close to that using all features. In a sense, BAHSIC is able to shortlist the most informative features.

8. Analysis of Brain Computer Interface Data

In this experiment, we show that BAHSIC selects features that are meaningful in practice. Here we use it to select a frequency band for a brain-computer interface (BCI) data set from the Berlin BCI group (Dornhege et al., 2004). The data contains EEG signals (118 channels, sampled at 100 Hz) from five healthy subjects ('aa', 'al', 'av', 'aw' and 'ay') recorded during two types of motor imaginations. The task is to classify the imagination for individual trials.

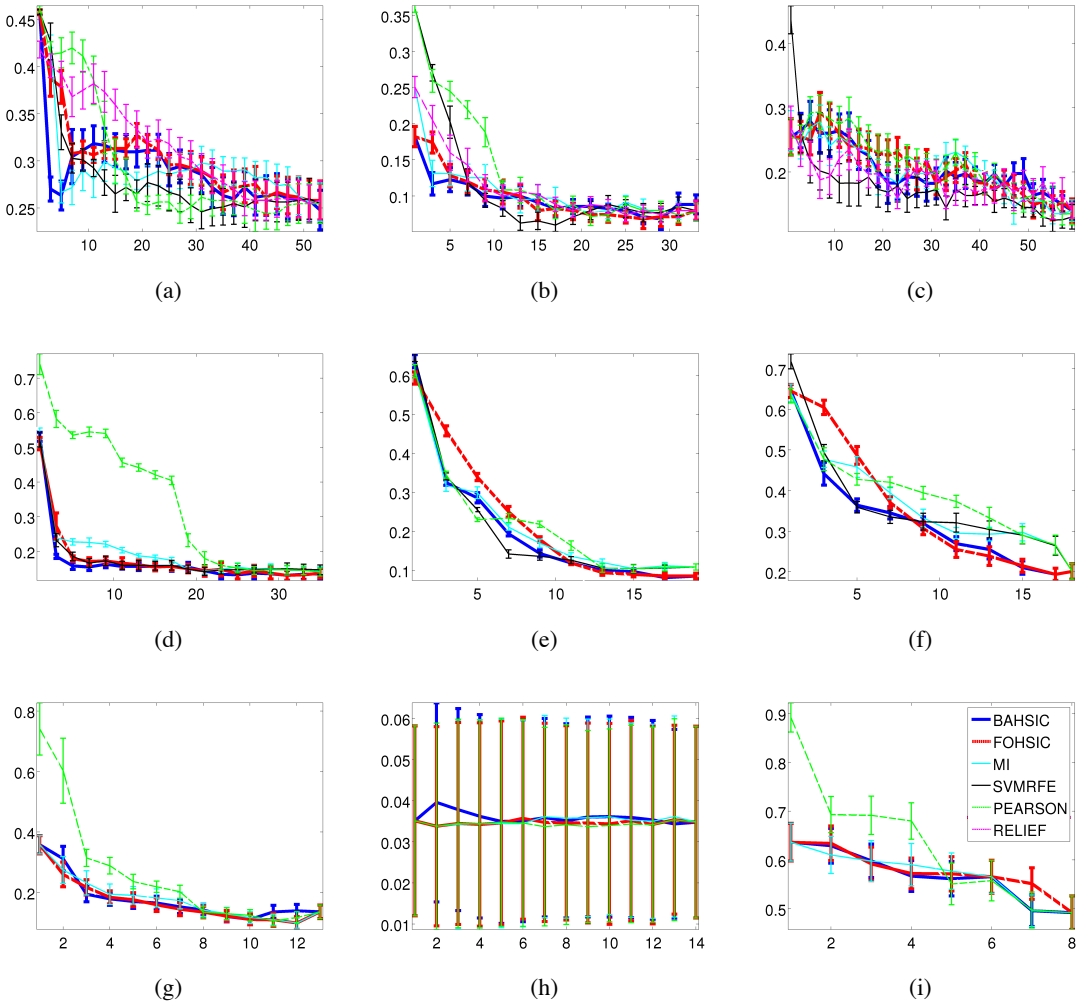


Figure 3: The performance of a classifier or a regressor (vertical axes) as a function of the number of selected features (horizontal axes). Note that the maximum of the horizontal axes are equal to the total number of features in each data set. (a-c) Balanced error rate by a SVM classifier on the binary data sets Covertype (1), Ionosphere (2) and Sonar (3) respectively; (d-f) balanced error rate by a one-versus-the-rest SVM classfier on multiclass data sets Satimage (22), Segment (23) and Vehicle (24) respectively; (g-i) percentage of variance *not*-explained by a SVR regressor on regression data set Housing (25), Body fat (26) and Abalone (27) respectively.

Our experiment proceeds in 3 steps: (*i*) A Fast Fourier transformation (FFT) is performed on each channel and the power spectrum is computed. (*ii*) The power spectra from all channels are averaged to obtain a single spectrum for each trial. (*iii*) BAHSIC is used to select the top 5 discriminative frequency components based on the power spectrum. The 5 selected frequencies and their 4 nearest neighbours are used to reconstruct the temporal signals (with all other Fourier coefficients

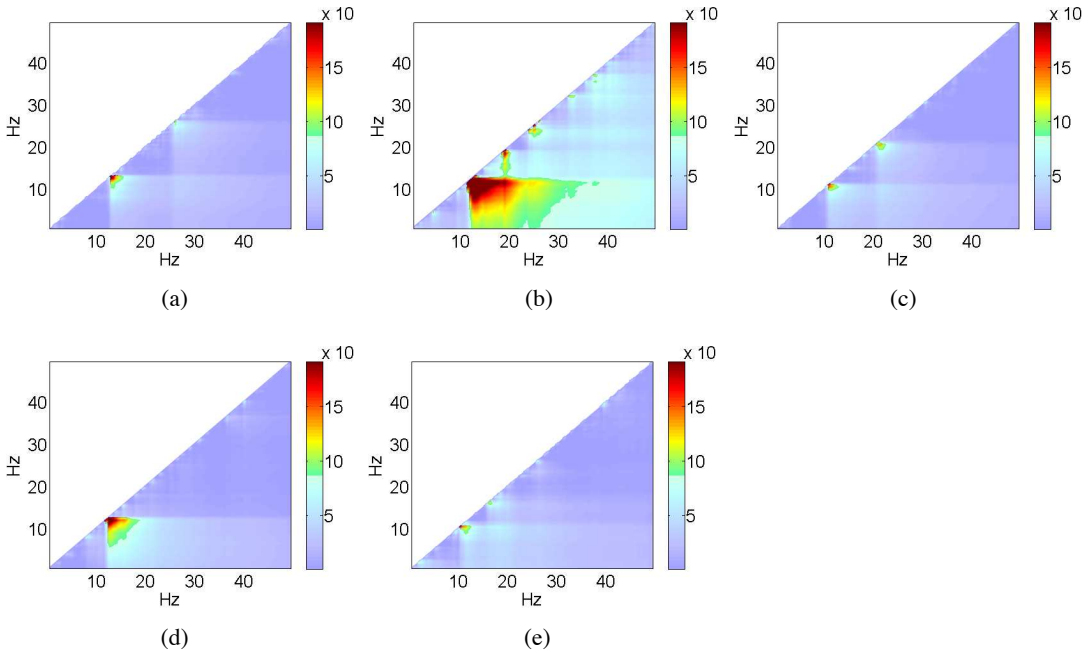


Figure 4: HSIC, encoded by the colour value for different frequency bands. The x-axis corresponds to the upper cutoff and the y-axis denotes the lower cutoff (clearly no signal can be found where the lower bound exceeds the upper bound). Red corresponds to strong dependence, whereas blue indicates that no dependence was found. The figures are for subject (a) 'aa', (b) 'al', (c) 'av', (d) 'aw' and (e) 'ay'.

eliminated). The result is then passed to a normal CSP method (Dornhege et al., 2004) for feature extraction and then classified using a linear SVM.

Automatic filtering using BAHSIC is then compared to other filtering approaches: normal CSP method with manual filtering (8-40 Hz), the CSSP method (Lemm et al., 2005) and the CSSSP method (Dornhege et al., 2006). All results presented in Table 2 are obtained using 50×2-fold cross-validation. Our method is very competitive and obtains the first and second place for 4 of the 5 subjects. While the CSSP and the CSSSP methods are *specialized* embedded methods (w.r.t. the CSP method) for frequency selection on BCI data, our method is entirely generic. BAHSIC decouples feature selection from CSP, while proving competitive.

In Figure 4, we use HSIC to visualize the responsiveness of different frequency bands to motor imagination. The horizontal and the vertical axes in each subfigure represent the lower and upper bounds for a frequency band, respectively. HSIC is computed for each of these bands. Dornhege et al. (2006) report that the μ rhythm (approx. 12 Hz) of EEG is most responsive to motor imagination, and that the β rhythm (approx. 22 Hz) is also responsive. We expect that HSIC will create a strong peak at the μ rhythm and a weaker peak at the β rhythm, and the absence of other responsive frequency components will create block patterns. Both predictions are confirmed in Figure 4. Furthermore, the large area of the red region for subject 'al' indicates good responsiveness of his μ rhythm. This also corresponds well with the lowest classification error obtained for him in Table 2.

Method	aa	al	av	aw	ay
CSP(8-40Hz)	17.5±2.5	3.1±1.2	32.1±2.5	7.3±2.7	**6.0±1.6**
CSSP	14.9±2.9	2.4±1.3	33.0±2.7	**5.4±1.9**	6.2±1.5
CSSSP	**12.2±2.1**	2.2±0.9	31.8±2.8	6.3±1.8	12.7±2.0
BAHSIC	13.7±4.3	**1.9±1.3**	**30.5±3.3**	6.1±3.8	9.0±6.0

Table 2: Classification errors (%) on BCI data after selecting a frequency range.

9. Analysis of Microarray Data

The fact that BAHSIC may be instantiated in numerous ways may create problems for application, that is, it is not immediately clear which criteria we might want to choose. Here we provide guidelines for choosing a specific member of the BAHSIC family by using gene selection as an illustration.

9.1 Data Sets

While some past work focused on analysis of a *specific* single microarray data set we decided to perform a large scale comparison of a raft of techniques on many data sets. We believe that this leads to a more accurate description of the performance of feature selectors. We ran our experiments on 28 data sets, of which 15 are two-class data sets and 13 are multiclass data sets. These data sets are assigned a reference number for convenience. Two-class data sets have a reference number less than or equal to 15, and multiclass data sets have reference numbers of 16 and above. Only one data set, yeast, has feature dimension less than 1000 (79 features). All other data sets have dimensions ranging from approximately 2000 to 25000. The number of samples varies between approximately 50 and 300 samples. A summary of the data sets and their sources is as follows:

- The six data sets studied in Ein-Dor et al. (2006). Three deal with breast cancer (van't Veer et al., 2002; van de Vijver et al., 2002; Wang et al., 2005) (numbered 1, 2 and 3), two with lung cancer (Bhattacharjee et al., 2001; Beer et al., 2002) (4, 5), and one with hepatocellular carcinoma (Iizuka et al., 2003) (6). The B cell lymphoma data set (Rosenwald et al., 2002) is not used because none of the tested methods produce classification errors lower than 40%.

- The six data sets studied in Warnat et al. (2005). Two deal with prostate cancer (Dhanasekaran et al., 2001; Welsh et al., 2001) (7, 8), two with breast cancer (Gruvberger et al., 2001; West, 2003) (9, 10), and two with leukaemia (Bullinger et al., 2004; Valk et al., 2004) (16, 17).

- Five commonly used bioinformatics benchmark data sets on colon cancer (Alon et al., 1999) (11), ovarian cancer (Berchuck et al., 2005) (12), leukaemia (Golub et al., 1999)(13), lymphoma (Alizadeh et al., 2000)(18), and yeast (Brown et al., 2000)(19).

- Nine data sets from the NCBI GEO database. The GDS IDs and reference numbers for this paper are GDS1962 (20), GDS330 (21), GDS531 (14), GDS589 (22), GDS968 (23), GDS1021 (24), GDS1027 (25), GDS1244 (26), GDS1319 (27), GDS1454 (28), and GDS1490 (15), respectively.

9.2 Classification Error and Robustness of Genes

We used stratified 10-fold cross-validation and SVMs to evaluate the predictive performance of the top 10 features selected by various members of BAHSIC. For two-class data sets, a nonlinear SVM with an Gaussian RBF kernel, $k(x,x') = \exp\left(-\frac{\|\mathbf{x}-\mathbf{x}'\|^2}{2\sigma^2}\right)$, was used. The regularization constant C and the kernel width σ were tuned on a grid of $\{0.1,1,10,10^2,10^3\} \times \{1,10,10^2,10^3\}$. Classification performance is measured as the fraction of misclassified samples. For multiclass data sets, all procedures are the same except that we used the SVM in a one-versus-the-rest fashion. A new BAHSIC member are also included in the comparison, with kernels $(\|\mathbf{x}-\mathbf{x}'\|+\varepsilon)^{-1}$ (dis; ε is a small positive number to avoid singularity) on the data.

The classification results for binary and multiclass data sets are reported in Table 3 and Table 4, respectively. In addition to error rate we also report the overlap between the top 10 gene lists created in each fold. The multiclass results are presented separately since some older members of the BAHSIC family, and some competitors, are not naturally extensible to multiclass data sets. From the experiments we make the following observations:

When comparing the overall performance of various gene selection algorithms, it is of primary interest to choose a method which works well *everywhere*, rather than one which sometimes works well and sometimes performs catastrophically. It turns out that the linear kernel (lin) outperforms all other methods in this regard, both for binary and multiclass problems.

To show this, we measure how various methods compare with the best performing one in each data set in Tables 3 and 4. The deviation between algorithms is taken as the square of the difference in performance. This measure is chosen because gene expression data is relative expensive to obtain, and we want an algorithm to select the best genes from them. If an algorithm selects genes that are far inferior to the best possible among all algorithms (catastrophic case), we downgrade the algorithm more heavily. Squaring the performance difference achieves exactly this effect, by penalising larger differences more heavily. In other words, we want to choose an algorithm that performs homogeneously well in all data sets. To provide a concise summary, we add these deviations over the data sets and take the square root as the measure of goodness. These scores (called ℓ_2 distance) are listed in Tables 3 and 4. In general, the smaller the ℓ_2 distance, the better the method. It can been seen that the linear kernel has the smallest ℓ_2 distance on both the binary and multiclass data sets.

9.3 Subtype Discrimination using Nonlinear Kernels

We now investigate why it is that nonlinear kernels (RBF and dis) provide better genes for classification in three data sets from Table 4 (data sets 18 Alizadeh et al., 2000, 27 (GDS1319), and 28 (GDS1454)). These data sets all represent multiclass problems, where at least two of the classes are subtypes with respect to the same supertype.[11] Ideally, the selected genes should contain information discriminating the classes. To visualise this information, we plot in Figure 5 the expression value of the top-ranked gene against that of a second gene ranked in the top 10. This second gene is chosen so that it has minimal correlation with the first gene. We use colours and shapes to distinguish data from different classes (data sets 18 and 28 each contain 3 classes, therefore we use

11. For data set 18, the 3 subtypes are diffuse large B-cell lymphoma and leukemia, follicular lymphoma, and chronic lymphocytic leukemia; For data set 27, the 4 subtypes are various C blastomere mutant embryos: wild type, pie-1, pie-1+pal-1, and mex-3+skn-1; For data set 28, the 3 subtypes are normal cell, IgV unmutated B-cell, and IgV mutated B-cell.

3 different colour and shape combinations for them; data set 27 has 4 classes, so we use 4 such combinations).

We found that genes selected using nonlinear kernels provide better separation between the two classes that correspond to the same supertype (red dots and green diamonds), while the genes selected with the linear kernel do not separate these subtypes well. In the case of data set 27, the increased discrimination between red and green comes at the cost of a greater number of errors in another class (black triangle), however these mistakes are less severe than the errors made between the two subtypes by the linear kernel. This eventually leads to better classification performance for the nonlinear kernels (see Table 4).

The principal characteristic of the data sets is that the blue square class is clearly separated from the rest, while the difference between the two subtypes (red dots and green diamonds) is less clear. The first gene provides information that distinguishes the blue square class, however it provides almost no information about the separation between the two subtypes. The linear kernel does not search for information complementary to the first gene, whereas nonlinear kernels are able to incorporate complementary information. In fact, the second gene that distinguishes the two subtypes (red dots and green diamonds) does not separate all classes. From this gene alone, the blue square class is heavily mixed with other classes. However, combining the two genes together results in better separation between all classes.

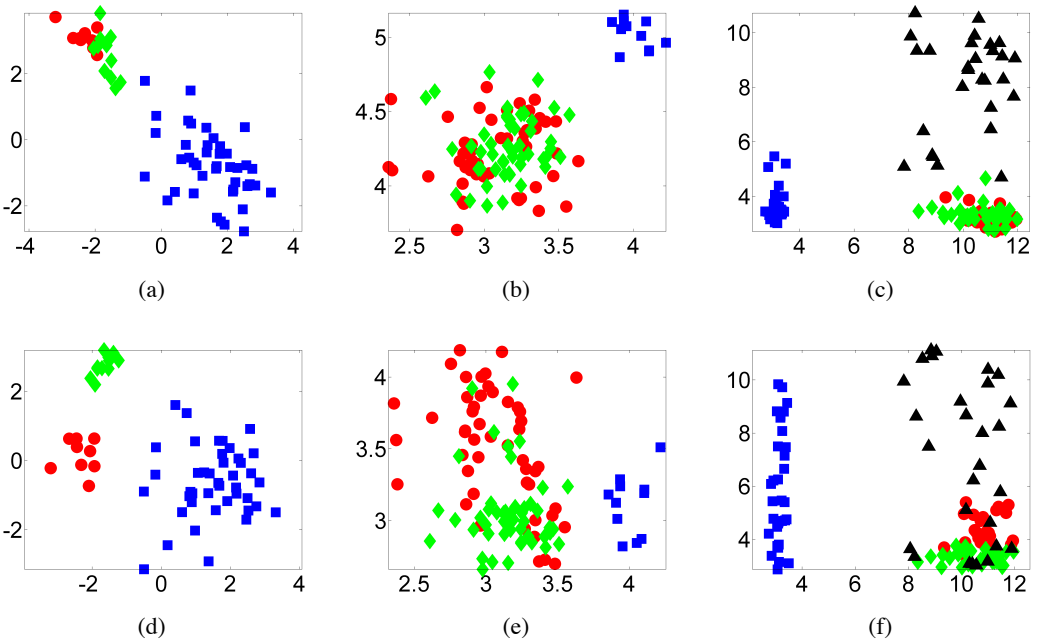


Figure 5: Nonlinear kernels (MUL and dis) select genes that discriminate subtypes (red dots and green diamonds) where the linear kernel fails. The two genes in the first row are representative of those selected by the linear kernel, while those in the second row are produced with a nonlinear kernel for the corresponding data sets. Different colors and shapes represent data from different classes. (a,d) data set 18; (b,e) data set 28; and (e,f) data set 27.

9.4 Rules of Thumb and Implication to Gene Activity

To conclude these experiments, considering the fact that the linear kernel performed best in our feature selection evaluation, yet also taking into account the existence of nonlinear interaction between genes (as demonstrated in Section 9.3), we propose the following two rules of thumb for gene selection:

1. Always apply a linear kernel for general purpose gene selection.
2. Apply a Gaussian kernel if nonlinear effects are present, such as multimodality or complementary effects of different genes.

This result should come as no surprise, due to the high dimensionality of microarray data sets, but we corroborate our claims by means of an extensive experimental evaluation. These experiments also imply a desirable property of gene activity as a whole: it correlates well with the observed outcomes. Multimodal and highly nonlinear situations exist, where a nonlinear feature selector is needed (as can be seen in the outcomes on data sets 18, 27 and 28), yet they occur relatively rarely in practice.

10. Conclusion

This paper provides a *unifying* framework for a raft of feature selection methods. This allows us to give tail bounds and asymptotic expansions for feature selectors. Moreover, we are able to design new feature selectors which work well in practice by means of the Hilbert-Schmidt Independence Criterion (HSIC).

The idea behind the resulting algorithm, BAHSIC, is to choose the feature subset that maximises the dependence between the data and labels. The absence of bias and good convergence properties of the empirical HSIC estimate provide a strong theoretical justification for using HSIC in this context. Although BAHSIC is a filter method, it still demonstrates good performance compared with more specialised methods in both artificial and real world data. It is also very competitive in terms of runtime performance.[12]

A variant of BAHSIC can also be used to perform feature selection for unlabeled data. In this case, we want to select a subset $\mathcal{T}$ of variables such that it is strongly correlated with the full data set. In other words, we want to find a compressed representation of the data itself in the hope that it is useful for a subsequent learning tasks. BAHSIC readily accommodates this by simply using the full data set X as the labels. Clearly, we want to maximize dependence between the selected variables and X without adding many variables which are simply very much correlated to each other. This ingredient is not yet explicitly formulated in the BAHSIC framework. We will investigate this in the future.

Acknowledgments

We thank Vishy Vishwanathan and Bernhard Schölkopf for helpful discussions. NICTA is funded through the Australian Government's *Baking Australia's Ability* initiative, in part through the Australian Research Council.

12. Code is available as part of the Elefant package at `http://elefant.developer.nicta.com.au`.

Data Set	pc	snr	pam	t	m-t	lods	lin	RBF	dis	rfe
1	12.7|3	11.4|3	11.4|**4**	12.9|3	12.9|**4**	12.9|**4**	15.5|3	19.1|1	13.9|2	14.3|0
2	33.2|1	33.9|2	33.9|1	29.5|1	29.5|1	**27.8**|1	32.9|2	31.5|**3**	32.8|2	34.2|0
3	37.4|0	37.4|0	37.4|0	**34.6**|**6**	**34.6**|**6**	**34.6**|**6**	37.4|1	37.4|0	37.4|0	37.4|0
4	41.6|0	38.8|0	41.6|0	40.7|**1**	40.7|0	**37.8**|0	41.6|0	41.6|0	39.7|0	41.6|0
5	27.8|0	**26.7**|0	27.8|0	**26.7**|**2**	**26.7**|**2**	**26.7**|**2**	27.8|0	27.8|0	27.6|0	27.8|0
6	30.0|2	**25.0**|0	31.7|0	**25.0**|**5**	**25.0**|**5**	**25.0**|**5**	30.0|0	31.7|0	30.0|1	30.0|0
7	**2.0**|**6**	**2.0**|5	**2.0**|5	28.7|4	26.3|4	26.3|4	**2.0**|3	**2.0**|4	30.0|0	**2.0**|0
8	3.3|3	**0.0**|4	**0.0**|4	**0.0**|4	3.3|**6**	3.3|**6**	3.3|2	3.3|1	6.7|2	**0.0**|0
9	**10.0**|**6**	**10.0**|**6**	**8.7**|4	34.0|5	37.7|**6**	37.7|**6**	12.0|3	**10.0**|5	12.0|1	**10.0**|0
10	16.0|2	18.0|2	**14.0**|2	**14.0**|8	22.0|**9**	22.0|**9**	16.0|2	16.0|0	18.0|0	32.5|0
11	12.9|5	12.9|5	12.9|5	19.5|0	22.1|0	33.6|0	11.2|4	**9.5**|**6**	16.0|4	19.0|0
12	30.3|2	36.0|2	31.3|2	26.7|**3**	35.7|0	35.7|0	**18.7**|1	35.0|0	33.0|1	29.7|0
13	8.4|5	11.1|0	7.0|5	22.1|3	27.9|**6**	15.4|1	7.0|2	9.6|0	11.1|0	**4.3**|1
14	20.8|1	20.8|1	20.2|0	20.8|**3**	20.8|**3**	20.8|**3**	20.8|0	20.2|0	**19.7**|0	20.8|0
15	**0.0**|7	0.7|1	**0.0**|5	4.0|1	0.7|**8**	0.7|**8**	**0.0**|3	**0.0**|2	2.0|2	**0.0**|1
best	3|2	4|1	5|1	5|6	3|10	5|9	3|0	4|2	1|0	5|0
ℓ_2	16.9	20.9	17.3	43.5	50.5	50.3	**13.2**	22.9	35.4	26.3

Table 3: Two-class data sets: classification error (%) and number of common genes (overlap) for 10-fold cross-validation using the top 10 selected features. Each *row* shows the results for a data set, and each *column* is a method. Each entry in the table contains two numbers separated by "|": the first number is the classification error and the second number is the number of overlaps. For classification error, the best result, and those results not significantly worse than it, are highlighted in bold (Matlab signrank test with 0.05 significance level; a table containing the standard errors is provided in the *supplementary material*). For the overlap, largest overlaps for each data set are highlighted (no significance test is performed). The second last *row* summarises the number of times a method was the best. The last *row* contains the ℓ_2 distance of the error vectors between a method and the best performing method on each data set. We use the following abbreviations: pc - Pearson's correlation, snr - signal-to-noise ratio, pam - shrunken centroid, t - t-statistics, m-t - moderated t-statistics, lods - B-statistics, lin - centroid, dis - $(\|\mathbf{x}-\mathbf{x}'\|+\varepsilon)^{-1}$, rfe - svm recursive feature elimination)

Data	16	17	18	19	20	21	22	23	24	25	26	27	28	**best**	ℓ_2
lin	36.7|1	**0.0**|3	5.0|**3**	10.5|6	35.0|3	**37.5**|**6**	**18.6**|1	**40.3**|**3**	**28.1**|**3**	26.6|**6**	**5.6**|**6**	27.9|7	45.1|1	6|6	**32.4**
RBF	33.3|**3**	5.1|4	**1.7**|**3**	**7.2**|**9**	33.3|0	40.0|1	22.1|0	72.5|0	39.5|0	**24.7**|4	**5.6**|**6**	22.1|**10**	**21.5**|3	5|5	37.9
dis	**29.7**|2	28.8|**5**	6.7|0	8.2|**9**	**29.4**|**7**	38.3|4	43.4|**4**	66.1|0	40.8|0	38.9|4	7.6|1	**8.2**|8	31.6|3	3|4	51.0

Table 4: Multiclass data sets: in this case *columns* are the data sets, and *rows* are the methods. The remaining conventions follow Table 3.

Appendix A. Feature Weighting Using HSIC

Besides the backward elimination algorithm, feature selection using HSIC can also proceed by converting problem (1) into a continuous optimization problem. By adding a penalty on the number of nonzero terms, such as a relaxed ℓ_0 "norm" of a weight vector over the features we are able to solve the problem with continuous optimization methods. Unfortunately, this approach does not perform as well as the the backward elimination procedure proposed in the main text. For completeness and since related methods are somewhat popular in the literature, the approach is described below.

We introduce a weighting $\mathbf{w} \in \mathbb{R}^n$ on the dimensions of the data: $x \longmapsto \mathbf{w} \circ x$, where $\circ$ denotes element-wise product. Thus feature selection using HSIC becomes an optimization problem with respect to $\mathbf{w}$ (for convenience we write HSIC as a function of $\mathbf{w}$, $\mathrm{HSIC}(\mathbf{w})$). To obtain a sparse solution of the selected features, the zero "norm" $\|\mathbf{w}\|_0$ is also incorporated into our objective function (clearly $\|.\|_0$ is not a proper norm). $\|\mathbf{w}\|_0$ computes the number of non-zero entries in $\mathbf{w}$ and the sparsity is achieved by imposing heavier penalty on solutions with large number of non-zero entries. In summary, feature selection using HSIC can be formulated as:

$$\mathbf{w} = \arg\max_{\mathbf{w}} \ \mathrm{HSIC}(\mathbf{w}) - \lambda \|\mathbf{w}\|_0 \ \text{ where } \mathbf{w} \in [0,\infty)^n. \tag{32}$$

The zero "norm" is not a continuous function. However, it can be approximated well by a concave function (Fung et al., 2002) ($\alpha = 5$ works well in practice):

$$\|\mathbf{w}\|_0 \approx \mathbf{1}^\top (\mathbf{1} - \exp -\alpha \mathbf{w}). \tag{33}$$

While the optimization problem in (32) is non-convex, we may use relatively more efficient optimization procedures for the concave approximation of the ℓ_0 norm. For instance, we may use the convex-concave procedure (CCCP) of Yuille and Rangarajan (2003). For a Gaussian kernel HSIC can be decomposed into the sum of a convex and a concave function:

$$\mathrm{HSIC}(\mathbf{w}) - \lambda\|\mathbf{w}\|_0 \approx \mathrm{tr}(\mathbf{K}(\mathbf{I} - m^{-1}\mathbf{1}\mathbf{1}^\top)\mathbf{L}(\mathbf{I} - m^{-1}\mathbf{1}\mathbf{1}^\top)) - \lambda \mathbf{1}^\top(\mathbf{1} - e^{-\alpha \mathbf{w}}).$$

Depending on the choice of $\mathbf{L}$ we need to assign all terms involving exp with positive coefficients into the convex and all terms involving negative coefficients to the concave function.

References

A. Alizadeh, M. Eisen, R. Davis, et al. Distinct types of diffuse large b-cell lymphoma identified by gene expression profiling. *Nature*, 403:503–511, 2000.

U. Alon, N Barkai, D. A. Notterman, K. Gish, S. Ybarra, D. Mack, and A. J. Levine. Broad patterns of gene expression revealed by clustering analysis of tumor and normal colon tissues probed by oligo-nucleotide arrays. In *Proc. Natl. Acad. Sci. USA*, pages 6745–6750, 1999.

Shun-ichi Amari and S. Wu. An information-geometrical method for improving performance of support vector machine classifiers. In D. Willshaw and A. Murray, editors, *Proceedings of ICANN'99*, volume 1, pages 85–90. IEE Press, 1999.

N. Anderson, P. Hall, and D. Titterington. Two-sample test statistics for measuring discrepancies between two multivariate probability density functions using kernel-based density estimates. *Journal of Multivariate Analysis*, 50:41–54, 1994.

F. R. Bach and M. I. Jordan. Kernel independent component analysis. *Journal of Machine Learning Research*, 3:1–48, 2002.

C. Baker. Joint measures and cross-covariance operators. *Transactions of the American Mathematical Society*, 186:273–289, 1973.

J. Bedo, C. Sanderson, and A. Kowalczyk. An efficient alternative to SVM based recursive feature elimination with applications in natural language processing and bioinformatics. In *Artificial Intelligence*, 2006.

D. G. Beer, S. L. Kardia, S. L. Huang, et al. Gene-expression profiles predict survival of patients with lung adenocarcinoma. *Nat. Med.*, 8:816–824, 2002.

A. Berchuck, E. Iversen, and J. Lancaster. Patterns of gene expression that characterize long-term survival in advanced stage serous ovarian cancers. *Clin. Cancer Res.*, 11:3686–3696, 2005.

A. Bhattacharjee, W. G. Richards, W. G. Staunton, et al. Classification of human lung carcinomas by mrna expression profiling reveals distinct adenocarcinoma subclasses. *Proc. Natl. Acad. Sci.*, 98:13790–13795, 2001.

M. Blaschko and A. Gretton. Learning taxonomies by dependence maximization. In *Advances in Neural Information Processing Systems 21*, pages 153–160. MIT Press, 2009.

P. S. Bradley and O. L. Mangasarian. Feature selection via concave minimization and support vector machines. In J. Shavlik, editor, *Proc. Intl. Conf. Machine Learning*, pages 82–90, San Francisco, California, 1998. Morgan Kaufmann Publishers. ftp://ftp.cs.wisc.edu/math-prog/tech-reports/98-03.ps.Z.

M. Brown, W. Grundy, D. Lin, N. Cristianini, C. Sugnet, T. Furey, M. Ares, and D. Haussler. Knowledge-based analysis of microarray gene expression data by using support vector machines. *Proc. Natl. Acad. Sci.*, 97:262–267, 2000.

L. Bullinger, K. Dohner, E. Bair, S. Frohling, R. F. Schlenk, R. Tibshirani, H. Dohner, and J. R. Pollack. Use of gene-expression profiling to identify prognostic subclasses in adult acute myeloid leukemia. *New England Journal of Medicine*, 350(16):1605–1616, Apr 2004.

M. Collins and N. Duffy. Convolution kernels for natural language. In T. G. Dietterich, S. Becker, and Z. Ghahramani, editors, *Advances in Neural Information Processing Systems 14*, pages 625–632, Cambridge, MA, 2001. MIT Press.

N. Cristianini, J. Kandola, A. Elisseeff, and J. Shawe-Taylor. On optimizing kernel alignment. Technical report, UC Davis Department of Statistics, 2003.

S. M. Dhanasekaran, T. R. Barrette, D. Ghosh, R. Shah, S. Varambally, K. Kurachi, K. J. Pienta, M. A. Rubin, and A. M. Chinnaiyan. Delineation of prognostic biomarkers in prostate cancer. *Nature*, 412(6849):822–826, Aug 2001.

G. Dornhege, B. Blankertz, G. Curio, and K. Müller. Boosting bit rates in non-invasive EEG single-trial classifications by feature combination and multi-class paradigms. *IEEE Trans. Biomed. Eng.*, 51:993–1002, 2004.

G. Dornhege, B. Blankertz, M. Krauledat, F. Losch, G. Curio, and K. Müller. Optimizing spatio-temporal filters for improving BCI. In *Advances in Neural Information Processing Systems 18*, 2006.

L. Ein-Dor, O. Zuk, and E. Domany. Thousands of samples are needed to generate a robust gene list for predicting outcome in cancer. *Proc. Natl. Acad. Sci. USA*, 103(15):5923–5928, Apr 2006.

Andrey Feuerverger. A consistent test for bivariate dependence. *International Statistical Review*, 61(3):419–433, 1993.

S. Fine and K. Scheinberg. Efficient SVM training using low-rank kernel representation. Technical report, IBM Watson Research Center, New York, 2000.

K. Fukumizu, F. R. Bach, and M. I. 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 *Advances in Neural Information Processing Systems 20*, pages 489–496, Cambridge, MA, 2008. MIT Press.

G. Fung, O. L. Mangasarian, and A. J. Smola. Minimal kernel classifiers. *Journal of Machine Learning Research*, 3:303–321, 2002.

T. Gärtner, P.A. Flach, and S. Wrobel. On graph kernels: Hardness results and efficient alternatives. In B. Schölkopf and M. K. Warmuth, editors, *Proc. Annual Conf. Computational Learning Theory*, pages 129–143. Springer, 2003.

T. R. Golub, D. K. Slonim, P. Tamayo, C. Huard, M. Gaasenbeek, J. P. Mesirov, H. Coller, M. L. Loh, J. R. Downing, M. A. Caligiuri, C. D. Bloomfield, and E. S. Lander. Molecular classification of cancer: Class discovery and class prediction by gene expression monitoring. *Science*, 286 (5439):531–537, Oct 1999.

A. Gretton, O. Bousquet, A.J. Smola, and B. Schölkopf. Measuring statistical dependence with Hilbert-Schmidt norms. In S. Jain, H. U. Simon, and E. Tomita, editors, *Proceedings of the International Conference on Algorithmic Learning Theory*, pages 63–77. Springer-Verlag, 2005a.

A. Gretton, A. Smola, O. Bousquet, R. Herbrich, A. Belitski, M. Augath, Y. Murayama, J. Pauls, B. Schölkopf, and N. Logothetis. Kernel constrained covariance for dependence measurement. In *AISTATS 10*, pages 112–119, 2005b.

A. Gretton, K. Borgwardt, M. Rasch, B. Schölkopf, and A. Smola. A kernel method for the two-sample problem. In *Advances in Neural Information Processing Systems 15*, pages 513–520, Cambridge, MA, 2007a. MIT Press.

A. Gretton, K. Borgwardt, M. Rasch, B. Schlkopf, and A. Smola. A kernel approach to comparing distributions. *Proceedings of the 22nd Conference on Artificial Intelligence (AAAI-07)*, pages 1637–1641, 2007b.

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, Cambridge, MA, 2008. MIT Press.

S. Gruvberger, M. Ringner, Y. Chen, S. Panavally, L. H. Saal, A. Borg, M. Ferno, C. Peterson, and P. S. Meltzer. Estrogen receptor status in breast cancer is associated with remarkably distinct gene expression patterns. *Cancer Res*, 61(16):5979–5984, Aug 2001.

C. Guestrin, A. Krause, and A. Singh. Near-optimal sensor placements in gaussian processes. In *International Conference on Machine Learning ICML'05*, 2005.

I. Guyon and A. Elisseeff. An introduction to variable and feature selection. *Journal of Machine Learning Research*, 3:1157–1182, March 2003.

I. Guyon, J. Weston, S. Barnhill, and V. Vapnik. Gene selection for cancer classification using support vector machines. *Machine Learning*, 46:389–422, 2002.

T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning*. Springer, New York, 2001.

Wassily Hoeffding. A class of statistics with asymptotically normal distribution. *The Annals of Mathematical Statistics*, 19(3):293–325, 1948.

N. Iizuka, M. Oka, H. Yamada-Okabe, et al. Oligonucleotide microarray for prediction of early intrahepatic recurrence of hepatocellular carcinoma after curative resection. *Lancet*, 361:923–929, 2003.

K. Kira and L. Rendell. A practical approach to feature selection. In *Proc. 9th Intl. Workshop on Machine Learning*, pages 249–256, 1992.

S. Lemm, B. Blankertz, G. Curio, and K.-R. Mülller. Spatio-spectral filters for improving the classification of single trial EEG. *IEEE Trans. Biomed. Eng.*, 52:1541–1548, 2005.

C. Leslie, E. Eskin, J. Weston, and W. S. Noble. Mismatch string kernels for SVM protein classification. In S. Becker, S. Thrun, and K. Obermayer, editors, *Advances in Neural Information Processing Systems 15*, volume 15, Cambridge, MA, 2002. MIT Press.

H. Lodhi, C. Saunders, J. Shawe-Taylor, N. Cristianini, and C. Watkins. Text classification using string kernels. *Journal of Machine Learning Research*, 2:419–444, February 2002.

Ingrid Lönnstedt and Terry Speed. Replicated microarray data. *Statistica Sinica*, 12:31–46, 2002.

Radford M. Neal. Assessing relevance determination methods using delve. In *Neural Networks and Machine Learning*, pages 97–129. Springer, 1998.

I Nemenman, F Shafee, and W Bialek. Entropy and inference, revisited. In *Neural Information Processing Systems*, volume 14, Cambridge, MA, 2002. MIT Press.

G. Nemhauser, L. Wolsey, and M. Fisher. An analysis of the approximations for maximizing submodular set functions. *Mathematical Programming*, 14:265–294, 1978.

J. Neumann, C. Schnörr, and G. Steidl. Combined SVM-based feature selection and classification. *Machine Learning*, 61:129–150, 2005.

N. Quadrianto, L. Song, and A. Smola. Kernelized sorting. In *Advances in Neural Information Processing Systems 22*, 2009.

A. Rosenwald, G. Wright, G. Chan, et al. The use of molecular profiling to predict survival after chemotherapy for diffuse large-b-cell lymphoma. *N. Engl. J. Med.*, 346:1937–1947, 2002.

B. Schölkopf. *Support Vector Learning*. R. Oldenbourg Verlag, Munich, 1997. Download: http://www.kernel-machines.org.

B. Schölkopf, P. L. Bartlett, A. J. Smola, and R. C. Williamson. Shrinking the tube: a new support vector regression algorithm. In M. S. Kearns, S. A. Solla, and D. A. Cohn, editors, *Advances in Neural Information Processing Systems 11*, pages 330–336, Cambridge, MA, 1999. MIT Press.

B. Schölkopf, K. Tsuda, and J.-P. Vert. *Kernel Methods in Computational Biology*. MIT Press, Cambridge, MA, 2004.

Bernhard Schölkopf and Alex Smola. *Learning with Kernels*. MIT Press, Cambridge, MA, 2002.

R. Serfling. *Approximation Theorems of Mathematical Statistics*. Wiley, New York, 1980.

G.K. Smyth. Linear models and empirical bayes methods for assessing differential expressionin microarray experiments. *Statistical Applications in Genetics and Molecular Biology*, 3, 2004.

L. Song, J. Bedo, K.M. Borgwardt, A. Gretton, and A.J. Smola. Gene selection via the BAHSIC family of algorithms. *Bioinformatics (ISMB)*, 23(13):i490–i498, 2007a.

L. Song, A. Smola, A. Gretton, K. Borgwardt, and J. Bedo. Supervised feature selection via dependence estimation. In *ICML*, pages 823–830. Omnipress, 2007b.

B. Sriperumbudur, A. Gretton, K. Fukumizu, G. Lanckriet, and B. Schölkopf. Injective Hilbert space embeddings of probability measures. In *Proc. Annual Conf. Computational Learning Theory*, pages 111–122, 2008.

B. Sriperumbudur, K. Fukumizu, A. Gretton, G. Lanckriet, and B. Schoelkopf. Kernel choice and classifiability for RKHS embeddings of probability distributions. In *Advances in Neural Information Processing Systems 22*, Red Hook, NY, 2009. Curran Associates Inc.

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

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

R. Tibshirani, T. Hastie, B. Narasimhan, and G. Chu. Diagnosis of multiple cancer types by shrunken centroids of gene expression. In *National Academy of Sciences*, volume 99, pages 6567–6572, 2002.

R. Tibshirani, T. Hastie, B. Narasimhan, and G. Chu. Class prediction by nearest shrunken centroids, with applicaitons to dna microarrays. *Stat Sci*, 18:104–117, 2003.

P.H.S. Torr. Solving Markov random fields using semidefinite programming. In *Proceedings of International Workshop on Artificial Intelligence and Statistics*, 2003.

P. J. Valk, R. G. Verhaak, M. A. Beijen, C. A. Erpelinck, S. Barjesteh van Waalwijk van Doorn-Khosrovani, J. M. Boer, H. B. Beverloo, M. J. Moorhouse, P. J. van der Spek, B. Lowenberg, and R. Delwel. Prognostically useful gene-expression profiles in acute myeloid leukemia. *New England Journal of Medicine*, 350(16):1617–1628, Apr 2004.

M. J. van de Vijver, Y. D. He, L. J. van 't Veer, H. Dai, A. A. Hart, D. W. Voskuil, G. J. Schreiber, J. L. Peterse, C. Roberts, M. J. Marton, M. Parrish, D. Atsma, A. Witteveen, A. Glas, L. Delahaye, T. van der Velde, H. Bartelink, S. Rodenhuis, E. T. Rutgers, S. H. Friend, and R. Bernards. A gene-expression signature as a predictor of survival in breast cancer. *New England Journal of Medicine*, 247:1999–2009, 2002.

L. J. van't Veer, H. Dai, M. J. van de Vijver, Y. D. He, A. A. M. Hart, et al. Gene expression profiling predicts clinical outcome of breast cancer. *Nature*, 415:530–536, 2002.

S. V. N. Vishwanathan and A. J. Smola. Fast kernels for string and tree matching. In S. Becker, S. Thrun, and K. Obermayer, editors, *Advances in Neural Information Processing Systems 15*, pages 569–576. MIT Press, Cambridge, MA, 2003.

S. V. N. Vishwanathan, A. J. Smola, and R. Vidal. Binet-Cauchy kernels on dynamical systems and its application to the analysis of dynamic scenes. *International Journal of Computer Vision*, 73 (1):95–119, 2007.

Yixin Wang, Jan G M Klijn, Yi Zhang, Anieta M Sieuwerts, Maxime P Look, Fei Yang, Dmitri Talantov, Mieke Timmermans, Marion E Meijer van Gelder, Jack Yu, Tim Jatkoe, Els M J J Berns, David Atkins, and John A Foekens. Gene-expression profiles to predict distant metastasis of lymph-node-negative primary breast cancer. *Lancet*, 365(9460):671–679, February 2005.

P. Warnat, R. Eils, and B. Brors. Cross-platform analysis of cancer microarray data improves gene expression based classification of phenotypes. *BMC Bioinformatics*, 6:265, Nov 2005.

J. B. Welsh, L. M. Sapinoso, A. I. Su, S. G. Kern, J. Wang-Rodriguez, C. A. Moskaluk, J. r. Frierson HF, and G. M. Hampton. Analysis of gene expression identifies candidate markers and pharmacological targets in prostate cancer. *Cancer Res*, 61(16):5974–5978, Aug 2001.

M. West. Bayesian factor regression models in the "large p, small n" paradigm. *Bayesian Statistics*, 7:723–732, 2003.

J. Weston, S. Mukherjee, O. Chapelle, M. Pontil, T. Poggio, and V. Vapnik. Feature selection for SVMs. In *Advances in Neural Information Processing Systems 13*, pages 668–674, 2000.

J. Weston, A. Elisseeff, B. Schölkopf, and M. Tipping. Use of zero-norm with linear models and kernel methods. *Journal of Machine Learning Research*, 3:1439–1461, 2003.

A.L. Yuille and A. Rangarajan. The concave-convex procedure. *Neural Computation*, 15:915–936, 2003.

M. Zaffalon and M. Hutter. Robust feature selection using distributions of mutual information. In A. Darwiche and N. Friedman, editors, *Proceedings of the 18th International Conference on Uncertainty in Artificial Intelligence (UAI-2002)*, pages 577–584, San Francisco, CA., 2002. Morgan Kaufmann.

Journal of Machine Learning Research 13 (2012) 1435-1468 Submitted 5/11; Revised 12/11; Published 5/12

Structured Sparsity via Alternating Direction Methods

Zhiwei (Tony) Qin ZQ2107@COLUMBIA.EDU
Donald Goldfarb GOLDFARB@COLUMBIA.EDU
Department of Industrial Engineering and Operations Research
Columbia University
New York, NY 10027, USA

Editor: Francis Bach

Abstract

We consider a class of sparse learning problems in high dimensional feature space regularized by a structured sparsity-inducing norm that incorporates prior knowledge of the group structure of the features. Such problems often pose a considerable challenge to optimization algorithms due to the non-smoothness and non-separability of the regularization term. In this paper, we focus on two commonly adopted sparsity-inducing regularization terms, the overlapping Group Lasso penalty l_1/l_2-norm and the l_1/l_∞-norm. We propose a unified framework based on the augmented Lagrangian method, under which problems with both types of regularization and their variants can be efficiently solved. As one of the core building-blocks of this framework, we develop new algorithms using a partial-linearization/splitting technique and prove that the accelerated versions of these algorithms require $O(\frac{1}{\sqrt{\varepsilon}})$ iterations to obtain an ε-optimal solution. We compare the performance of these algorithms against that of the alternating direction augmented Lagrangian and FISTA methods on a collection of data sets and apply them to two real-world problems to compare the relative merits of the two norms.

Keywords: structured sparsity, overlapping Group Lasso, alternating direction methods, variable splitting, augmented Lagrangian

1. Introduction

For feature learning problems in a high-dimensional space, sparsity in the feature vector is usually a desirable property. Many statistical models have been proposed in the literature to enforce sparsity, dating back to the classical Lasso model (l_1-regularization) (Tibshirani, 1996; Chen et al., 1999). The Lasso model is particularly appealing because it can be solved by very efficient proximal gradient methods; for example, see Combettes and Pesquet (2011). However, the Lasso does not take into account the structure of the features (Zou and Hastie, 2005). In many real applications, the features in a learning problem are often highly correlated, exhibiting a group structure. Structured sparsity has been shown to be effective in those cases. The Group Lasso model (Yuan and Lin, 2006; Bach, 2008; Roth and Fischer, 2008) assumes disjoint groups and enforces sparsity on the pre-defined groups of features. This model has been extended to allow for groups that are hierarchical as well as overlapping (Jenatton et al., 2011; Kim and Xing, 2010; Bach, 2010) with a wide array of applications from gene selection (Kim and Xing, 2010) to computer vision (Huang et al., 2009; Jenatton et al., 2010). For image denoising problems, extensions with non-integer block sizes and adaptive partitions have been proposed by Peyre and Fadili (2011) and Peyre et al. (2011). In this paper, we consider the following basic model of minimizing the squared-error loss with a regularization term

to induce group sparsity:

$$\min_{x\in\mathbb{R}^m} L(x)+\Omega(x), \tag{1}$$

where

$$\begin{aligned} L(x) &= \frac{1}{2}\|Ax-b\|^2, \qquad A\in\mathbb{R}^{n\times m}, \\ \Omega(x) &= \begin{cases} \Omega_{l_1/l_2}(x)\equiv\lambda\sum_{s\in\mathcal{S}} w_s\|x_s\|, & \text{or} \\ \Omega_{l_1/l_\infty}(x)\equiv\lambda\sum_{s\in\mathcal{S}} w_s\|x_s\|_\infty & , \end{cases} \end{aligned} \tag{2}$$

$\mathcal{S}=\{s_1,\cdots,s_{|\mathcal{S}|}\}$ is the set of group indices with $|\mathcal{S}|=J$, and the elements (features) in the groups possibly overlap (Chen et al., 2010; Mairal et al., 2010; Jenatton et al., 2011; Bach, 2010). In this model, $\lambda, w_s, \mathcal{S}$ are all pre-defined. $\|\cdot\|$ without a subscript denotes the l_2-norm. We note that the penalty term $\Omega_{l_1/l_2}(x)$ in (2) is different from the one proposed by Jacob et al. (2009),[1] although both are called overlapping Group Lasso penalties. In particular, (1)-(2) cannot be cast into a non-overlapping group lasso problem as done by Jacob et al. (2009).

1.1 Related Work

Two proximal gradient methods have been proposed to solve a close variant of (1) with an l_1/l_2 penalty,

$$\min_{x\in\mathbb{R}^m} L(x)+\Omega_{l_1/l_2}(x)+\lambda\|x\|_1, \tag{3}$$

which has an additional l_1-regularization term on x. Chen et al. (2010) replace $\Omega_{l_1/l_2}(x)$ with a smooth approximation $\Omega_\eta(x)$ by using Nesterov's smoothing technique (Nesterov, 2005) and solve the resulting problem by the Fast Iterative Shrinkage Thresholding algorithm (FISTA) (Beck and Teboulle, 2009). The parameter η is a smoothing parameter, upon which the practical and theoretical convergence speed of the algorithm critically depends. Liu and Ye (2010) also apply FISTA to solve (3), but in each iteration, they transform the computation of the proximal operator associated with the combined penalty term into an equivalent constrained smooth problem and solve it by Nesterov's accelerated gradient descent method (Nesterov, 2005). Mairal et al. (2010) apply the accelerated proximal gradient method to (1) with l_1/l_∞ penalty and propose a network flow algorithm to solve the proximal problem associated with $\Omega_{l_1/l_\infty}(x)$. The method proposed by Mosci et al. (2010) for solving the Group Lasso problem in Jacob et al. (2009) is in the same spirit as the method of Liu and Ye (2010), but their approach uses a projected Newton method.

1.2 Our Contributions

We take a unified approach to tackle problem (1) with both l_1/l_2- and l_1/l_∞-regularizations. Our strategy is to develop efficient algorithms based on the Alternating Linearization Method with Skipping (ALM-S) (Goldfarb et al., 2011) and FISTA for solving an equivalent constrained version of problem (1) (to be introduced in Section 2) in an augmented Lagrangian method framework. Specifically, we make the following contributions in this paper:

- We build a general framework based on the augmented Lagrangian method, under which learning problems with both l_1/l_2 and l_1/l_∞ regularizations (and their variants) can be solved. This framework allows for experimentation with its key building blocks.

1. This norm has been further investigated and renamed as latent Group Lasso (Obozinski et al., 2011).

- We propose new algorithms: ALM-S with partial splitting (APLM-S) and FISTA with partial linearization (FISTA-p), to serve as the key building block for this framework. We prove that APLM-S and FISTA-p have convergence rates of $O(\frac{1}{k})$ and $O(\frac{1}{k^2})$ respectively, where k is the number of iterations. Our algorithms are easy to implement and tune, and they do not require line-search, eliminating the need to evaluate the objective function at every iteration.

- We evaluate the quality and speed of the proposed algorithms and framework against state-of-the-art approaches on a rich set of synthetic test data and compare the l_1/l_2 and l_1/l_∞ models on breast cancer gene expression data (Van De Vijver et al., 2002) and a video sequence background subtraction task (Mairal et al., 2010).

2. A Variable-Splitting Augmented Lagrangian Framework

In this section, we present a unified framework, based on variable splitting and the augmented Lagrangian method for solving (1) with both l_1/l_2- and l_1/l_∞-regularizations. This framework reformulates problem (1) as an equivalent linearly-constrained problem, by using the following variable-splitting procedure.

Let $y \in \mathbb{R}^{\sum_{s\in\mathcal{S}}|s|}$ be the vector obtained from the vector $x \in \mathbb{R}^m$ by repeating components of x so that no component of y belongs to more than one group. Let $M = \sum_{s\in\mathcal{S}}|s|$. The relationship between x and y is specified by the linear constraint $Cx = y$, where the (i,j)-th element of the matrix $C \in \mathbb{R}^{M\times m}$ is

$$C_{i,j} = \begin{cases} 1, & \text{if } y_i \text{ is a replicate of } x_j, \\ 0, & \text{otherwise.} \end{cases}$$

For examples of C, refer to Chen et al. (2010). Consequently, (1) is equivalent to

$$\begin{aligned} \min \quad & F_{obj}(x,y) \equiv \frac{1}{2}\|Ax-b\|^2 + \tilde{\Omega}(y) \\ s.t. \quad & Cx = y, \end{aligned} \tag{4}$$

where $\tilde{\Omega}(y)$ is the non-overlapping group-structured penalty term corresponding to $\Omega(y)$ defined in (2).

Note that C is a highly sparse matrix, and $D = C^TC$ is a diagonal matrix with the diagonal entries equal to the number of times that each entry of x is included in some group. Problem (4) now includes two sets of variables x and y, where x appears only in the loss term $L(x)$ and y appears only in the penalty term $\tilde{\Omega}(y)$.

All the non-overlapping versions of $\Omega(\cdot)$, including the Lasso and Group Lasso, are special cases of $\Omega(\cdot)$, with $C = I$, that is, $x = y$. Hence, (4) in this case is equivalent to applying variable-splitting on x. Problems with a composite penalty term, such as the Elastic Net, $\lambda_1\|x\|_1 + \lambda_2\|x\|^2$, can also be reformulated in a similar way by merging the smooth part of the penalty term ($\lambda_2\|x\|^2$ in the case of the Elastic Net) with the loss function $L(x)$.

To solve (4), we apply the augmented Lagrangian method (Hestenes, 1969; Powell, 1972; Nocedal and Wright, 1999; Bertsekas, 1999) to it. This method, Algorithm 1, minimizes the augmented Lagrangian

$$\mathcal{L}(x,y,v) = \frac{1}{2}\|Ax-b\|^2 - v^T(Cx-y) + \frac{1}{2\mu}\|Cx-y\|^2 + \tilde{\Omega}(y) \tag{5}$$

exactly for a given Lagrange multiplier v in every iteration followed by an update to v. The parameter μ in (5) controls the amount of weight that is placed on violations of the constraint $Cx = y$. Algorithm 1 can also be viewed as a dual ascent algorithm applied to $P(v) = \min_{x,y} \mathcal{L}(x,y,v)$ (Bertsekas, 1976), where v is the dual variable, $\frac{1}{\mu}$ is the step-length, and $Cx - y$ is the gradient $\nabla_v P(v)$. This

Algorithm 1 AugLag

1: Choose x^0, y^0, v^0.
2: **for** $l = 0, 1, \cdots$ **do**
3: $(x^{l+1}, y^{l+1}) \leftarrow \arg\min_{x,y} \mathcal{L}(x,y,v^l)$
4: $v^{l+1} \leftarrow v^l - \frac{1}{\mu}(Cx^{l+1} - y^{l+1})$
5: Update μ according to the chosen updating scheme.
6: **end for**

algorithm does not require μ to be very small to guarantee convergence to the solution of problem (4) (Nocedal and Wright, 1999). However, solving the problem in Line 3 of Algorithm 1 exactly can be very challenging in the case of structured sparsity. We instead seek an approximate minimizer of the augmented Lagrangian via the abstract subroutine ApproxAugLagMin(x,y,v). The following theorem (Rockafellar, 1973) guarantees the convergence of this inexact version of Algorithm 1.

Theorem 1 *Let* $\alpha^l := \mathcal{L}(x^l, y^l, v^l) - \inf_{x \in \mathbb{R}^m, y \in \mathbb{R}^M} \mathcal{L}(x,y,v^l)$ *and* F^* *be the optimal value of problem* (4). *Suppose problem* (4) *satisfies the modified Slater's condition, and*

$$\sum_{l=1}^{\infty} \sqrt{\alpha^l} < +\infty. \tag{6}$$

Then, the sequence $\{v^l\}$ *converges to* v^*, *which satisfies*

$$\inf_{x \in \mathbb{R}^m, y \in \mathbb{R}^M} \left(F_{obj}(x,y) - (v^*)^T (Cx - y) \right) = F^*,$$

while the sequence $\{x^l, y^l\}$ *satisfies* $\lim_{l \to \infty} Cx^l - y^l = 0$ *and* $\lim_{l \to \infty} F_{obj}(x^l, y^l) = F^*$.

The condition (6) requires the augmented Lagrangian subproblem be solved with increasing accuracy. We formally state this framework in Algorithm 2. We index the iterations of Algorithm

Algorithm 2 OGLasso-AugLag

1: Choose x^0, y^0, v^0.
2: **for** $l = 0, 1, \cdots$ **do**
3: $(x^{l+1}, y^{l+1}) \leftarrow$ ApproxAugLagMin(x^l, y^l, v^l), to compute an approximate minimizer of $\mathcal{L}(x,y,v^l)$
4: $v^{l+1} \leftarrow v^l - \frac{1}{\mu}(Cx^{l+1} - y^{l+1})$
5: Update μ according to the chosen updating scheme.
6: **end for**

2 by l and call them 'outer iterations'. In Sections 3, we develop algorithms that implement ApproxAugLagMin(x,y,v). The iterations of these subroutine are indexed by k and are called 'inner iterations'.

3. Methods for Approximately Minimizing the Augmented Lagrangian

In this section, we use the overlapping Group Lasso penalty $\Omega(x) = \lambda \sum_{s\in\mathcal{S}} w_s \|x_s\|$ to illustrate the optimization algorithms under discussion. The case of l_1/l_∞-regularization will be discussed in Section 4. From now on, we assume without loss of generality that $w_s = 1$ for every group s.

3.1 Alternating Direction Augmented Lagrangian (ADAL) Method

The well-known Alternating Direction Augmented Lagrangian (ADAL) method (Eckstein and Bertsekas, 1992; Gabay and Mercier, 1976; Glowinski and Marroco, 1975; Boyd et al., 2010)[2] approximately minimizes the augmented Lagrangian by minimizing (5) with respect to x and y alternatingly and then updates the Lagrange multiplier v on each iteration (e.g., see Bertsekas and Tsitsiklis, 1989, Section 3.4). Specifically, the single-iteration procedure that serves as the procedure ApproxAugLagMin(x,y,v) is given below as Algorithm 3.

Algorithm 3 ADAL

1: Given x^l, y^l, and v^l.
2: $x^{l+1} \leftarrow \arg\min_x \mathcal{L}(x, y^l, v^l)$
3: $y^{l+1} \leftarrow \arg\min_y \mathcal{L}(x^{l+1}, y, v^l)$
4: **return** x^{l+1}, y^{l+1}.

The ADAL method, also known as the alternating direction method of multipliers (ADMM) and the split Bregman method, has recently been applied to problems in signal and image processing (Combettes and Pesquet, 2011; Afonso et al., 2009; Goldstein and Osher, 2009) and low-rank matrix recovery (Lin et al., 2010). Its convergence has been established by Eckstein and Bertsekas (1992). This method can accommodate a sum of more than two functions. For example, by applying variable-splitting (e.g., see Bertsekas and Tsitsiklis, 1989; Boyd et al., 2010) to the problem $\min_x f(x) + \sum_{i=1}^K g_i(C_i x)$, it can be transformed into

$$\begin{aligned} \min_{x,y_1,\cdots,y_K} \quad & f(x) + \sum_{i=1}^K g_i(y_i) \\ s.t. \quad & y_i = C_i x, \quad i = 1,\cdots,K. \end{aligned}$$

The subproblems corresponding to y_i's can thus be solved simultaneously by the ADAL method. This so-called simultaneous direction method of multipliers (SDMM) (Setzer et al., 2010) is related to Spingarn's method of partial inverses (Spingarn, 1983) and has been shown to be a special instance of a more general parallel proximal algorithm with inertia parameters (Pesquet and Pustelnik, 2010).

Note that the problem solved in Line 3 of Algorithm 3,

$$y^{l+1} = \arg\min_y \mathcal{L}(x^{l+1}, y, v^l) \equiv \arg\min_y \left\{ \frac{1}{2\mu}\|d^l - y\|^2 + \tilde{\Omega}(y) \right\}, \tag{7}$$

where $d^l = Cx^{l+1} - \mu v^l$, is group-separable and hence can be solved in parallel. As in Qin et al. (2010), each subproblem can be solved by applying the block soft-thresholding operator, $T(d_s^l, \mu\lambda) \equiv$

2. Recently, Mairal et al. (2011) also applied ADAL with two variants based on variable-splitting to the overlapping Group Lasso problem.

$\frac{d_s^l}{\|d_s^l\|}\max(0,\|d_s^l\|-\lambda\mu), s=1,\cdots,J$. Solving for x^{l+1} in Line 2 of Algorithm 3, that is,

$$x^{l+1}=\arg\min_x \mathcal{L}(x,y^l,v^l)\equiv\arg\min_x\left\{\frac{1}{2}\|Ax-b\|^2-(v^l)^TCx+\frac{1}{2\mu}\|Cx-y^l\|^2\right\}, \tag{8}$$

involves solving the linear system

$$(A^TA+\frac{1}{\mu}D)x=A^Tb+C^Tv^l+\frac{1}{\mu}C^Ty^l, \tag{9}$$

where the matrix on the left hand side of (9) has dimension $m\times m$. Many real-world data sets, such as gene expression data, are highly under-determined. Hence, the number of features (m) is much larger than the number of samples (n). In such cases, one can use the Sherman-Morrison-Woodbury formula,

$$(A^TA+\frac{1}{\mu}D)^{-1}=\mu D^{-1}-\mu^2D^{-1}A^T(I+\mu AD^{-1}A^T)^{-1}AD^{-1},$$

and solve instead an $n\times n$ linear system involving the matrix $I+\mu AD^{-1}A^T$. In addition, as long as μ stays the same, one has to factorize $A^TA+\frac{1}{\mu}D$ or $I+\mu AD^{-1}A^T$ only once and store their factors for subsequent iterations.

When both n and m are very large, it might be infeasible to compute or store A^TA, not to mention its eigen-decomposition, or the Cholesky decomposition of $A^TA+\frac{1}{\mu}D$. In this case, one can solve the linear systems using the preconditioned Conjugate Gradient (PCG) method (Golub and Van Loan, 1996). Similar comments apply to the other algorithms proposed in Sections 3.2 - 3.4 below.Alternatively, we can apply FISTA to Line 3 in Algorithm 2 (see Section 3.5).

3.2 ALM-S: partial split (APLM-S)

We now consider applying the Alternating Linearization Method with Skipping (ALM-S) from Goldfarb et al. (2011) to approximately minimize (5). In particular, we apply variable splitting (Section 2) to the variable y, to which the group-sparse regularizer $\tilde{\Omega}$ is applied, (the original ALM-S splits both variables x and y,) and re-formulate (5) as follows.

$$\begin{array}{ll}\min\limits_{x,y,y} & \frac{1}{2}\|Ax-b\|^2-v^T(Cx-y)+\frac{1}{2\mu}\|Cx-y\|^2+\tilde{\Omega}(y)\\ s.t. & y=y.\end{array} \tag{10}$$

Note that the Lagrange multiplier v is fixed here. Defining

$$f(x,y) := \frac{1}{2}\|Ax-b\|^2-v^T(Cx-y)+\frac{1}{2\mu}\|Cx-y\|^2, \tag{11}$$

$$g(y) = \tilde{\Omega}(y)=\lambda\sum_s\|y_s\|, \tag{12}$$

problem (10) is of the form

$$\begin{array}{ll}\min & f(x,y)+g(y)\\ s.t. & y=y,\end{array} \tag{13}$$

to which we now apply partial-linearization.

3.2.1 PARTIAL LINEARIZATION AND CONVERGENCE RATE ANALYSIS

Let us define

$$\begin{aligned} F(x,y) &:= f(x,y)+g(y) = \mathcal{L}(x,y;v), \\ \mathcal{L}_\rho(x,y,y,\gamma) &:= f(x,y)+g(y)+\gamma^T(y-y)+\frac{1}{2\rho}\|y-y\|^2, \end{aligned} \tag{14}$$

where γ is the Lagrange multiplier in the augmented Lagrangian (14) corresponding to problem (13). We now present our partial-split alternating linearization algorithm to implement ApproxAugLagMin(x,y,v) in Algorithm 2.

Algorithm 4 APLM-S

1: Given x^0, y^0, v. Choose ρ, γ^0, such that $-\gamma^0 \in \partial g(y^0)$. Define $f(x,y)$ as in (11).
2: **for** $k = 0, 1, \cdots$ until stopping criterion is satisfied **do**
3: $(x^{k+1}, y^{k+1}) \leftarrow \arg\min_{x,y} \mathcal{L}_\rho(x,y,y^k,\gamma^k)$.
4: **if** $F(x^{k+1},y^{k+1}) > \mathcal{L}_\rho(x^{k+1},y^{k+1},y^k,\gamma^k)$ **then**
5: $y^{k+1} \leftarrow y^k$
6: $x^{k+1} \leftarrow \arg\min_x f(x,y^{k+1}) \equiv \arg\min_x \mathcal{L}_\rho(x;y^{k+1},y^k,\gamma^k)$
7: **end if**
8: $y^{k+1} \leftarrow p_f(x^{k+1},y^{k+1}) \equiv \arg\min_y \mathcal{L}_\rho(x^{k+1},y^{k+1},y,\nabla_y f(x^{k+1},y^{k+1}))$
9: $\gamma^{k+1} \leftarrow \nabla_y f(x^{k+1},y^{k+1}) - \frac{y^{k+1}-y^{k+1}}{\rho}$
10: **end for**
11: **return** (x^{K+1}, y^{K+1})

We note that in Line 6 in Algorithm 4,

$$x^{k+1} = \arg\min_x \mathcal{L}_\rho(x;y^{k+1},y^k,\gamma^k) \equiv \arg\min_x f(x;y^{k+1}) \equiv \arg\min_x f(x;y^k). \tag{15}$$

Now, we have a variant of Lemma 2.2 in Goldfarb et al. (2011).

Lemma 1 *For any (x,y), if $q := \arg\min_y \mathcal{L}_\rho(x,y,y,\nabla_y f(x,y))$, and*

$$F(x,q) \le \mathcal{L}_\rho(x,y,q,\nabla_y f(x,y)), \tag{16}$$

then for any (x,y),

$$2\rho(F(x,y)-F(x,q)) \ge \|q-y\|^2 - \|y-y\|^2 + 2\rho((x-x)^T\nabla_x f(x,y)). \tag{17}$$

Similarly, for any y, if $(p,q) := \arg\min_{x,y} \mathcal{L}_\rho(x,y,y,-\gamma_g(y))$, $\gamma_g(y)$ is a sub-gradient of g at y, and

$$F(p,q) \le \mathcal{L}_\rho((p,q),y,-\gamma_g(y)), \tag{18}$$

then for any (x,y),

$$2\rho(F(x,y)-F(p,q)) \ge \|q-y\|^2 - \|y-y\|^2. \tag{19}$$

Proof See Appendix A. ■

Algorithm 4 checks condition (18) at Line 4 because the function g is non-smooth and condition (18) may not hold no matter what the value of ρ is. When this condition is violated, a skipping step occurs in which the value of y is set to the value of y in the previous iteration (Line 5) and $\mathcal{L}_\rho$ re-minimized with respect to x (Line 6) to ensure convergence. Let us define a *regular iteration* of Algorithm 4 to be an iteration where no skipping step occurs, that is, Lines 5 and 6 are not executed. Likewise, we define a *skipping iteration* to be an iteration where a skipping step occurs. Now, we are ready to state the iteration complexity result for APLM-S.

Theorem 2 *Assume that $\nabla_y f(x,y)$ is Lipschitz continuous in y with Lipschitz constant $L_y(f)$, that is, for any x, $\|\nabla_y f(x,y) - \nabla_y f(x,z)\| \le L_y(f)\|y-z\|$, for all y and z. For $\rho \le \frac{1}{L_y(f)}$, the iterates (x^k, y^k) in Algorithm 4 satisfy*

$$F(x^k,y^k) - F(x^*,y^*) \le \frac{\|y^0 - y^*\|^2}{2\rho(k+k_n)}, \quad \forall k, \tag{20}$$

where (x^*,y^*) is an optimal solution to (10)*, and k_n is the number of regular iterations among the first k iterations.*

Proof See Appendix B. ■

Remark 1 *For Theorem 2 to hold, we need $\rho \le \frac{1}{L_y(f)}$. From the definition of $f(x,y)$ in* (11)*, it is easy to see that $L_y(f) = \frac{1}{\mu}$ regardless of the loss function $L(x)$. Hence, we set $\rho = \mu$, so that condition* (16) *in Lemma 1 is satisfied.*

In Section 3.3, we will discuss the case where the iterations entirely consist of skipping steps. We will show that this is equivalent to ISTA (Beck and Teboulle, 2009) with partial linearization as well as a variant of ADAL. In this case, the inner Lagrange multiplier γ is redundant.

3.2.2 SOLVING THE SUBPROBLEMS

We now show how to solve the subproblems in Algorithm 4. First, observe that since $\rho = \mu$,

$$\begin{aligned} \arg\min_y \mathcal{L}_\rho(x,y,y,\nabla_y f(x,y)) &\equiv \arg\min_y \left\{ \nabla_y f(x,y)^T y + \frac{1}{2\mu}\|y-y\|^2 + g(y) \right\} \\ &\equiv \arg\min_y \left\{ \frac{1}{2\mu}\|d-y\|^2 + \lambda \sum_s \|y_s\| \right\}, \end{aligned}$$

where $d = Cx - \mu v$. Hence, y can be obtained by applying the block soft-thresholding operator $T(d_s, \mu\lambda)$ as in Section 3.1. Next consider the subproblem

$$\min_{(x,y)} \mathcal{L}_\rho(x,y,y,\gamma) \equiv \min_{(x,y)} \left\{ f(x,y) + \gamma^T(y-y) + \frac{1}{2\mu}\|y-y\|^2 \right\}. \tag{21}$$

It is easy to verify that solving the linear system given by the optimality conditions for (21) by block Gaussian elimination yields the system

$$\left(A^TA+\frac{1}{2\mu}D\right)x = r_x+\frac{1}{2}C^Tr_y$$

for computing x, where $r_x = A^Tb+C^Tv$ and $r_y = -v+\gamma+\frac{y}{\rho}$. Then y can be computed as $(\frac{\mu}{2})(r_y+\frac{1}{\mu}Cx)$.

As in Section 3.1, only one Cholesky factorization of $A^TA+\frac{1}{2\mu}D$ is required for each invocation of Algorithm 4. Hence, the amount of work involved in each iteration of Algorithm 4 is comparable to that of an ADAL iteration.

It is straightforward to derive an accelerated version of Algorithm 4, which we shall refer to as FAPLM-S, that corresponds to a partial-split version of the FALM algorithm proposed by Goldfarb et al. (2011) and also requires $O(\sqrt{\frac{L(f)}{\varepsilon}})$ iterations to obtain an ε-optimal solution. In Section 3.4, we present an algorithm FISTA-p, which is a special version of FAPLM-S in which every iteration is a skipping iteration and which has a much simpler form than FAPLM-S, while having essentially the same iteration complexity.

It is also possible to apply ALM-S directly, which splits both x and y, to solve the augmented Lagrangian subproblem. Similar to (10), we reformulate (5) as

$$\begin{aligned}\min_{(x,y),(x,y)} \quad & \frac{1}{2}\|Ax-b\|^2-v^T(Cx-y)+\frac{1}{2\mu}\|Cx-y\|^2+\lambda\sum_s\|y_s\| \\ s.t. \quad & x=x, \\ & y=y.\end{aligned} \tag{22}$$

The functions f and g are defined as in (11) and (12), except that now we write g as $g(x,y)$ even though the variable x does not appear in the expression for g. It can be shown that y admits exactly the same expression as in APLM-S, whereas x is obtained by a gradient step, $x-\rho\nabla_x f(x,y)$. To obtain x, we solve the linear system

$$\left(A^TA+\frac{1}{\mu+\rho}D+\frac{1}{\rho}I\right)x = r_x+\frac{\rho}{\mu+\rho}C^Tr_y, \tag{23}$$

after which y is computed by $y=\left(\frac{\mu\rho}{\mu+\rho}\right)\left(r_y+\frac{1}{\mu}Cx\right)$.

Remark 2 *For ALM-S, the Lipschitz constant for $\nabla f(x,y)$ $L_f = \lambda_{max}(A^TA)+\frac{1}{\mu}d_{max}$, where $d_{max} = \max_i D_{ii} \geq 1$. For the complexity results in Goldfarb et al. (2011) to hold, we need $\rho \leq \frac{1}{L_f}$. Since $\lambda_{max}(A^TA)$ is usually not known, it is necessary to perform a backtracking line-search on ρ to ensure that $F(x^{k+1},y^{k+1}) \leq \mathcal{L}_\rho(x^{k+1},y^{k+1},x^k,y^k,\gamma^k)$. In practice, we adopted the following continuation scheme instead. We initially set $\rho=\rho_0=\frac{\mu}{d_{max}}$ and decreased ρ by a factor of β after a given number of iterations until ρ reached a user-supplied minimum value ρ_{min}. This scheme prevents ρ from being too small, and hence negatively impacting computational performance. However, in both cases the left-hand-side of the system* (23) *has to be re-factorized every time ρ is updated.*

As we have seen above, the Lipschitz constant resulting from splitting both x and y is potentially much larger than $\frac{1}{\mu}$. Hence, partial-linearization reduces the Lipschitz constant and hence improves

the bound on the right-hand-side of (20) and allows Algorithm 4 to take larger step sizes (equal to μ). Compared to ALM-S, solving for x in the skipping step (Line 6) becomes harder. Intuitively, APLM-S does a better job of 'load-balancing' by managing a better trade-off between the hardness of the subproblems and the practical convergence rate.

3.3 ISTA: Partial Linearization (ISTA-p)

We can also minimize the augmented Lagrangian (5), which we write as $\mathcal{L}(x,y,v) = f(x,y) + g(y)$ with $f(x,y)$ and $g(y)$ defined as in (11) and (12), using a variant of ISTA that only linearizes $f(x,y)$ with respect to the y variables. As in Section 3.2, we can set $\rho = \mu$ and guarantee the convergence properties of ISTA-p (see Corollary 1 below). Formally, let (x,y) be the current iterate and (x^+,y^+) be the next iterate. We compute y^+ by

$$\begin{aligned} y^+ &= \arg\min_{y'} \mathcal{L}_\rho(x,y,y',\nabla_y f(x,y)) \\ &= \arg\min_{y'} \left\{ \frac{1}{2\mu} \sum_j (\|y'_j - d_{y_j}\|^2 + \lambda \|y'_j\|) \right\}, \end{aligned} \tag{24}$$

where $d_y = Cx - \mu v$. Hence the solution y^+ to problem (24) is given blockwise by $T([d_y]_j, \mu\lambda), j = 1,\cdots,J$.

Now given y^+, we solve for x^+ by

$$\begin{aligned} x^+ &= \arg\min_{x'} f(x',y^+) \\ &= \arg\min_{x'} \left\{ \frac{1}{2}\|Ax'-b\|^2 - v^T(Cx'-y^+) + \frac{1}{2\mu}\|Cx'-y^+\|^2 \right\} \end{aligned} \tag{25}$$

The algorithm that implements subroutine ApproxAugLagMin(x,y,v) in Algorithm 2 by ISTA with partial linearization is stated below as Algorithm 5.

Algorithm 5 ISTA-p (partial linearization)

1: Given x^0, y^0, v. Choose ρ. Define $f(x,y)$ as in (11).
2: **for** $k = 0,1,\cdots$ until stopping criterion is satisfied **do**
3: $\quad x^{k+1} \leftarrow \arg\min_x f(x;y^k)$
4: $\quad y^{k+1} \leftarrow \arg\min_y \mathcal{L}_\rho(x^{k+1}, y^k, y, \nabla_y f(x^{k+1}, y^k))$
5: **end for**
6: **return** (x^{K+1}, y^{K+1})

As we remarked in Section 3.2, Algorithm 5 is equivalent to Algorithm 4 (APLM-S) where every iteration is a skipping iteration. Hence, we have from Theorem 2.

Corollary 1 *Assume $\nabla_y f(\cdot,\cdot)$ is Lipschitz continuous with Lipschitz constant $L_y(f)$. For $\rho \le \frac{1}{L_y(f)}$, the iterates (x^k,y^k) in Algorithm 5 satisfy*

$$F(x^k,y^k) - F(x^*,y^*) \le \frac{\|y^0 - y^*\|^2}{2\rho k}, \quad \forall k,$$

where (x^*,y^*) is an optimal solution to (10).

It is easy to see that (24) is equivalent to (7), and that (25) is the same as (8) in ADAL.

Remark 3 *We have shown that with a fixed v, the ISTA-p iterations are exactly the same as the ADAL iterations. The difference between the two algorithms is that ADAL updates the (outer) Lagrange multiplier v in each iteration, while in ISTA-p, v stays the same throughout the inner iterations. We can thus view ISTA-p as a variant of ADAL with delayed updating of the Lagrange multiplier.*

The 'load-balancing' behavior discussed in Section 3.2 is more obvious for ISTA-p. As we will see in Section 3.5, if we apply ISTA (with full linearization) to minimize (5), solving for x is simply a gradient step. Here, we need to minimize $f(x,y)$ with respect to x exactly, while being able to take larger step sizes in the other subproblem, due to the smaller associated Lipschitz constant.

3.4 FISTA-p

We now present an accelerated version FISTA-p of ISTA-p. FISTA-p is a special case of FAPLM-S with a skipping step occurring in every iteration.We state the algorithm formally as Algorithm 6. The iteration complexity of FISTA-p (and FAPLM-S) is given by the following theorem.

Algorithm 6 FISTA-p (partial linearization)

1: Given x^0, y^0, v. Choose ρ, and $z^0 = y^0$. Define $f(x,y)$ as in (11).
2: **for** $k = 0, 1, \cdots, K$ **do**
3: $\quad x^{k+1} \leftarrow \arg\min_x f(x; z^k)$
4: $\quad y^{k+1} \leftarrow \arg\min_y \mathcal{L}_\rho(x^{k+1}, z^k, y, \nabla_y f(x^{k+1}, z^k))$
5: $\quad t_{k+1} \leftarrow \frac{1+\sqrt{1+4t_k^2}}{2}$
6: $\quad z^{k+1} \leftarrow y^{k+1} + \left(\frac{t_k - 1}{t_{k+1}}\right)(y^{k+1} - y^k)$
7: **end for**
8: **return** (x^{K+1}, y^{K+1})

Theorem 3 *Assuming that $\nabla_y f(\cdot)$ is Lipschitz continuous with Lipschitz constant $L_y(f)$ and $\rho \leq \frac{1}{L_y(f)}$, the sequence $\{x^k, y^k\}$ generated by Algorithm 6 satisfies*

$$F(x^k, y^k) - F(x^*, y^*) \leq \frac{2\|y^0 - y^*\|^2}{\rho(k+1)^2},$$

Although we need to solve a linear system in every iteration of Algorithms 4, 5, and 6, the left-hand-side of the system stays constant throughout the invocation of the algorithms because, following Remark 1, we can always set $\rho = \mu$. Hence, no line-search is necessary, and this step essentially requires only one backward- and one forward-substitution, the complexity of which is the same as a gradient step.

3.5 ISTA/FISTA: Full Linearization

ISTA solves the following problem in each iteration to produce the next iterate $\begin{pmatrix} x^+ \\ y^+ \end{pmatrix}$.

$$\begin{aligned} \min_{x',y'} \quad & \frac{1}{2\rho}\left\|\begin{pmatrix} x' \\ y' \end{pmatrix} - d\right\|^2 + \lambda\sum_s \|y_s\| \\ \equiv \quad & \frac{1}{2\rho}\|x' - d_x\|^2 + \sum_j \frac{1}{2\rho}\left(\|y'_j - d_{y_j}\|^2 + \lambda\|y'_j\|\right), \end{aligned} \tag{26}$$

where $d = \begin{pmatrix} d_x \\ d_y \end{pmatrix} = \begin{pmatrix} x \\ y \end{pmatrix} - \rho\nabla f(x,y)$, and $f(x,y)$ is defined in (11). It is easy to see that we can solve for x^+ and y^+ separately in (26). Specifically,

$$\begin{aligned} x^+ &= d_x, \\ y_j^+ &= \frac{d_{y_j}}{\|d_{y_j}\|}\max(0, \|d_{y_j}\| - \lambda\rho), \quad j = 1,\ldots,J. \end{aligned} \tag{27}$$

Using ISTA to solve the outer augmented Lagrangian (5) subproblem is equivalent to taking only skipping steps in ALM-S. In our experiments, we used the accelerated version of ISTA, that is, FISTA (Algorithm 7) to solve (5).

Algorithm 7 FISTA

1: Given x^0, y^0, v. Choose ρ^0. Set $t_0 = 0, z_x^0 = x^0, z_y^0 = y^0$. Define $f(x,y)$ as in (11).
2: **for** $k = 0, 1, \cdots$ until stopping criterion is satisfied **do**
3: Perform a backtracking line-search on ρ, starting from ρ_0.
4: $\begin{pmatrix} d_x \\ d_y \end{pmatrix} = \begin{pmatrix} z_x^k \\ z_y^k \end{pmatrix} - \rho\nabla f(z_x^k, z_y^k)$
5: $x^{k+1} \leftarrow d_x$
6: $y_j^{k+1} \leftarrow \frac{d_{y_j}}{\|d_{y_j}\|}\max(0, \|d_{y_j}\| - \lambda\rho), \quad j = 1,\ldots,J.$
7: $t_{k+1} \leftarrow \frac{1+\sqrt{1+4t_k^2}}{2}$
8: $z_x^{k+1} \leftarrow x^k + \frac{t_k - 1}{t_{k+1}}(x^{k+1} - x^k)$
9: $z_y^{k+1} \leftarrow y^k + \frac{t_k - 1}{t_{k+1}}(y^{k+1} - y^k)$
10: **end for**
11: **return** (x^{K+1}, y^{K+1})

FISTA (resp. ISTA) is, in fact, an inexact version of FISTA-p (resp. ISTA-p), where we minimize with respect to x a linearized approximation

$$\tilde{f}(x, z^k) := f(x^k, z^k) + \nabla_x f(x^k, z^k)(x - x^k) + \frac{1}{2\rho}\|x - x^k\|^2$$

of the quadratic objective function $f(x, z^k)$ in (25). The update to x in Line 3 of Algorithm 6 is replaced by (27) as a result. Similar to FISTA-p, FISTA is also a special skipping version of the full-split FALM-S. Considering that FISTA has an iteration complexity of $O(\frac{1}{k^2})$, it is not surprising that FISTA-p has the same iteration complexity.

Remark 4 *Since FISTA requires only the gradient of $f(x,y)$, it can easily handle any smooth convex loss function, such as the logistic loss for binary classification, $L(x) = \sum_{i=1}^{N} \log(1+\exp(-b_i a_i^T x))$, where a_i^T is the i-th row of A, and b is the vector of labels. Moreover, when the scale of the data $(\min\{n,m\})$ is so large that it is impractical to compute the Cholesky factorization of A^TA, FISTA is a good choice to serve as the subroutine ApproxAugLagMin(x,y,v) in OGLasso-AugLag.*

4. Overlapping Group l_1/l_∞-Regularization

The subproblems with respect to y (or y) involved in all the algorithms presented in the previous sections take the following form

$$\min_y \frac{1}{2\rho}\|c-y\|^2 + \tilde{\Omega}(y), \tag{28}$$

where $\tilde{\Omega}(y) = \lambda \sum_{s\in\tilde{S}} w_s \|y_s\|_\infty$ in the case of l_1/l_∞-regularization. In (7), for example, $c = Cx - \mu v$. The solution to (28) is the proximal operator of $\tilde{\Omega}$ (Combettes and Wajs, 2006; Combettes and Pesquet, 2011). Similar to the classical Group Lasso, this problem is block-separable and hence all blocks can be solved simultaneously.

Again, for notational simplicity, we assume $w_s = 1 \quad \forall s \in \tilde{S}$ and omit it from now on. For each $s \in \tilde{S}$, the subproblem in (28) is of the form

$$\min_{y_s} \quad \frac{1}{2}\|c_s - y_s\|^2 + \rho\lambda\|y_s\|_\infty. \tag{29}$$

As shown by Wright et al. (2009), the optimal solution to the above problem is $c_s - P(c_s)$, where P denotes the orthogonal projector onto the ball of radius $\rho\lambda$ in the dual norm of the l_∞-norm, that is, the l_1-norm. The Euclidean projection onto the simplex can be computed in (expected) linear time (Duchi et al., 2008; Brucker, 1984). Duchi et al. (2008) show that the problem of computing the Euclidean projection onto the l_1-ball can be reduced to that of finding the Euclidean projection onto the simplex in the following way. First, we replace c_s in problem (29) by $|c_s|$, where the absolute value is taken component-wise. After we obtain the projection z_s onto the simplex, we can construct the projection onto the l_1-ball by setting $y_s^* = sign(c_s)z_s$, where $sign(\cdot)$ is also taken component-wise.

5. Experiments

We tested the OGLasso-AugLag framework (Algorithm 2) with four subroutines: ADAL, FISTA, FISTA-p, and APLM-S. We implemented the framework with the first three subroutines in C++ to compare them with the ProxFlow algorithm proposed by Mairal et al. (2010). We used the C interface and BLAS and LAPACK subroutines provided by the AMD Core Math Library (ACML).[3] To compare with ProxGrad (Chen et al., 2010), we implemented the framework and all four algorithms in Matlab. We did not include ALM-S in our experiments because it is time-consuming to find the right ρ for the inner loops as discussed in Remark 2, and our preliminary computational experience showed that ALM-S was slower than the other algorithms, even when the heuristic ρ-setting scheme discussed in Remark 2 was used, because a large number of steps were skipping steps, which meant

3. ACML can be found at `http://developer.amd.com/libraries/acml/pages/default.aspx`. Ideally, we should have used the Intel Math Kernel Library (Intel MKL), which is optimized for Intel processors, but Intel MKL is not freely available.

Algorithm	Outer rel. dual residual s^{l+1}	Inner iteration	
		Rel. primal residual	Rel. objective gradient residual
ADAL	$\frac{\|C^T(y^{l+1}-y^l)\|}{\|C^T y^l\|}$	-	-
FISTA-p	$\frac{\|C^T(y^{K+1}-z^K)\|}{\|C^T z^K\|}$	$\frac{\|y^{k+1}-z^k\|}{\|z^k\|}$	$\frac{\|C^T(y^{k+1}-z^k)\|}{\|C^T z^k\|}$
APLM-S	$\frac{\|C^T(y^{K+1}-y^{K+1})\|}{\|C^T y^{K+1}\|}$	$\frac{\|y^{k+1}-y^{k+1}\|}{\|y^{k+1}\|}$	$\frac{\|C^T(y^{k+1}-y^{k+1}\|)}{\|C^T y^{k+1}\|}$
FISTA	$\frac{\left\|\begin{pmatrix} x^{K+1} \\ y^{K+1}\end{pmatrix}-\begin{pmatrix} z_x^K \\ z_y^K\end{pmatrix}\right\|}{\left\|\begin{pmatrix} z_x^K \\ z_y^K\end{pmatrix}\right\|}$	$\frac{\left\|\begin{pmatrix} x^{k+1} \\ y^{k+1}\end{pmatrix}-\begin{pmatrix} z_x^k \\ z_y^k\end{pmatrix}\right\|}{\left\|\begin{pmatrix} z_x^k \\ z_y^k\end{pmatrix}\right\|}$	$\frac{\left\|\begin{pmatrix} x^{k+1} \\ y^{k+1}\end{pmatrix}-\begin{pmatrix} z_x^k \\ z_y^k\end{pmatrix}\right\|}{\left\|\begin{pmatrix} z_x^k \\ z_y^k\end{pmatrix}\right\|}$

Table 1: Specification of the quantities used in the outer and inner stopping criteria.

that the computation involved in solving the linear systems in those steps was wasted. All of our experiments were performed on a laptop PC with an Intel Core 2 Duo 2.0 GHz processor and 4 Gb of memory.

5.1 Algorithm Parameters and Termination Criteria

Each algorithm (framework + subroutine)[4] required several parameters to be set and termination criteria to be specified. We used stopping criteria based on the primal and dual residuals suggested by Boyd et al. (2010). We specify the criteria for each of the algorithms below, but defer their derivation to Appendix C. The maximum number of outer iterations was set to 500, and the tolerance for the outer loop was set at $\varepsilon_{out} = 10^{-4}$. The number of inner-iterations was capped at 2000, and the tolerance at the l-th outer iteration for the inner loops was ε_{in}^l. Our termination criterion for the outer iterations was

$$\max\{r^l, s^l\} \leq \varepsilon_{out},$$

where $r^l = \frac{\|Cx^l - y^l\|}{\max\{\|Cx^l\|, \|y^l\|\}}$ is the outer relative primal residual and s^l is the relative dual residual, which is given for each algorithm in Table 1. Recall that $K+1$ is the index of the last inner iteration of the l-th outer iteration; for example, for APLM-S, (x^{l+1}, y^{l+1}) takes the value of the last inner iterate (x^{K+1}, y^{K+1}). We stopped the inner iterations when the maximum of the relative primal residual and the relative objective gradient for the inner problem was less than ε_{in}^l. (See Table 1 for the expressions of these two quantities.) We see there that s^{l+1} can be obtained directly from the relative gradient residual computed in the last inner iteration of the l-th outer iteration.

We set $\mu_0 = 0.01$ in all algorithms except that we set $\mu_0 = 0.1$ in ADAL for the data sets other than the first synthetic set and the breast cancer data set. We set $\rho = \mu$ in FISTA-p and APLM-S and $\rho_0 = \mu$ in FISTA.

For Theorem 1 to hold, the solution returned by the function ApproxAugLagMin(x, y, v) has to become increasingly more accurate over the outer iterations. However, it is not possible to evaluate the sub-optimality quantity α^l in (6) exactly because the optimal value of the augmented Lagrangian $\mathcal{L}(x, y, v^l)$ is not known in advance. In our experiments, we used the maximum of the relative primal

4. For conciseness, we use the subroutine names (e.g., FISTA-p) to represent the full algorithms that consist of the OGLasso-AugLag framework and the subroutines.

and dual residuals ($\max\{r^l, s^l\}$) as a surrogate to α^l for two reasons: First, it has been shown (Boyd et al., 2010) that r^l and s^l are closely related to α^l. Second, the quantities r^l and s^l are readily available as bi-products of the inner and outer iterations. To ensure that the sequence $\{\varepsilon_{in}^l\}$ satisfies (6), we basically set:

$$\varepsilon_{in}^{l+1} = \beta_{in}\varepsilon_{in}^l, \tag{30}$$

with $\varepsilon_{in}^0 = 0.01$ and $\beta_{in} = 0.5$. However, since we terminate the outer iterations at $\varepsilon_{out} > 0$, it is not necessary to solve the subproblems to an accuracy much higher than the one for the outer loop. On the other hand, it is also important for ε_{in}^l to decrease to below ε_{out}, since s^l is closely related to the quantities involved in the inner stopping criteria. Hence, we slightly modified (30) and used $\varepsilon_{in}^{l+1} = \max\{\beta_{in}\varepsilon_{in}^l, 0.2\varepsilon_{out}\}$.

Recently, we became aware of an alternative 'relative error' stopping criterion (Eckstein and Silva, 2012) for the inner loops, which guarantees convergence of Algorithm 2. In our context, this criterion essentially requires that the absolute dual residual is less than a fraction of the absolute primal residual. For FISTA-p, for instance, this condition requires that the $(l+1)$-th iterate satisfies

$$2\left\|\begin{pmatrix} w_x^0 - x^{l+1} \\ w_y^l - y^{l+1} \end{pmatrix}\right\| s^{l+1} + \frac{(s^{l+1})^2}{\mu^2} \leq \sigma(r^{l+1})^2,$$

where r and s are the numerators in the expressions for r and s respectively, $\sigma = 0.99$, w_x^0 is a constant, and w_y is an auxiliary variable updated in each outer iteration by $w_y^{l+1} = w_y^l - \frac{1}{\mu^2}C^T(y^{K+1} - z^K)$. We experimented with this criterion but did not find any computational advantage over the heuristic based on the relative primal and dual residuals.

5.2 Strategies for Updating μ

The penalty parameter μ in the outer augmented Lagrangian (5) not only controls the infeasibility in the constraint $Cx = y$, but also serves as the step-length in the y-subproblem (and the x-subproblem in the case of FISTA). We adopted two kinds of strategies for updating μ. The first one simply kept μ fixed. In this case, choosing an appropriate μ_0 was important for good performance. This was especially true for ADAL in our computational experiments. Usually, a μ_0 in the range of 10^{-1} to 10^{-3} worked well.

The second strategy is a dynamic scheme based on the values r^l and s^l (Boyd et al., 2010). Since $\frac{1}{\mu}$ penalizes the primal infeasibility, a small μ tends to result in a small primal residual. On the other hand, a large μ tends to yield a small dual residual. Hence, to keep r^l and s^l approximately balanced in each outer iteration, our scheme updated μ as follows:

$$\mu^{l+1} \leftarrow \begin{cases} \max\{\beta\mu^l, \mu_{min}\}, & \text{if } r^l > \tau s^l \\ \min\{\mu^l/\beta, \mu_{max}\}, & \text{if } s^l > \tau r^l \\ \mu^l, & \text{otherwise,} \end{cases}$$

where we set $\mu_{max} = 10$, $\mu_{min} = 10^{-6}$, $\tau = 10$ and $\beta = 0.5$, except for the first synthetic data set, where we set $\beta = 0.1$ for ADAL, FISTA-p, and APLM-S.

5.3 Synthetic Examples

To compare our algorithms with the ProxGrad algorithm of Chen et al. (2010), we first tested a synthetic data set (ogl) using the procedure reported by Chen et al. (2010) and Jacob et al. (2009).

The sequence of decision variables x were arranged in groups of ten, with adjacent groups having an overlap of three variables. The support of x was set to the first half of the variables. Each entry in the design matrix A and the non-zero entries of x were sampled from i.i.d. standard Gaussian distributions, and the output b was set to $b = Ax + \varepsilon$, where the noise $\varepsilon \sim \mathcal{N}(0, I)$. Two sets of data were generated as follows: (a) Fix $n = 5000$ and vary the number of groups J from 100 to 1000 with increments of 100. (b) Fix $J = 200$ and vary n from 1000 to 10000 with increments of 1000. The stopping criterion for ProxGrad was the same as the one used for FISTA, and we set its smoothing parameter to 10^{-3}. Figure 1 plots the CPU times taken by the Matlab version of our algorithms and ProxGrad (also in Matlab) on theses scalability tests on l_1/l_2-regularization. A subset of the numerical results on which these plots are based is presented in Tables 4 and 5.

The plots clearly show that the alternating direction methods were much faster than ProxGrad on these two data sets. Compared to ADAL, FISTA-p performed slightly better, while it showed obvious computational advantage over its general version APLM-S. In the plot on the left of Figure 1, FISTA exhibited the advantage of a gradient-based algorithm when both n and m are large. In that case (towards the right end of the plot), the Cholesky factorizations required by ADAL, APLM-S, and FISTA-p became relatively expensive. When $\min\{n, m\}$ is small or the linear systems can be solved cheaply, as the plot on the right shows, FISTA-p and ADAL have an edge over FISTA due to the smaller numbers of inner iterations required.

We generated a second data set (dct) using the approach of Mairal et al. (2010) for scalability tests on both the l_1/l_2 and l_1/l_∞ group penalties. The design matrix A was formed from over-complete dictionaries of discrete cosine transforms (DCT). The set of groups were all the contiguous sequences of length five in one-dimensional space. x had about 10% non-zero entries, selected randomly. We generated the output as $b = Ax + \varepsilon$, where $\varepsilon \sim \mathcal{N}(0, 0.01\|Ax\|^2)$. We fixed $n = 1000$ and varied the number of features m from 5000 to 30000 with increments of 5000. This set of data leads to considerably harder problems than the previous set because the groups are heavily overlapping, and the DCT dictionary-based design matrix exhibits local correlations. Due to the excessive running time required on Matlab, we ran the C++ version of our algorithms for this data set, leaving out APLM-S and ProxGrad, whose performance compared to the other algorithms is already fairly clear from Figure 1. For ProxFlow, we set the tolerance on the relative duality gap to 10^{-4}, the same as ε_{out}, and kept all the other parameters at their default values.

Figure 2 presents the CPU times required by the algorithms versus the number of features. In the case of l_1/l_2-regularization, it is clear that FISTA-p outperformed the other two algorithms. For l_1/l_∞-regularization, ADAL and FISTA-p performed equally well and compared favorably to ProxFlow. In both cases, the growth of the CPU times for FISTA follows the same trend as that for FISTA-p, and they required a similar number of outer iterations, as shown in Tables 6 and 7. However, FISTA lagged behind in speed due to larger numbers of inner iterations. Unlike in the case of the ogl data set, Cholesky factorization was not a bottleneck for FISTA-p and ADAL here because we needed to compute it only once.

To simulate the situation where computing or caching A^TA and its Cholesky factorization is not feasible, we switched ADAL and FISTA-p to PCG mode by always using PCG to solve the linear systems in the subproblems. We compared the performance of ADAL, FISTA-p, and FISTA on the previous data set for both l_1/l_2 and l_1/l_∞ models. The results for ProxFlow are copied from from Figure 2 and Table 9 to serve as a reference. We experimented with the fixed-value and the dynamic updating schemes for μ on all three algorithms. From Figure 3, it is clear that the performance of FISTA-p was significantly improved by using the dynamic scheme. For ADAL,

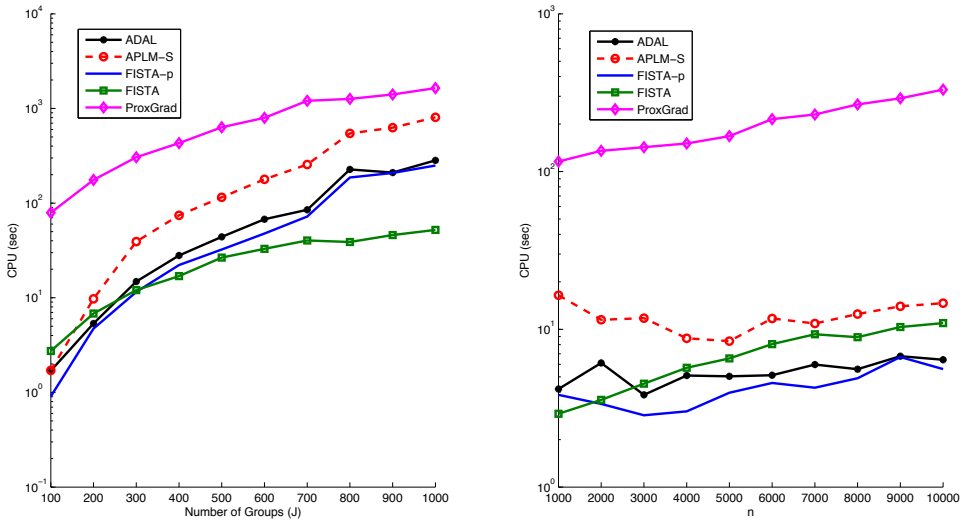


Figure 1: Scalability test results of the algorithms on the synthetic overlapping Group Lasso data sets from Chen et al. (2010). The scale of the y-axis is logarithmic. The dynamic scheme for μ was used for all algorithms except ProxGrad.

however, the dynamic scheme worked well only in the l_1/l_2 case, whereas the performance turned worse in general in the l_1/l_∞ case. We did not include the results for FISTA with the dynamic scheme because the solutions obtained were considerably more suboptimal than the ones obtained with the fixed-μ scheme. Tables 8 and 9 report the best results of the algorithms in each case. The plots and numerical results show that FISTA-p compares favorably to ADAL and stays competitive to ProxFlow. In terms of the quality of the solutions, FISTA-p and ADAL also did a better job than FISTA, as evidenced in Table 9. On the other hand, the gap in CPU time between FISTA and the other three algorithms is less obvious.

5.4 Real-world Examples

To demonstrate the practical usefulness of our algorithms, we tested our algorithms on two real-world applications.

5.4.1 Breast Cancer Gene Expressions

We used the breast cancer data set (Van De Vijver et al., 2002) with canonical pathways from MSigDB (Subramanian et al., 2005). The data was collected from 295 breast cancer tumor samples and contains gene expression measurements for 8,141 genes. The goal was to select a small set of the most relevant genes that yield the best prediction performance. A detailed description of the data set can be found in Chen et al. (2010) and Jacob et al. (2009). In our experiment, we performed a regression task to predict the length of survival of the patients. The canonical pathways naturally provide grouping information of the genes. Hence, we used them as the groups for the group-structured regularization term $\Omega(\cdot)$.

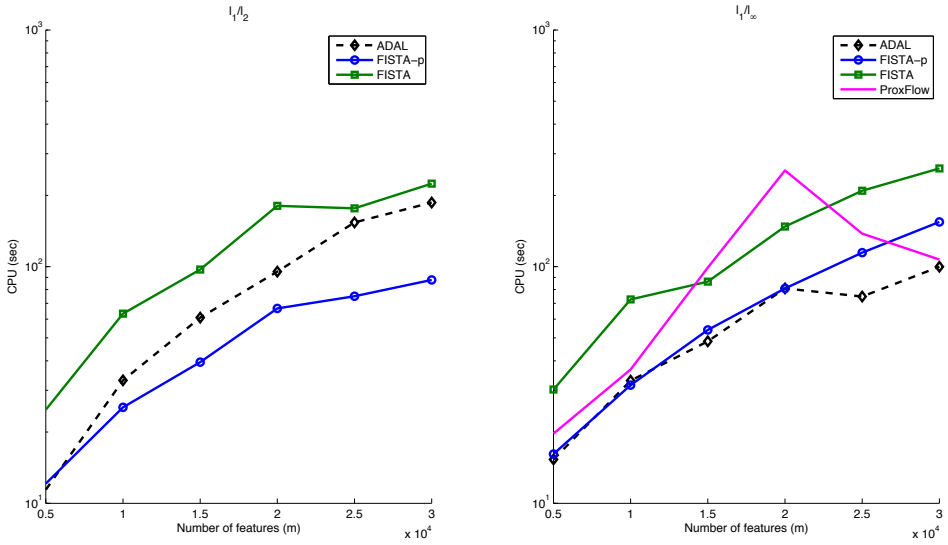


Figure 2: Scalability test results on the DCT set with l_1/l_2-regularization (left column) and l_1/l_∞-regularization (right column). The scale of the y-axis is logarithmic. All of FISTA-p, FITSA, and ADAL were run with a fixed $\mu = \mu_0$.

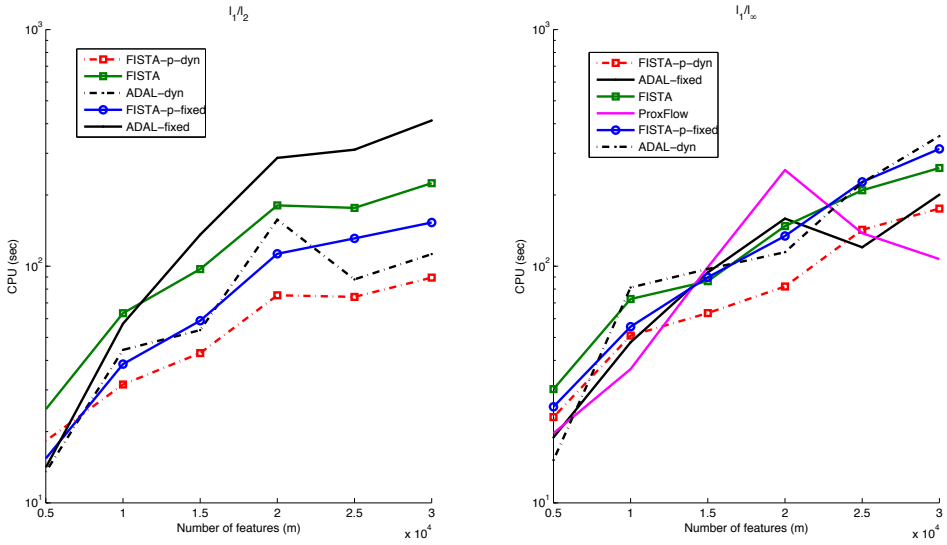


Figure 3: Scalability test results on the DCT set with l_1/l_2-regularization (left column) and l_1/l_∞-regularization (right column). The scale of the y-axis is logarithmic. FISTA-p and ADAL are in PCG mode. The dotted lines denote the results obtained with the dynamic updating scheme for μ.

Data sets	N (no. samples)	J (no. groups)	group size	average frequency
BreastCancerData	295	637	23.7 (avg)	4

Table 2: The Breast Cancer Data Set

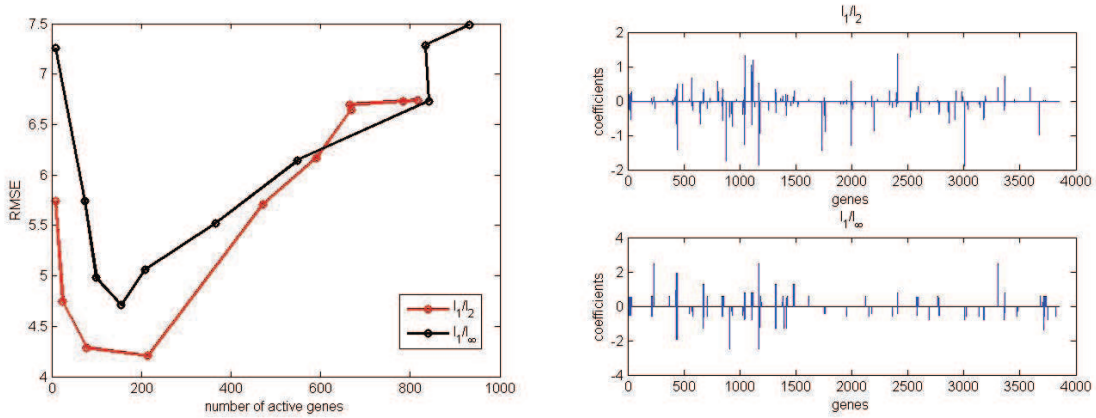


Figure 4: On the left: Plot of root-mean-squared-error against the number of active genes for the Breast Cancer data. The plot is based on the regularization path for ten different values for λ. The total CPU time (in Matlab) using FISTA-p was 51 seconds for l_1/l_2-regularization and 115 seconds for l_1/l_∞-regularization. On the right: The recovered sparse gene coefficients for predicting the length of the survival period. The value of λ used here was the one minimizing the RMSE in the plot on the left.

Table 2 summarizes the data attributes. The numerical results for the l_1/l_2-norm are collected in Table 10, which show that FISTA-p and ADAL were the fastest on this data set. Again, we had to tune ADAL with different initial values (μ_0) and updating schemes of μ for speed and quality of the solution, and we eventually kept μ constant at 0.01. The dynamic updating scheme for μ also did not work for FISTA, which returned a very suboptimal solution in this case. We instead adopted a simple scheme of decreasing μ by half every 10 outer iterations. Figure 6 graphically depicts the performance of the different algorithms. In terms of the outer iterations, APLM-S behaved identically to FISTA-p, and FISTA also behaved similarly to ADAL. However, APLM-S and FISTA were considerably slower due to larger numbers of inner iterations.

We plot the root-mean-squared-error (RMSE) over different values of λ (which lead to different numbers of active genes) in the left half of Figure 4. The training set consists of 200 randomly selected samples, and the RMSE was computed on the remaining 95 samples. l_1/l_2-regularization achieved lower RMSE in this case. However, l_1/l_∞-regularization yielded better group sparsity as shown in Figure 5. The sets of active genes selected by the two models were very similar as illustrated in the right half of Figure 4. In general, the magnitudes of the coefficients returned by l_1/l_∞-regularization tended to be similar within a group, whereas those returned by l_1/l_2-regularization did not follow that pattern. This is because l_1/l_∞-regularization penalizes only the maximum element, rather than all the coefficients in a group, resulting in many coefficients having the same magnitudes.

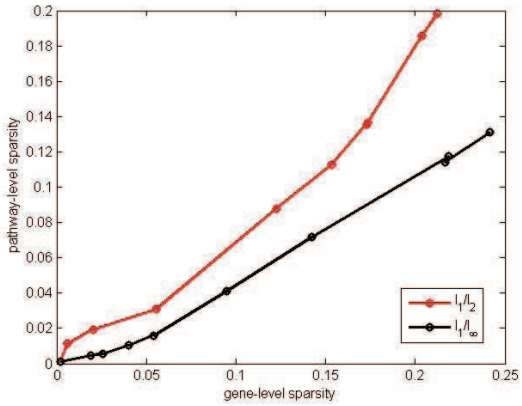


Figure 5: Pathway-level sparsity v.s. Gene-level sparsity.

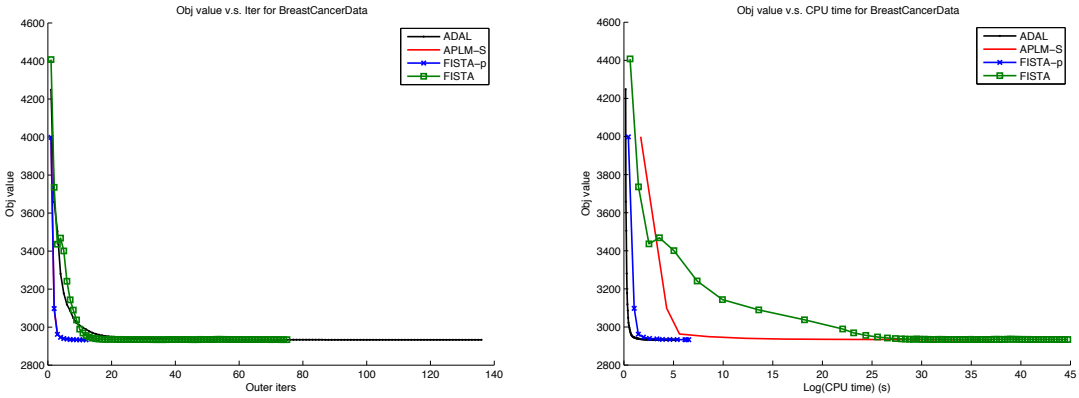


Figure 6: Objective values v.s. Outer iters and Objective values v.s. CPU time plots for the Breast Cancer data. The results for ProxGrad are not plotted due to the different objective function that it minimizes. The red (APLM-S) and blue (FISTA-p) lines overlap in the left column.

5.4.2 VIDEO SEQUENCE BACKGROUND SUBTRACTION

We next considered the video sequence background subtraction task from Mairal et al. (2010) and Huang et al. (2009). The main objective here is to segment out foreground objects in an image (frame), given a sequence of m frames from a fixed camera. The data used in this experiment is available online [5] (Toyama et al., 1999). The basic setup of the problem is as follows. We represent each frame of n pixels as a column vector $A_j \in \mathbb{R}^n$ and form the matrix $A \in \mathbb{R}^{n\times m}$ as $A \equiv \begin{pmatrix} A_1 & A_2 & \cdots & A_m \end{pmatrix}$. The test frame is represented by $b \in \mathbb{R}^n$. We model the relationship between b and A by $b \approx Ax + e$, where x is assumed to be sparse, and e is the 'noise' term which is also assumed to be sparse. Ax is thus a sparse linear combination of the video frame sequence and

5. Data can be found at `http://research.microsoft.com/en-us/um/people/jckrumm/wallflower/testimages.htm`.

accounts for the background present in both A and b. e contains the sparse foreground objects in b. The basic model with l_1-regularization (Lasso) is

$$\min_{x,e} \frac{1}{2}\|Ax+e-b\|^2+\lambda(\|x\|_1+\|e\|_1). \tag{31}$$

It has been shown in Mairal et al. (2010) that we can significantly improve the quality of segmentation by applying a group-structured regularization $\Omega(\cdot)$ on e, where the groups are all the overlapping $k\times k$-square patches in the image. Here, we set $k=3$. The model thus becomes

$$\min_{x,e} \frac{1}{2}\|Ax+e-b\|^2+\lambda(\|x\|_1+\|e\|_1+\Omega(e)). \tag{32}$$

Note that (32) still fits into the group-sparse framework if we treat the l_1-regularization terms as the sum of the group norms, where the each groups consists of only one element.

We also considered an alternative model, where a Ridge regularization is applied to x and an Elastic-Net penalty (Zou and Hastie, 2005) to e. This model

$$\min_{x,e} \frac{1}{2}\|Ax+e-b\|^2+\lambda_1\|e\|_1+\lambda_2(\|x\|^2+\|e\|^2) \tag{33}$$

does not yield a sparse x, but sparsity in x is not a crucial factor here. It is, however, well suited for our partial linearization methods (APLM-S and FISTA-p), since there is no need for the augmented Lagrangian framework. Of course, we can also apply FISTA to solve (33).

We recovered the foreground objects by solving the above optimization problems and applying the sparsity pattern of e as a mask for the original test frame. A hand-segmented evaluation image from Toyama et al. (1999) served as the ground truth. The regularization parameters λ,λ_1, and λ_2 were selected in such a way that the recovered foreground objects matched the ground truth to the maximum extent.

FISTA-p was used to solve all three models. The l_1 model (31) was treated as a special case of the group regularization model (32), with each group containing only one component of the feature vector.[6] For the Ridge/Elastic-Net penalty model, we applied FISTA-p directly without the outer augmented Lagrangian layer.

The solutions for the $l_1/l_2, l_1/l_\infty$, and Lasso models were not strictly sparse in the sense that those supposedly zero feature coefficients had non-zero (albeit extremely small) magnitudes, since we enforced the linear constraints $Cx=y$ through an augmented Lagrangian approach. To obtain sparse solutions, we truncated the non-sparse solutions using thresholds ranging from 10^{-9} to 10^{-3} and selected the threshold that yielded the best accuracy.

Note that because of the additional feature vector e, the data matrix is effectively $\tilde{A}=\begin{pmatrix} A & I_n \end{pmatrix}\in \mathbb{R}^{n\times(m+n)}$. For solving (32), FISTA-p has to solve the linear system

$$\begin{pmatrix} A^TA+\frac{1}{\mu}D_x & A^T \\ A & I_n+\frac{1}{\mu}D_e \end{pmatrix}\begin{pmatrix} x \\ e \end{pmatrix}=\begin{pmatrix} r_x \\ r_e \end{pmatrix},$$

where D is a diagonal matrix, and D_x, D_e, r_x, r_e are the components of D and r corresponding to x and e respectively. In this example, n is much larger than m, for example, $n=57600, m=200$. To

6. We did not use the original version of FISTA to solve the model as an l_1-regularization problem because it took too long to converge in our experiments due to extremely small step sizes.

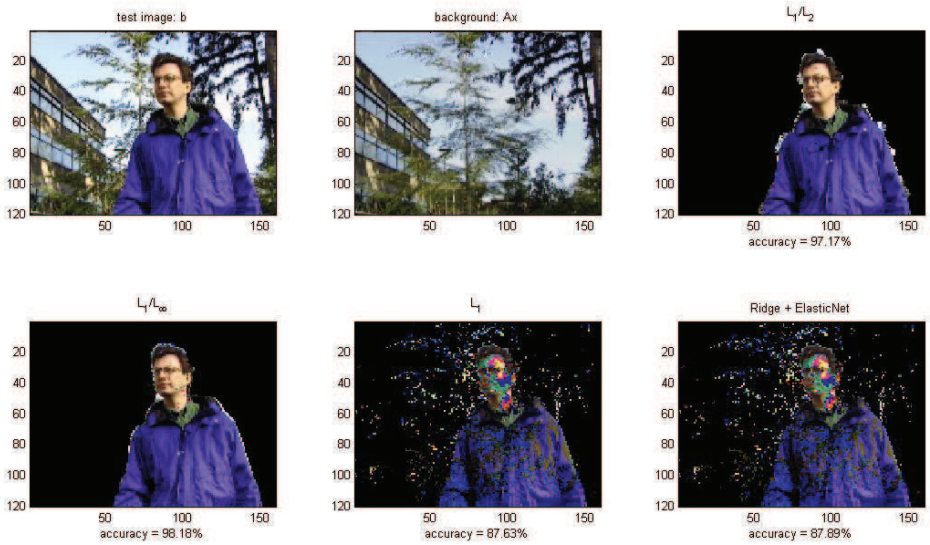


Figure 7: Separation results for the video sequence background substraction example. Each training image had 120×160 RGB pixels. The training set contained 200 images in sequence. The accuracy indicated for each of the different models is the percentage of pixels that matched the ground truth.

avoid solving a system of size $n \times n$, we took the Schur complement of $I_n + \frac{1}{\mu} D_e$ and solved instead the positive definite $m \times m$ system

$$\begin{aligned} \left(A^T A + \frac{1}{\mu} D_x - A^T (I + \frac{1}{\mu} D_e)^{-1} A \right) x &= r_x - A^T (I + \frac{1}{\mu} D_e)^{-1} r_e, \\ e &= diag(\mathbf{1} + \frac{1}{\mu} D_e)^{-1} (r_e - Ax). \end{aligned}$$

The l_1/l_∞ model yielded the best background separation accuracy (marginally better than the l_1/l_2 model), but it also was the most computationally expensive. (See Table 3 and Figure 7.) Although the Ridge/Elastic-Net model yielded as poor separation results as the Lasso (l_1) model, it was orders of magnitude faster to solve using FISTA-p. We again observed that the dynamic scheme for μ worked better for FISTA-p than for ADAL. For a constant μ over the entire run, ADAL took at least twice as long as FISTA-p to produce a solution of the same quality. A typical run of FISTA-p on this problem with the best selected λ took less than 10 outer iterations. On the other hand, ADAL took more than 500 iterations to meet the stopping criteria.

5.5 Comments on Results

The computational results exhibit two general patterns. First, the simpler algorithms (FISTA-p and ADAL) were significantly faster than the more general algorithms, such as APLM-S. Interestingly, the majority of the APLM-S inner iterations consisted of a skipping step for the tests on synthetic

Model	Accuracy (percent)	Total CPU time (s)	No. parameter values on reg path
l_1/l_2	97.17	2.48e+003	8
l_1/l_∞	98.18	4.07e+003	6
l_1	87.63	1.61e+003	11
ridge + elastic net	87.89	1.82e+002	64

Table 3: Computational results for the video sequence background subtraction example. The algorithm used is FISTA-p. We used the Matlab version for the ease of generating the images. The C++ version runs at least four times faster from our experience in the previous experiments. We report the best accuracy found on the regularization path of each model. The total CPU time is recorded for computing the entire regularization path, with the specified number of different regularization parameter values.

data and the breast cancer data, which means that APLM-S essentially behaved like ISTA-p in these cases. Indeed, FISTA-p generally required the same number of outer-iterations as APLM-S but much fewer inner-iterations, as predicted by theory. In addition, no computational steps were wasted and no function evaluations were required for FISTA-p and ADAL. Second, FISTA-p converged faster (required less iterations) than its full-linearization counterpart FISTA. We have suggested possible reasons for this in Section 3. On the other hand, FISTA was very effective for data both of whose dimensions were large because it required only gradient computations and soft-thresholding operations, and did not require linear systems to be solved.

Our experiments showed that the performance of ADAL (as well as the quality of the solution that it returned) varied a lot as a function of the parameter settings, and it was tricky to tune them optimally. In contrast, FISTA-p exhibited fairly stable performance for a simple set of parameters that we rarely had to alter and in general performed better than ADAL.

It may seem straight-forward to apply FISTA directly to the Lasso problem (31) without the augmented Lagrangian framework.[7] However, as we have seen in our experiments, FISTA took much longer than AugLag-FISTA-p to solve this problem. We believe that this is further evidence of the 'load-balancing' property of the latter algorithm that we discussed in Section 3.2. It also demonstrates the versatility of our approach to regularized learning problems.

6. Conclusion

We have built a unified framework for solving sparse learning problems involving group-structured regularization, in particular, the l_1/l_2- or l_1/l_∞-regularization of arbitrarily overlapping groups of variables. For the key building-block of this framework, we developed new efficient algorithms based on alternating partial-linearization/splitting, with proven convergence rates. In addition, we have also incorporated ADAL and FISTA into our framework. Computational tests on several sets of synthetic test data demonstrated the relative strength of the algorithms, and through two real-world applications we compared the relative merits of these structured sparsity-inducing norms. Among the algorithms studied, FISTA-p and ADAL performed the best on most of the data sets, and FISTA

7. To avoid confusion with our algorithms that consist of inner-outer iterations, we prefix our algorithms with 'AugLag' here.

appeared to be a good alternative choice for large-scale data. From our experience, FISTA-p is easier to configure and is more robust to variations in the algorithm parameters. Together, they form a flexible and versatile suite of methods for group-sparse problems of different sizes.

Acknowledgments

We would like to thank Katya Scheinberg and Shiqian Ma for many helpful discussions, and Xi Chen for providing the Matlab code for ProxGrad. We also thank the three anonymous reviewers for their valuable suggestions and comments. This research was supported in part by NSF Grant DMS 10-16571, ONR Grant N00014-08-1-1118 and DOE Grant DE-FG02-08ER25856.

Appendix A. Proof of Lemma 1

$$\begin{aligned} F(x,y)-F(x,q) &\geq F(x,y)-\mathcal{L}_\rho(x,y,q,\nabla_y f(x,y)) \\ &= F(x,y)-\left(f(x,y)+\nabla_y f(x,y)^T(q-y)+\frac{1}{2\rho}\|q-y\|^2+g(q)\right). \end{aligned} \tag{34}$$

From the optimality of q, we also have

$$\gamma_g(q)+\nabla_y f(x,y)+\frac{1}{\rho}(q-y)=0. \tag{35}$$

Since $F(x,y)=f(x,y)+g(y)$, and f and g are convex functions, for any (x,y),

$$F(x,y)\geq g(q)+(y-q)^T\gamma_g(q)+f(x,y)+(y-y)^T\nabla_y f(x,y)+(x-x)^T\nabla_x f(x,y). \tag{36}$$

Therefore, from (34), (35), and (36), it follows that

$$\begin{aligned} F(x,y)-F(x,q) &\geq g(q)+(y-q)^T\gamma_g(q)+f(x,y)+(y-y)^T\nabla_y f(x,y) \\ &\quad +(x-x)^T\nabla_x f(x,y) \\ &\quad -\left(f(x,y)+\nabla_y f(x,y)^T(q-y)+\frac{1}{2\rho}\|q-y\|^2+g(q)\right) \\ &= (y-q)^T(\gamma_g(q)+\nabla_y f(x,y))-\frac{1}{2\rho}\|q-y\|^2+(x-x)^T\nabla_x f(x,y) \\ &= (y-q)^T\left(-\frac{1}{\rho}(q-y)\right)-\frac{1}{2\rho}\|q-y\|^2+(x-x)^T\nabla_x f(x,y) \\ &= \frac{1}{2\rho}(\|q-y\|^2-\|y-y\|^2)+(x-x)^T\nabla_x f(x,y). \end{aligned}$$

The proof for the second part of the lemma is very similar, but we give it for completeness.

$$F(x,y)-F(p,q)\geq F(x,y)-\left(f(p,q)+g(y)+\gamma_g(y)^T(q-y)+\frac{1}{2\rho}\|q-y\|^2\right) \tag{37}$$

By the optimality of (p,q), we have

$$\nabla_x f(p,q) = 0, \tag{38}$$

$$\nabla_y f(p,q)+\gamma_g(y)+\frac{1}{\rho}(q-y) = 0. \tag{39}$$

Since $F(x,y) = f(x,y) + g(y)$, it follows from the convexity of both f and g and (38) that

$$F(x,y) \geq g(y) + (y-y)^T \gamma_g(y) + f(p,q) + (y-q)^T \nabla_y f(p,q). \tag{40}$$

Now combining (37), (39), and (40), it follows that

$$\begin{aligned} F(x,y) - F(p,q) &\geq (y-q)^T (\gamma_g(y) + \nabla_y f(p,q)) - \frac{1}{2\rho}\|q-y\|^2 \\ &= (y-q)^T \left(\frac{1}{\rho}(y-q)\right) - \frac{1}{2\rho}\|q-y\|^2 \\ &= \frac{1}{2\rho}(\|q-y\|^2 - \|y-y\|^2). \end{aligned}$$

Appendix B. Proof of Theorem 2

Let I be the set of all regular iteration indices among the first $k-1$ iterations, and let I_c be its complement. For all $n \in I_c, y^{n+1} = y^n$.

For $n \in I$, we can apply Lemma 1 since (18) automatically holds, and (16) holds when $\rho \leq \frac{1}{L(f)}$. In (19), by letting $(x,y) = (x^*, y^*)$, and $y = y^n$, we get $(p,q) = (x^{n+1}, y^{n+1})$, and

$$2\rho(F(x^*,y^*) - F(x^{n+1},y^{n+1})) \geq \|y^{n+1} - y^*\|^2 - \|y^n - y^*\|^2. \tag{41}$$

In (17), by letting $(x,y) = (x^*,y^*), (x,y) = (x^{n+1},y^{n+1})$, we get $q = y^{n+1}$ and

$$\begin{aligned} 2\rho(F(x^*,y^*) - F(x^{n+1},y^{n+1})) &\geq \|y^{n+1} - y^*\|^2 - \|y^{n+1} - y^*\|^2 \\ &\quad + (x^* - x^{n+1})^T \nabla_x f(x^{n+1}, y^{n+1}) \\ &= \|y^{n+1} - y^*\|^2 - \|y^{n+1} - y^*\|^2., \end{aligned} \tag{42}$$

since $\nabla_x f(x^{n+1}, y^{n+1}) = 0$, for $n \in I$ by (38) and for $n \in I_c$ by (15). Adding (42) to (41), we get

$$2\rho(2F(x^*,y^*) - F(x^{n+1},y^{n+1}) - F(x^{n+1},y^{n+1})) \geq \|y^{n+1} - y^*\|^2 - \|y^n - y^*\|^2. \tag{43}$$

For $n \in I_c$, since $\nabla_x f(x^{n+1}, y^{n+1}) = 0$, we have that (42) holds. Since $y^{n+1} = y^n$, it follows that

$$2\rho(F(x^*,y^*) - F(x^{n+1},y^{n+1})) \geq \|y^{n+1} - y^*\|^2 - \|y^n - y^*\|^2. \tag{44}$$

Summing (43) and (44) over $n = 0, 1, \ldots, k-1$ and observing that $2|I| + |I_c| = k + k_n$, we obtain

$$\begin{aligned} & 2\rho\left((k+k_n)F(x^*,y^*) - \sum_{n=1}^{k-1} F(x^{n+1},y^{n+1}) - \sum_{n\in I} F(x^{n+1},y^{n+1})\right) \qquad (45) \\ \geq & \sum_{n=0}^{k-1} (\|y^{n+1} - y^*\|^2 - \|y^n - y^*\|^2) \\ = & \|y^k - y^*\|^2 - \|y^0 - y^*\|^2 \\ \geq & -\|y^0 - y^*\|^2. \end{aligned}$$

In Lemma 1, by letting $(x,y) = (x^{n+1}, y^{n+1})$ in (17) instead of (x^*, y^*), we have from (42) that

$$2\rho(F(x^{n+1},y^{n+1}) - F(x^{n+1},y^{n+1})) \geq \|y^{n+1} - y^{n+1}\|^2 \geq 0. \tag{46}$$

Similarly, for $n \in I$, if we let $(x,y) = (x^n, y^n)$ instead of (x^*, y^*) in (41), we have

$$2\rho(F(x^n,y^n) - F(x^{n+1},y^{n+1})) \geq \|y^{n+1} - y^n\|^2 \geq 0. \tag{47}$$

For $n \in I_c$, $y^{n+1} = y^n$; from (15), since $x^{n+1} = \arg\min_x F(x,y)$ with $y = y^n = y^{n+1}$,

$$2\rho(F(x^n,y^n) - F(x^{n+1},y^{n+1})) \geq 0. \tag{48}$$

Hence, from (46) and (47) to (48), $F(x^n,y^n) \geq F(x^n,y^n) \geq F(x^{n+1},y^{n+1}) \geq F(x^{n+1},y^{n+1})$. Then, we have

$$\sum_{n=0}^{k-1} F(x^{n+1},y^{n+1}) \geq kF(x^k,y^k), \text{and} \quad \sum_{n\in I} F(x^{n+1},y^{n+1}) \geq k_n F(x^k,y^k). \tag{49}$$

Combining (45) and (49) yields $2\rho(k+k_n)(F(x^*,y^*) - F(x^k,y^k)) \geq -\|y^0 - y^*\|^2$.

Appendix C. Derivation of the Stopping Criteria

In this section, we show that the quantities that we use in our stopping criteria correspond to the primal and dual residuals (Boyd et al., 2010) for the outer iterations and the gradient residuals for the inner iterations. We first consider the inner iterations.

FISTA-p The necessary and sufficient optimality conditions for problem (10) or (13) are primal feasibility

$$y^* - y^* = 0, \tag{50}$$

and vanishing of the gradient of the objective function at (x^*, y^*), that is,

$$\begin{aligned} 0 &= \nabla_x f(x^*,y^*), & (51)\\ 0 &\in \nabla_y f(x^*,y^*) + \partial g(y^*). & (52) \end{aligned}$$

Since $y^{k+1} = z^k$, the primal residual is thus $y^{k+1} - y^{k+1} = y^{k+1} - z^k$. It follows from the optimality of x^{k+1} in Line 3 of Algorithm 6 that

$$\begin{aligned} &A^T(Ax^{k+1} - b) - C^T v^l + \frac{1}{\mu}C^T(Cx^{k+1} - y^{k+1}) + \frac{1}{\mu}C^T(y^{k+1} - z^k) = 0\\ &\Rightarrow \nabla_x f(x^{k+1},y^{k+1}) = \frac{1}{\mu}C^T(z^k - y^{k+1}). \end{aligned}$$

Similarly, from the optimality of y^{k+1} in Line 4, we have that

$$\begin{aligned} 0 &\in \partial g(y^{k+1}) + \nabla_y f(x^{k+1},z^k) + \frac{1}{\rho}(y^{k+1} - z^k)\\ &= \partial g(y^{k+1}) + \nabla_y f(x^{k+1},y^{k+1}) - \frac{1}{\mu}(y^{k+1} - z^k) + \frac{1}{\rho}(y^{k+1} - z^k)\\ &= \partial g(y^{k+1}) + \nabla_y f(x^{k+1},y^{k+1}), \end{aligned}$$

where the last step follows from $\mu = \rho$. Hence, we see that $\frac{1}{\mu}C^T(z^k - y^{k+1})$ is the gradient residual corresponding to (51), while (52) is satisfied in every inner iteration.

APLM-S The primal residual is $y^{k+1} - y^{k+1}$ from (50). Following the derivation for FISTA-p, it is not hard to verify that (52) is always satisfied, and the gradient residual corresponding to (51) is $\frac{1}{\mu}C^T(y^{k+1} - y^{k+1})$.

FISTA Similar to FISTA-p, the necessary and sufficient optimality conditions for problem (22) are primal feasibility

$$(x^*, y^*) = (x^*, y^*),$$

and vanishing of the objective gradient at (x^*, y^*),

$$\begin{aligned} 0 &= \nabla_x f(x^*, y^*), \\ 0 &\in \nabla_y f(x^*, y^*) + \partial g(y^*). \end{aligned}$$

Clearly, the primal residual is $(x^{k+1} - z_x^k, y^{k+1} - z_y^k)$ since $(x^{k+1}, y^{k+1}) \equiv (z_x^k, z_y^k)$. From the optimality of (x^{k+1}, y^{k+1}), it follows that

$$\begin{aligned} 0 &= \nabla_x f(z_x^k, z_y^k) + \frac{1}{\rho}(x^{k+1} - z_x^k), \\ 0 &\in \partial g(y^{k+1}) + \nabla_y f(z_x^k, z_y^k) + \frac{1}{\rho}(y^{k+1} - z_y^k). \end{aligned}$$

Here, we simply use $\frac{1}{\rho}(x^{k+1} - z_x^k)$ and $\frac{1}{\rho}(y^{k+1} - z_y^k)$ to approximate the gradient residuals.

Next, we consider the outer iterations. The necessary and sufficient optimality conditions for problem (4) are primal feasibility

$$Cx^* - y^* = 0,$$

and dual feasibility

$$\begin{aligned} 0 &= \nabla L(x^*) - C^T v^*, \\ 0 &\in \partial\tilde{\Omega}(y^*) + v^*. \end{aligned}$$

Clearly, the primal residual is $r^l = Cx^l - y^l$. The dual residual is $\begin{pmatrix} \nabla L(x^{l+1}) - C^T(v^l - \frac{1}{\mu}(Cx^{l+1} - y^{l+1})) \\ \partial\tilde{\Omega}(y^{l+1}) + v^l - \frac{1}{\mu}(Cx^{l+1} - y^{l+1}) \end{pmatrix}$, recalling that $v^{l+1} = v^l - \frac{1}{\mu}(Cx^{l+1} - y^{l+1})$. The above is simply the gradient of the augmented Lagrangian (5) evaluated at (x^l, y^l, v^l). Now, since the objective function of an inner iteration is the augmented Lagrangian with $v = v^l$, the dual residual for an outer iteration is readily available from the gradient residual computed for the last inner iteration of the outer iteration.

Appendix D. Numerical Results

See Tables 4 to 10.

References

M.V. Afonso, J.M. Bioucas-Dias, and M.A.T. Figueiredo. An augmented Lagrangian approach to the constrained optimization formulation of imaging inverse problems. *Image Processing, IEEE Transactions on*, (99):1–1, 2009. ISSN 1057-7149.

Data Sets	Algs	CPU (s)	Iters	Avg Sub-iters	$F(x)$
ogl-5000-100-10-3	ADAL	1.70e+000	61	1.00e+000	1.9482e+005
	APLM-S	1.71e+000	8	4.88e+000	1.9482e+005
	FISTA-p	9.08e-001	8	4.38e+000	1.9482e+005
	FISTA	2.74e+000	10	7.30e+000	1.9482e+005
	ProxGrad	7.92e+001	3858	-	-
ogl-5000-600-10-3	ADAL	6.75e+001	105	1.00e+000	1.4603e+006
	APLM-S	1.79e+002	9	1.74e+001	1.4603e+006
	FISTA-p	4.77e+001	9	8.56e+000	1.4603e+006
	FISTA	3.28e+001	12	1.36e+001	1.4603e+006
	ProxGrad	7.96e+002	5608	-	-
ogl-5000-1000-10-3	ADAL	2.83e+002	151	1.00e+000	2.6746e+006
	APLM-S	8.06e+002	10	2.76e+001	2.6746e+006
	FISTA-p	2.49e+002	10	1.28e+001	2.6746e+006
	FISTA	5.21e+001	13	1.55e+001	2.6746e+006
	ProxGrad	1.64e+003	6471	-	-

Table 4: Numerical results for ogl set 1. For ProxGrad, Avg Sub-Iters and $F(x)$ fields are not applicable since the algorithm is not based on an outer-inner iteration scheme, and the objective function that it minimizes is different from ours. We tested ten problems with $J = 100, \cdots, 1000$, but only show the results for three of them to save space.

Data Sets	Algs	CPU (s)	Iters	Avg Sub-iters	$F(x)$
ogl-1000-200-10-3	ADAL	4.18e+000	77	1.00e+000	9.6155e+004
	APLM-S	1.64e+001	9	2.32e+001	9.6156e+004
	FISTA-p	3.85e+000	9	1.02e+001	9.6156e+004
	FISTA	2.92e+000	11	1.44e+001	9.6158e+004
	ProxGrad	1.16e+002	4137	-	-
ogl-5000-200-10-3	ADAL	5.04e+000	63	1.00e+000	4.1573e+005
	APLM-S	8.42e+000	8	8.38e+000	4.1576e+005
	FISTA-p	3.96e+000	9	6.56e+000	4.1572e+005
	FISTA	6.54e+000	10	9.70e+000	4.1573e+005
	ProxGrad	1.68e+002	4345	-	-
ogl-10000-200-10-3	ADAL	6.41e+000	44	1.00e+000	1.0026e+006
	APLM-S	1.46e+001	10	7.60e+000	1.0026e+006
	FISTA-p	5.60e+000	10	5.50e+000	1.0026e+006
	FISTA	1.09e+001	10	8.50e+000	1.0027e+006
	ProxGrad	3.31e+002	6186	-	-

Table 5: Numerical results for ogl set 2. We ran the test for ten problems with $n = 1000, \cdots, 10000$, but only show the results for three of them to save space.

Data Sets	Algs	CPU (s)	Iters	Avg Sub-iters	$F(x)$
ogl-dct-1000-5000-1	ADAL	1.14e+001	194	1.00e+000	8.4892e+002
	FISTA-p	1.21e+001	20	1.11e+001	8.4892e+002
	FISTA	2.49e+001	24	2.51e+001	8.4893e+002
ogl-dct-1000-10000-1	ADAL	3.31e+001	398	1.00e+000	1.4887e+003
	FISTA-p	2.54e+001	41	5.61e+000	1.4887e+003
	FISTA	6.33e+001	44	1.74e+001	1.4887e+003
ogl-dct-1000-15000-1	ADAL	6.09e+001	515	1.00e+000	2.7506e+003
	FISTA-p	3.95e+001	52	4.44e+000	2.7506e+003
	FISTA	9.73e+001	54	1.32e+001	2.7506e+003
ogl-dct-1000-20000-1	ADAL	9.52e+001	626	1.00e+000	3.3415e+003
	FISTA-p	6.66e+001	63	6.10e+000	3.3415e+003
	FISTA	1.81e+002	64	1.61e+001	3.3415e+003
ogl-dct-1000-25000-1	ADAL	1.54e+002	882	1.00e+000	4.1987e+003
	FISTA-p	7.50e+001	88	3.20e+000	4.1987e+003
	FISTA	1.76e+002	89	8.64e+000	4.1987e+003
ogl-dct-1000-30000-1	ADAL	1.87e+002	957	1.00e+000	4.6111e+003
	FISTA-p	8.79e+001	96	2.86e+000	4.6111e+003
	FISTA	2.24e+002	94	8.54e+000	4.6111e+003

Table 6: Numerical results for dct set 2 (scalability test) with l_1/l_2-regularization. All three algorithms were ran in factorization mode with a fixed $\mu = \mu_0$.

F. Bach. Consistency of the group lasso and multiple kernel learning. *The Journal of Machine Learning Research*, 9:1179–1225, 2008.

F. Bach. Structured sparsity-inducing norms through submodular functions. In J. Lafferty, C. K. I. Williams, J. Shawe-Taylor, R.S. Zemel, and A. Culotta, editors, *Advances in Neural Information Processing Systems 23*, pages 118–126. 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. Multiplier methods: a survey. *Automatica*, 12(2):133–145, 1976.

D.P. Bertsekas. *Nonlinear Programming*. Athena Scientific Belmont, MA, 1999. ISBN 1886529000.

D.P. Bertsekas and J.N. Tsitsiklis. *Parallel and Distributed Computation: Numerical Methods*. Prentice-Hall, Inc., 1989.

S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Machine Learning*, 3(1):1–123, 2010.

P. Brucker. An O(n) algorithm for quadratic knapsack problems. *Operations Research Letters*, 3 (3):163–166, 1984.

Data Sets	Algs	CPU (s)	Iters	Avg Sub-iters	$F(x)$
ogl-dct-1000-5000-1	ADAL	1.53e+001	266	1.00e+000	7.3218e+002
	FISTA-p	1.61e+001	10	3.05e+001	7.3219e+002
	FISTA	3.02e+001	16	4.09e+001	7.3233e+002
	ProxFlow	1.97e+001	-	-	7.3236e+002
ogl-dct-1000-10000-1	ADAL	3.30e+001	330	1.00e+000	1.2707e+003
	FISTA-p	3.16e+001	10	3.10e+001	1.2708e+003
	FISTA	7.27e+001	24	3.25e+001	1.2708e+003
	ProxFlow	3.67e+001	-	-	1.2709e+003
ogl-dct-1000-15000-1	ADAL	4.83e+001	328	1.00e+000	2.2444e+003
	FISTA-p	5.40e+001	15	2.52e+001	2.2444e+003
	FISTA	8.64e+001	23	2.66e+001	2.2449e+003
	ProxFlow	9.91e+001	-	-	2.2467e+003
ogl-dct-1000-20000-1	ADAL	8.09e+001	463	1.00e+000	2.6340e+003
	FISTA-p	8.09e+001	16	2.88e+001	2.6340e+003
	FISTA	1.48e+002	26	2.93e+001	2.6342e+003
	ProxFlow	2.55e+002	-	-	2.6357e+003
ogl-dct-1000-25000-1	ADAL	7.48e+001	309	1.00e+000	3.5566e+003
	FISTA-p	1.15e+002	30	1.83e+001	3.5566e+003
	FISTA	2.09e+002	38	2.30e+001	3.5568e+003
	ProxFlow	1.38e+002	-	-	3.5571e+003
ogl-dct-1000-30000-1	ADAL	9.99e+001	359	1.00e+000	3.7057e+003
	FISTA-p	1.55e+002	29	2.17e+001	3.7057e+003
	FISTA	2.60e+002	39	2.25e+001	3.7060e+003
	ProxFlow	1.07e+002	-	-	3.7063e+003

Table 7: Numerical results for dct set 2 (scalability test) with l_1/l_∞-regularization. The algorithm configurations are exactly the same as in Table 6.

S.S. Chen, D.L. Donoho, and M.A. Saunders. Atomic decomposition by basis pursuit. *SIAM Journal on Scientific Computing*, 20(1):33–61, 1999. ISSN 1064-8275.

X. Chen, Q. Lin, S. Kim, J. Peña, J.G. Carbonell, and E.P. Xing. An efficient proximal-gradient method for single and multi-task regression with structured sparsity. *Arxiv Preprint arXiv:1005.4717*, 2010.

P.L. Combettes and J.C. Pesquet. Proximal splitting methods in signal processing. *Fixed-Point Algorithms for Inverse Problems in Science and Engineering*, pages 185–212, 2011.

P.L. Combettes and V.R. Wajs. Signal recovery by proximal forward-backward splitting. *Multiscale Modeling and Simulation*, 4(4):1168–1200, 2006. ISSN 1540-3459.

Data Sets	Algs	CPU (s)	Iters	Avg Sub-iters	$F(x)$
ogl-dct-1000-5000-1	FISTA-p	1.83e+001	12	2.34e+001	8.4892e+002
	FISTA	2.49e+001	24	2.51e+001	8.4893e+002
	ADAL	1.35e+001	181	1.00e+000	8.4892e+002
ogl-dct-1000-10000-1	FISTA-p	3.16e+001	14	1.73e+001	1.4887e+003
	FISTA	6.33e+001	44	1.74e+001	1.4887e+003
	ADAL	4.43e+001	270	1.00e+000	1.4887e+003
ogl-dct-1000-15000-1	FISTA-p	4.29e+001	14	1.51e+001	2.7506e+003
	FISTA	9.73e+001	54	1.32e+001	2.7506e+003
	ADAL	5.37e+001	216	1.00e+000	2.7506e+003
ogl-dct-1000-20000-1	FISTA-p	7.53e+001	13	2.06e+001	3.3416e+003
	FISTA	1.81e+002	64	1.61e+001	3.3415e+003
	ADAL	1.57e+002	390	1.00e+000	3.3415e+003
ogl-dct-1000-25000-1	FISTA-p	7.41e+001	15	1.47e+001	4.1987e+003
	FISTA	1.76e+002	89	8.64e+000	4.1987e+003
	ADAL	8.79e+001	231	1.00e+000	4.1987e+003
ogl-dct-1000-30000-1	FISTA-p	8.95e+001	14	1.58e+001	4.6111e+003
	FISTA	2.24e+002	94	8.54e+000	4.6111e+003
	ADAL	1.12e+002	249	1.00e+000	4.6111e+003

Table 8: Numerical results for the DCT set with l_1/l_2-regularization. FISTA-p and ADAL were ran in PCG mode with the dynamic scheme for updating μ. μ was fixed at μ_0 for FISTA.

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 25th International Conference on Machine Learning*, pages 272–279. ACM, 2008.

J. Eckstein and D.P. Bertsekas. On the Douglas-Rachford splitting method and the proximal point algorithm for maximal monotone operators. *Mathematical Programming*, 55(1):293–318, 1992. ISSN 0025-5610.

J. Eckstein and P.J.S. Silva. A practical relative error criterion for augmented lagrangians. *Mathematical Programming*, pages 1–30, 2012.

D. Gabay and B. Mercier. A dual algorithm for the solution of nonlinear variational problems via finite element approximation. *Computers & Mathematics with Applications*, 2(1):17–40, 1976. ISSN 0898-1221.

R. Glowinski and A. Marroco. Sur l'approximation, par elements finis d'ordre un, et la resolution, par penalisation-dualite d'une classe de problemes de dirichlet non lineares. *Rev. Francaise d'Automat. Inf. Recherche Operationelle*, (9):41–76, 1975.

D. Goldfarb, S. Ma, and K. Scheinberg. Fast alternating linearization methods for minimizing the sum of two convex functions. *Mathematical Programming Series A*, 2011.

Data Sets	Algs	CPU (s)	Iters	Avg Sub-iters	$F(x)$
ogl-dct-1000-5000-1	FISTA-p	2.30e+001	11	2.93e+001	7.3219e+002
	ADAL	1.89e+001	265	1.00e+000	7.3218e+002
	FISTA	3.02e+001	16	4.09e+001	7.3233e+002
	ProxFlow	1.97e+001	-	-	7.3236e+002
ogl-dct-1000-10000-1	FISTA-p	5.09e+001	11	3.16e+001	1.2708e+003
	ADAL	4.77e+001	323	1.00e+000	1.2708e+003
	FISTA	7.27e+001	24	3.25e+001	1.2708e+003
	ProxFlow	3.67e+001	-	-	1.2709e+003
ogl-dct-1000-15000-1	FISTA-p	6.33e+001	12	2.48e+001	2.2445e+003
	ADAL	9.41e+001	333	1.00e+000	2.2444e+003
	FISTA	8.64e+001	23	2.66e+001	2.2449e+003
	ProxFlow	9.91e+001	-	-	2.2467e+003
ogl-dct-1000-20000-1	FISTA-p	8.21e+001	12	2.42e+001	2.6341e+003
	ADAL	1.59e+002	415	1.00e+000	2.6340e+003
	FISTA	1.48e+002	26	2.93e+001	2.6342e+003
	ProxFlow	2.55e+002	-	-	2.6357e+003
ogl-dct-1000-25000-1	FISTA-p	1.43e+002	13	2.98e+001	3.5567e+003
	ADAL	1.20e+002	310	1.00e+000	3.5566e+003
	FISTA	2.09e+002	38	2.30e+001	3.5568e+003
	ProxFlow	1.38e+002	-	-	3.5571e+003
ogl-dct-1000-30000-1	FISTA-p	1.75e+002	13	3.18e+001	3.7057e+003
	ADAL	2.01e+002	361	1.00e+000	3.7057e+003
	FISTA	2.60e+002	39	2.25e+001	3.7060e+003
	ProxFlow	1.07e+002	-	-	3.7063e+003

Table 9: Numerical results for the DCT set with l_1/l_∞-regularization. FISTA-p and ADAL were ran in PCG mode. The dynamic updating scheme for μ was applied to FISTA-p, while μ was fixed at μ_0 for ADAL and FISTA.

Data Sets	Algs	CPU (s)	Iters	Avg Sub-iters	$F(x)$
BreastCancerData	ADAL	6.24e+000	136	1.00e+000	2.9331e+003
	APLM-S	4.02e+001	12	4.55e+001	2.9331e+003
	FISTA-p	6.86e+000	12	1.48e+001	2.9331e+003
	FISTA	5.11e+001	75	1.29e+001	2.9340e+003
	ProxGrad	7.76e+002	6605	1.00e+000	-

Table 10: Numerical results for Breast Cancer Data using l_1/l_2-regularization. In this experiment, we kept μ constant at 0.01 for ADAL. The CPU time is for a single run on the entire data set with the value of λ selected to minimize the RMSE in Figure 4.

T. Goldstein and S. Osher. The split bregman method for l1-regularized problems. *SIAM Journal on Imaging Sciences*, 2:323, 2009.

G.H. Golub and C.F. Van Loan. *Matrix Computations*. Johns Hopkins Univ Pr, 1996. ISBN 0801854148.

M.R. Hestenes. Multiplier and gradient methods. *Journal of Optimization Theory and Applications*, 4(5):303–320, 1969.

J. Huang, T. Zhang, and D. Metaxas. Learning with structured sparsity. In *Proceedings of the 26th Annual International Conference on Machine Learning*, pages 417–424. ACM, 2009.

L. Jacob, G. Obozinski, and J.P. Vert. Group lasso with overlap and graph lasso. In *Proceedings of the 26th Annual International Conference on Machine Learning*, pages 433–440. ACM, 2009.

R. Jenatton, G. Obozinski, and F. Bach. Structured sparse principal component analysis. *AISTATS*, 2010.

R. Jenatton, J.Y. Audibert, F. Bach, et al. Structured variable selection with sparsity-inducing norms. *Journal of Machine Learning Research*, 12:2777–2824, 2011.

S. Kim and E.P. Xing. Tree-guided group lasso for multi-task regression with structured sparsity. In *Proceedings of the 27th Annual International Conference on Machine Learning*, 2010.

Z. Lin, M. Chen, L. Wu, and Y. Ma. The augmented lagrange multiplier method for exact recovery of corrupted low-rank matrices. *Arxiv Preprint arXiv:1009.5055*, 2010.

J. Liu and J. Ye. Fast overlapping group lasso. *Arxiv Preprint arXiv:1009.0306*, 2010.

J. Mairal, R. Jenatton, G. Obozinski, and F. Bach. Network flow algorithms for structured sparsity. In J. Lafferty, C. K. I. Williams, J. Shawe-Taylor, R.S. Zemel, and A. Culotta, editors, *Advances in Neural Information Processing Systems 23*, pages 1558–1566. 2010.

J. Mairal, R. Jenatton, G. Obozinski, and F. Bach. Convex and network flow optimization for structured sparsity. *The Journal of Machine Learning Research*, 12:2681–2720, 2011.

S. Mosci, S. Villa, A. Verri, and L. Rosasco. A primal-dual algorithm for group sparse regularization with overlapping groups. In *Neural Information Processing Systems*, 2010.

Y. Nesterov. Smooth minimization of non-smooth functions. *Mathematical Programming*, 103(1): 127–152, 2005. ISSN 0025-5610.

J. Nocedal and S.J. Wright. *Numerical Optimization*. Springer Verlag, 1999.

G. Obozinski, L. Jacob, and J.P. Vert. Group lasso with overlaps: the latent group lasso approach. *Arxiv Preprint arXiv:1110.0413*, 2011.

J.C. Pesquet and N. Pustelnik. A parallel inertial proximal optimization method. *Preprint*, 2010.

G. Peyre and J. Fadili. Group sparsity with overlapping partition functions. In *EUSIPCO 2011*, 2011.

G. Peyre, J. Fadili, and C. Chesneau. Adaptive structured block sparsity via dyadic partitioning. In *EUSIPCO 2011*, 2011.

M.J.D. Powell. A method for nonlinear constraints in minimization problems. In R. Fletcher, editor, *Optimization*. Academic Press, New York, New York, 1972.

Z. Qin, K. Scheinberg, and D. Goldfarb. Efficient block-coordinate descent algorithms for the group lasso. Technical report, Department of Industrial Engineering and Operations Research, Columbia University, 2010.

R.T. Rockafellar. The multiplier method of hestenes and powell applied to convex programming. *Journal of Optimization Theory and Applications*, 12(6):555–562, 1973.

V. Roth and B. Fischer. The group-lasso for generalized linear models: uniqueness of solutions and efficient algorithms. In *Proceedings of the 25th International Conference on Machine Learning*, pages 848–855. ACM, 2008.

S. Setzer, G. Steidl, and T. Teuber. Deblurring poissonian images by split bregman techniques. *Journal of Visual Communication and Image Representation*, 21(3):193–199, 2010.

J.E. Spingarn. Partial inverse of a monotone operator. *Applied Mathematics & Optimization*, 10(1): 247–265, 1983.

A. Subramanian, P. Tamayo, V.K. Mootha, S. Mukherjee, B.L. Ebert, M.A. Gillette, A. Paulovich, S.L. Pomeroy, T.R. Golub, E.S. Lander, et al. Gene set enrichment analysis: a knowledge-based approach for interpreting genome-wide expression profiles. *Proceedings of the National Academy of Sciences of the United States of America*, 102(43):15545, 2005.

R. Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society. Series B (Methodological)*, 58(1):267–288, 1996.

K. Toyama, J. Krumm, B. Brumitt, and B. Meyers. Wallflower: Principles and practice of background maintenance. In *ICCV*, page 255. Published by the IEEE Computer Society, 1999.

M.J. Van De Vijver, Y.D. He, L.J. van't Veer, H. Dai, A.A.M. Hart, D.W. Voskuil, G.J. Schreiber, J.L. Peterse, C. Roberts, M.J. Marton, et al. A gene-expression signature as a predictor of survival in breast cancer. *New England Journal of Medicine*, 347(25):1999, 2002.

S.J. Wright, R.D. Nowak, and M.A.T. Figueiredo. Sparse reconstruction by separable approximation. *IEEE Transactions on Signal Processing*, 57(7):2479–2493, 2009.

M. Yuan and Y. Lin. Model selection and estimation in regression with grouped variables. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 68(1):49–67, 2006.

H. Zou and T. Hastie. Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 67(2):301–320, 2005. ISSN 1467-9868.

Journal of Machine Learning Research 13 (2012) 1469-1587 Submitted 8/11; Published 5/12

Activized Learning: Transforming Passive to Active with Improved Label Complexity*

Steve Hanneke SHANNEKE@STAT.CMU.EDU
Department of Statistics
Carnegie Mellon University
Pittsburgh, PA 15213 USA

Editor: Sanjoy Dasgupta

Abstract

We study the theoretical advantages of active learning over passive learning. Specifically, we prove that, in noise-free classifier learning for VC classes, any passive learning algorithm can be transformed into an active learning algorithm with asymptotically strictly superior label complexity for all nontrivial target functions and distributions. We further provide a general characterization of the magnitudes of these improvements in terms of a novel generalization of the disagreement coefficient. We also extend these results to active learning in the presence of label noise, and find that even under broad classes of noise distributions, we can typically guarantee strict improvements over the known results for passive learning.

Keywords: active learning, selective sampling, sequential design, statistical learning theory, PAC learning, sample complexity

1. Introduction and Background

The recent rapid growth in data sources has spawned an equally rapid expansion in the number of potential applications of machine learning methodologies to extract useful concepts from these data. However, in many cases, the bottleneck in the application process is the need to obtain accurate annotation of the raw data according to the target concept to be learned. For instance, in webpage classification, it is straightforward to rapidly collect a large number of webpages, but training an accurate classifier typically requires a human expert to examine and label a number of these webpages, which may require significant time and effort. For this reason, it is natural to look for ways to reduce the total number of labeled examples required to train an accurate classifier. In the traditional machine learning protocol, here referred to as *passive learning*, the examples labeled by the expert are sampled independently at random, and the emphasis is on designing learning algorithms that make the most effective use of the number of these labeled examples available. However, it is possible to go beyond such methods by altering the protocol itself, allowing the learning algorithm to sequentially *select* the examples to be labeled, based on its observations of the labels of previously-selected examples; this interactive protocol is referred to as *active learning*. The objective in designing this selection mechanism is to focus the expert's efforts toward labeling only the most informative data for the learning process, thus eliminating some degree of redundancy in the information content of the labeled examples.

*. Some of these (and related) results previously appeared in the author's doctoral dissertation (Hanneke, 2009b).

It is now well-established that active learning can sometimes provide significant practical and theoretical advantages over passive learning, in terms of the number of labels required to obtain a given accuracy. However, our current understanding of active learning in general is still quite limited in several respects. First, since we are lacking a complete understanding of the potential capabilities of active learning, we are not yet sure to what standards we should aspire for active learning algorithms to meet, and in particular this challenges our ability to characterize how a "good" active learning algorithm should behave. Second, since we have yet to identify a complete set of general principles for the design of effective active learning algorithms, in many cases the most effective known active learning algorithms have problem-specific designs (e.g., designed specifically for linear separators, or decision trees, etc., under specific assumptions on the data distribution), and it is not clear what components of their design can be abstracted and transferred to the design of active learning algorithms for different learning problems (e.g., with different types of classifiers, or different data distributions). Finally, we have yet to fully understand the scope of the relative benefits of active learning over passive learning, and in particular the conditions under which such improvements are achievable, as well as a general characterization of the potential magnitudes of these improvements. In the present work, we take steps toward closing this gap in our understanding of the capabilities, general principles, and advantages of active learning.

Additionally, this work has a second theme, motivated by practical concerns. To date, the machine learning community has invested decades of research into constructing solid, reliable, and well-behaved *passive* learning algorithms, and into understanding their theoretical properties. We might hope that an equivalent amount of effort is *not* required in order to discover and understand effective active learning algorithms. In particular, rather than starting from scratch in the design and analysis of active learning algorithms, it seems desirable to leverage this vast knowledge of passive learning, to whatever extent possible. For instance, it may be possible to design active learning algorithms that *inherit* certain desirable behaviors or properties of a given passive learning algorithm. In this way, we can use a given passive learning algorithm as a *reference point*, and the objective is to design an active learning algorithm with performance guarantees strictly superior to those of the passive algorithm. Thus, if the passive learning algorithm has proven effective in a variety of common learning problems, then the active learning algorithm should be even better for those *same* learning problems. This approach also has the advantage of immediately supplying us with a collection of theoretical guarantees on the performance of the active learning algorithm: namely, improved forms of all known guarantees on the performance of the given passive learning algorithm.

Due to its obvious practical advantages, this general line of informal thinking dominates the existing literature on empirically-tested heuristic approaches to active learning, as most of the published heuristic active learning algorithms make use of a passive learning algorithm as a subroutine (e.g., SVM, logistic regression, k-NN, etc.), constructing sets of labeled examples and feeding them into the passive learning algorithm at various times during the execution of the active learning algorithm (see the references in Section 7). Below, we take a more rigorous look at this general strategy. We develop a reduction-style framework for studying this approach to the design of active learning algorithms relative to a given passive learning algorithm. We then proceed to develop and analyze a variety of such methods, to realize this approach in a very general sense.

Specifically, we explore the following fundamental questions.

- Is there a general procedure that, given any passive learning algorithm, transforms it into an active learning algorithm requiring significantly fewer labels to achieve a given accuracy?
- If so, how large is the reduction in the number of labels required by the resulting active learning algorithm, compared to the number of labels required by the original passive algorithm?
- What are sufficient conditions for an *exponential* reduction in the number of labels required?
- To what extent can these methods be made robust to imperfect or noisy labels?

In the process of exploring these questions, we find that for many interesting learning problems, the techniques in the existing literature are not capable of realizing the full potential of active learning. Thus, exploring this topic in generality requires us to develop novel insights and entirely new techniques for the design of active learning algorithms. We also develop corresponding natural complexity quantities to characterize the performance of such algorithms. Several of the results we establish here are more general than any related results in the existing literature, and in many cases the algorithms we develop use significantly fewer labels than any previously published methods.

1.1 Background

The term *active learning* refers to a family of supervised learning protocols, characterized by the ability of the learning algorithm to pose queries to a teacher, who has access to the target concept to be learned. In practice, the teacher and queries may take a variety of forms: a human expert, in which case the queries may be questions or annotation tasks; nature, in which case the queries may be scientific experiments; a computer simulation, in which case the queries may be particular parameter values or initial conditions for the simulator; or a host of other possibilities. In our present context, we will specifically discuss a protocol known as *pool-based* active learning, a type of sequential design based on a collection of unlabeled examples; this seems to be the most common form of active learning in practical use today (e.g., Settles, 2010; Baldridge and Palmer, 2009; Gangadharaiah, Brown, and Carbonell, 2009; Hoi, Jin, Zhu, and Lyu, 2006; Luo, Kramer, Goldgof, Hall, Samson, Remsen, and Hopkins, 2005; Roy and McCallum, 2001; Tong and Koller, 2001; McCallum and Nigam, 1998). We will not discuss alternative models of active learning, such as *online* (Dekel, Gentile, and Sridharan, 2010) or *exact* (Hegedüs, 1995). In the pool-based active learning setting, the learning algorithm is supplied with a large collection of unlabeled examples (the *pool*), and is allowed to select any example from the pool to request that it be labeled. After observing the label of this example, the algorithm can then select another unlabeled example from the pool to request that it be labeled. This continues sequentially for a number of rounds until some halting condition is satisfied, at which time the algorithm returns a function intended to approximately mimic and generalize the observed labeling behavior. This setting contrasts with *passive learning*, where the learning algorithm is supplied with a collection of *labeled* examples without any interaction.

Supposing the labels received agree with some true target concept, the objective is to use this returned function to approximate the true target concept on future (previously unobserved) data points. The hope is that, by carefully selecting which examples should be labeled, the algorithm can achieve improved accuracy while using fewer labels compared to passive learning. The motivation for this setting is simple. For many modern machine learning problems, unlabeled examples are inexpensive and available in abundance, while annotation is time-consuming or expensive. For instance, this is the case in the aforementioned webpage classification problem, where the pool would

be the set of all webpages, and labeling a webpage requires a human expert to examine the website content. Settles (2010) surveys a variety of other applications for which active learning is presently being used. To simplify the discussion, in this work we focus specifically on *binary classification*, in which there are only two possible labels. The results generalize naturally to multiclass classification as well.

As the above description indicates, when studying the advantages of active learning, we are primarily interested in the number of label requests sufficient to achieve a given accuracy, a quantity referred to as the *label complexity* (Definition 1 below). Although active learning has been an active topic in the machine learning literature for many years now, our *theoretical* understanding of this topic was largely lacking until very recently. However, within the past few years, there has been an explosion of progress. These advances can be grouped into two categories: namely, the *realizable case* and the *agnostic case*.

1.1.1 THE REALIZABLE CASE

In the realizable case, we are interested in a particularly strict scenario, where the true label of any example is *determined* by a function of the features (covariates), and where that function has a specific known form (e.g., linear separator, decision tree, union of intervals, etc.); the set of classifiers having this known form is referred to as the *concept space*. The natural formalization of the realizable case is very much analogous to the well-known PAC model for passive learning (Valiant, 1984). In the realizable case, there are obvious examples of learning problems where active learning can provide a significant advantage compared to passive learning; for instance, in the problem of learning *threshold* classifiers on the real line (Example 1 below), a kind of *binary search* strategy for selecting which examples to request labels for naturally leads to *exponential* improvements in label complexity compared to learning from random labeled examples (passive learning). As such, there is a natural attraction to determine how general this phenomenon is. This leads us to think about general-purpose learning strategies (i.e., which can be instantiated for more than merely threshold classifiers on the real line), which exhibit this binary search behavior in various special cases.

The first such general-purpose strategy to emerge in the literature was a particularly elegant strategy proposed by Cohn, Atlas, and Ladner (1994), typically referred to as CAL after its discoverers (Meta-Algorithm 2 below). The strategy behind CAL is the following. The algorithm examines each example in the unlabeled pool in sequence, and if there are two classifiers in the concept space consistent with all previously-observed labels, but which disagree on the label of this next example, then the algorithm requests that label, and otherwise it does not. For this reason, below we refer to the general family of algorithms inspired by CAL as *disagreement-based* methods. Disagreement-based methods are sometimes referred to as "mellow" active learning, since in some sense this is the *least* we can expect from a reasonable active learning algorithm; it never requests the label of an example whose label it can *infer* from information already available, but otherwise makes no attempt to seek out particularly informative examples to request the labels of. That is, the notion of *informativeness* implicit in disagreement-based methods is a *binary* one, so that an example is either informative or not informative, but there is no further ranking of the informativeness of examples. The disagreement-based strategy is quite general, and obviously leads to algorithms that are at least *reasonable*, but Cohn, Atlas, and Ladner (1994) did not study the label complexity achieved by their strategy in any generality.

In a Bayesian variant of the realizable setting, Freund, Seung, Shamir, and Tishby (1997) studied an algorithm known as *query by committee* (QBC), which in some sense represents a Bayesian variant of CAL. However, QBC *does* distinguish between different levels of informativeness beyond simple disagreement, based on the *amount* of disagreement on a random unlabeled example. They were able to analyze the label complexity achieved by QBC in terms of a type of information gain, and found that when the information gain is lower bounded by a positive constant, the algorithm achieves a label complexity exponentially smaller than the known results for passive learning. In particular, this is the case for the threshold learning problem, and also for the problem of learning higher-dimensional (nearly balanced) linear separators when the data satisfy a certain (uniform) distribution. Below, we will not discuss this analysis further, since it is for a slightly different (Bayesian) setting. However, the results below in our present setting do have interesting implications for the Bayesian setting as well, as discussed in the recent work of Yang, Hanneke, and Carbonell (2011).

The first general analysis of the label complexity of active learning in the (non-Bayesian) realizable case came in the breakthrough work of Dasgupta (2005). In that work, Dasgupta proposed a quantity, called the *splitting index*, to characterize the label complexities achievable by active learning. The splitting index analysis is noteworthy for several reasons. First, one can show it provides nearly tight bounds on the *minimax* label complexity for a given concept space and data distribution. In particular, the analysis matches the exponential improvements known to be possible for threshold classifiers, as well as generalizations to higher-dimensional homogeneous linear separators under near-uniform distributions (as first established by Dasgupta, Kalai, and Monteleoni, 2005, 2009). Second, it provides a novel notion of *informativeness* of an example, beyond the simple binary notion of informativeness employed in disagreement-based methods. Specifically, it describes the informativeness of an example in terms of the number of *pairs* of well-separated classifiers for which at least one out of each pair will be contradicted, supposing the least-favorable label. Finally, unlike any other existing work on active learning (present work included), it provides an elegant description of the *trade-off* between the number of label requests and the number of unlabeled examples needed by the learning algorithm. Another interesting byproduct of Dasgupta's work is a better understanding of the *nature* of the improvements achievable by active learning in the general case. In particular, his work clearly illustrates the need to study the label complexity as a quantity that varies depending on the particular target concept and data distribution. We will see this issue arise in many of the examples below.

Coming from a slightly different perspective, Hanneke (2007a) later analyzed the label complexity of active learning in terms of an extension of the *teaching dimension* (Goldman and Kearns, 1995). Related quantities were previously used by Hegedüs (1995) and Hellerstein, Pillaipakkamnatt, Raghavan, and Wilkins (1996) to tightly characterize the number of membership queries sufficient for *Exact* learning; Hanneke (2007a) provided a natural generalization to the *PAC* learning setting. At this time, it is not clear how this quantity relates to the splitting index. From a practical perspective, in some instances it may be easier to calculate (see the work of Nowak, 2008 for a discussion related to this), though in other cases the opposite seems true.

The next progress toward understanding the label complexity of active learning came in the work of Hanneke (2007b), who introduced a quantity called the *disagreement coefficient* (Definition 9 below), accompanied by a technique for analyzing disagreement-based active learning algorithms. In particular, implicit in that work, and made explicit in the later work of Hanneke (2011), was the first general characterization of the label complexities achieved by the original CAL strategy for

active learning in the realizable case, stated in terms of the disagreement coefficient. The results of the present work are direct descendants of that 2007 paper, and we will discuss the disagreement coefficient, and results based on it, in substantial detail below. Disagreement-based active learners such as CAL are known to be sometimes suboptimal relative to the splitting index analysis, and therefore the disagreement coefficient analysis sometimes results in larger label complexity bounds than the splitting index analysis. However, in many cases the label complexity bounds based on the disagreement coefficient are surprisingly good considering the simplicity of the methods. Furthermore, as we will see below, the disagreement coefficient has the practical benefit of often being fairly straightforward to calculate for a variety of learning problems, particularly when there is a natural geometric interpretation of the classifiers and the data distribution is relatively smooth. As we discuss below, it can also be used to bound the label complexity of active learning in noisy settings. For these reasons (simplicity of algorithms, ease of calculation, and applicability beyond the realizable case), subsequent work on the label complexity of active learning has tended to favor the disagreement-based approach, making use of the disagreement coefficient to bound the label complexity (Dasgupta, Hsu, and Monteleoni, 2007; Friedman, 2009; Beygelzimer, Dasgupta, and Langford, 2009; Wang, 2009; Balcan, Hanneke, and Vaughan, 2010; Hanneke, 2011; Koltchinskii, 2010; Beygelzimer, Hsu, Langford, and Zhang, 2010; Mahalanabis, 2011; Wang, 2011). A significant part of the present paper focuses on extending and generalizing the disagreement coefficient analysis, while still maintaining the relative ease of calculation that makes the disagreement coefficient so useful.

In addition to many positive results, Dasgupta (2005) also pointed out several negative results, even for very simple and natural learning problems. In particular, for many problems, the minimax label complexity of active learning will be no better than that of passive learning. In fact, Balcan, Hanneke, and Vaughan (2010) later showed that, for a certain type of active learning algorithm—namely, *self-verifying* algorithms, which themselves adaptively determine how many label requests they need to achieve a given accuracy—there are even particular target concepts and data distributions for which *no* active learning algorithm of that type can outperform passive learning. Since all of the above label complexity analyses (splitting index, teaching dimension, disagreement coefficient) apply to certain respective self-verifying learning algorithms, these negative results are also reflected in all of the existing general label complexity analyses.

While at first these negative results may seem discouraging, Balcan, Hanneke, and Vaughan (2010) noted that if we do not require the algorithm to be self-verifying, instead simply measuring the number of label requests the algorithm needs to *find* a good classifier, rather than the number needed to both find a good classifier *and verify* that it is indeed good, then these negative results vanish. In fact, (shockingly) they were able to show that for any concept space with finite VC dimension, and any fixed data distribution, for any given passive learning algorithm there is an active learning algorithm with asymptotically superior label complexity for *every* nontrivial target concept! A positive result of this generality and strength is certainly an exciting advance in our understanding of the advantages of active learning. But perhaps equally exciting are the unresolved questions raised by that work, as there are potential opportunities to strengthen, generalize, simplify, and elaborate on this result. First, note that the above statement allows the active learning algorithm to be specialized to the particular distribution according to which the (unlabeled) data are sampled, and indeed the active learning method used by Balcan, Hanneke, and Vaughan (2010) in their proof has a rather strong direct dependence on the data distribution (which cannot be removed by simply replacing some calculations with data-dependent estimators). One interesting question is whether

an alternative approach might avoid this direct distribution-dependence in the algorithm, so that the claim can be strengthened to say that the active algorithm is superior to the passive algorithm for all nontrivial target concepts *and data distributions*. This question is interesting both theoretically, in order to obtain the strongest possible theorem on the advantages of active learning, as well as practically, since direct access to the distribution from which the data are sampled is typically *not* available in practical learning scenarios. A second question left open by Balcan, Hanneke, and Vaughan (2010) regards the *magnitude* of the gap between the active and passive label complexities. Specifically, although they did find particularly nasty learning problems where the label complexity of active learning will be close to that of passive learning (though always better), they hypothesized that for most natural learning problems, the improvements over passive learning should typically be *exponentially large* (as is the case for threshold classifiers); they gave many examples to illustrate this point, but left open the problem of characterizing general sufficient conditions for these exponential improvements to be achievable, even when they are not achievable by self-verifying algorithms. Another question left unresolved by Balcan, Hanneke, and Vaughan (2010) is whether this type of general improvement guarantee might be realized by a computationally *efficient* active learning algorithm. Finally, they left open the question of whether such general results might be further generalized to settings that involve noisy labels. The present work picks up where Balcan, Hanneke, and Vaughan (2010) left off in several respects, making progress on each of the above questions, in some cases completely resolving the question.

1.1.2 THE AGNOSTIC CASE

In addition to the above advances in our understanding of active learning in the realizable case, there has also been wonderful progress in making these methods robust to imperfect teachers, feature space underspecification, and model misspecification. This general topic goes by the name *agnostic active learning*, from its roots in the agnostic PAC model (Kearns, Schapire, and Sellie, 1994). In contrast to the realizable case, in the *agnostic case*, there is not necessarily a perfect classifier of a known form, and indeed there may even be *label noise* so that there is no perfect classifier of *any* form. Rather, we have a given set of classifiers (e.g., linear separators, or depth-limited decision trees, etc.), and the objective is to identify a classifier whose accuracy is not much worse than the best classifier of that type. Agnostic learning is strictly more general, and often more difficult, than realizable learning; this is true for both passive learning and active learning. However, for a given agnostic learning problem, we might still hope that active learning can achieve a given accuracy using fewer labels than required for passive learning.

The general topic of agnostic active learning got its first taste of real progress from Balcan, Beygelzimer, and Langford (2006a, 2009) with the publication of the A^2 (agnostic active) algorithm. This method is a noise-robust disagreement-based algorithm, which can be applied with essentially arbitrary types of classifiers under arbitrary noise distributions. It is interesting both for its effectiveness and (as with CAL) its elegance. The original work of Balcan, Beygelzimer, and Langford (2006a, 2009) showed that, in some special cases (thresholds, and homogeneous linear separators under a uniform distribution), the A^2 algorithm does achieve improved label complexities compared to the known results for passive learning.

Using a different type of general active learning strategy, Hanneke (2007a) found that the *teaching dimension* analysis (discussed above for the realizable case) can be extended beyond the realizable case, arriving at general bounds on the label complexity under arbitrary noise distributions.

These bounds improve over the known results for passive learning in many cases. However, the algorithm requires direct access to a certain quantity that depends on the noise distribution (namely, the noise rate, defined in Section 6 below), which would not be available in many real-world learning problems.

Later, Hanneke (2007b) established a general characterization of the label complexities achieved by A^2, expressed in terms of the disagreement coefficient. The result holds for arbitrary types of classifiers (of finite VC dimension) and arbitrary noise distributions, and represents the natural generalization of the aforementioned realizable-case analysis of CAL. In many cases, this result shows improvements over the known results for passive learning. Furthermore, because of the simplicity of the disagreement coefficient, the bound can be calculated for a variety of natural learning problems.

Soon after this, Dasgupta, Hsu, and Monteleoni (2007) proposed a new active learning strategy, which is also effective in the agnostic setting. Like A^2, the new algorithm is a noise-robust disagreement-based method. The work of Dasgupta, Hsu, and Monteleoni (2007) is significant for at least two reasons. First, they were able to establish a general label complexity bound for this method based on the disagreement coefficient. The bound is similar in form to the previous label complexity bound for A^2 by Hanneke (2007b), but improves the dependence of the bound on the disagreement coefficient. Second, the proposed method of Dasgupta, Hsu, and Monteleoni (2007) set a new standard for computational and aesthetic simplicity in agnostic active learning algorithms. This work has since been followed by related methods of Beygelzimer, Dasgupta, and Langford (2009) and Beygelzimer, Hsu, Langford, and Zhang (2010). In particular, Beygelzimer, Dasgupta, and Langford (2009) develop a method capable of learning under an essentially arbitrary loss function; they also show label complexity bounds similar to those of Dasgupta, Hsu, and Monteleoni (2007), but applicable to a larger class of loss functions, and stated in terms of a generalization of the disagreement coefficient for arbitrary loss functions.

While the above results are encouraging, the guarantees reflected in these label complexity bounds essentially take the form of (at best) constant factor improvements; specifically, in some cases the bounds improve the dependence on the noise rate factor (defined in Section 6 below), compared to the known results for passive learning. In fact, Kääriäinen (2006) showed that any label complexity bound depending on the noise distribution only via the noise rate cannot do better than this type of constant-factor improvement. This raised the question of whether, with a more detailed description of the noise distribution, one can show improvements in the *asymptotic form* of the label complexity compared to passive learning. Toward this end, Castro and Nowak (2008) studied a certain refined description of the noise conditions, related to the margin conditions of Mammen and Tsybakov (1999), which are well-studied in the passive learning literature. Specifically, they found that in some special cases, under certain restrictions on the noise distribution, the asymptotic form of the label complexity *can* be improved compared to passive learning, and in some cases the improvements can even be *exponential* in magnitude; to achieve this, they developed algorithms specifically tailored to the types of classifiers they studied (threshold classifiers and boundary fragment classes). Balcan, Broder, and Zhang (2007) later extended this result to general homogeneous linear separators under a uniform distribution. Following this, Hanneke (2009a, 2011) generalized these results, showing that both of the published general agnostic active learning algorithms (Balcan, Beygelzimer, and Langford, 2009; Dasgupta, Hsu, and Monteleoni, 2007) can also achieve these types of improvements in the asymptotic form of the label complexity; he further proved general bounds on the label complexities of these methods, again based on the disagreement coefficient, which apply to arbitrary types of classifiers, and which reflect these types of improvements

(under conditions on the disagreement coefficient). Wang (2009) later bounded the label complexity of A^2 under somewhat different noise conditions, in particular identifying weaker noise conditions sufficient for these improvements to be exponential in magnitude (again, under conditions on the disagreement coefficient). Koltchinskii (2010) has recently improved on some of Hanneke's results, refining certain logarithmic factors and simplifying the proofs, using a slightly different algorithm based on similar principles. Though the present work discusses only classes of finite VC dimension, most of the above references also contain results for various types of nonparametric classes with infinite VC dimension.

At present, all of the published bounds on the label complexity of agnostic active learning also apply to *self-verifying* algorithms. As mentioned, in the realizable case, it is typically possible to achieve significantly better label complexities if we do not require the active learning algorithm to be self-verifying, since the verification of learning may be more difficult than the learning itself (Balcan, Hanneke, and Vaughan, 2010). We might wonder whether this is also true in the agnostic case, and whether agnostic active learning algorithms that are not self-verifying might possibly achieve significantly better label complexities than the existing label complexity bounds described above. We investigate this in depth below.

1.2 Summary of Contributions

In the present work, we build on and extend the above results in a variety of ways, resolving a number of open problems. The main contributions of this work can be summarized as follows.

- We formally define a notion of a universal activizer, a meta-algorithm that transforms any passive learning algorithm into an active learning algorithm with asymptotically strictly superior label complexities for all nontrivial distributions and target concepts in the concept space.
- We analyze the existing strategy of disagreement-based active learning from this perspective, precisely characterizing the conditions under which this strategy can lead to a universal activizer for VC classes in the realizable case.
- We propose a new type of active learning algorithm, based on shatterable sets, and construct universal activizers for all VC classes in the realizable case based on this idea; in particular, this overcomes the issue of distribution-dependence in the existing results mentioned above.
- We present a novel generalization of the disagreement coefficient, along with a new asymptotic bound on the label complexities achievable by active learning in the realizable case; this new bound is often significantly smaller than the existing results in the published literature.
- We state new concise sufficient conditions for exponential improvements over passive learning to be achievable in the realizable case, including a significant weakening of known conditions in the published literature.
- We present a new general-purpose active learning algorithm for the agnostic case, based on the aforementioned idea involving shatterable sets.
- We prove a new asymptotic bound on the label complexities achievable by active learning in the presence of label noise (the agnostic case), often significantly smaller than any previously published results.

- We formulate a general conjecture on the theoretical advantages of active learning over passive learning in the presence of arbitrary types of label noise.

1.3 Outline of the Paper

The paper is organized as follows. In Section 2, we introduce the basic notation used throughout, formally define the learning protocol, and formally define the label complexity. We also define the notion of an *activizer*, which is a procedure that transforms a passive learning algorithm into an active learning algorithm with asymptotically superior label complexity. In Section 3, we review the established technique of *disagreement-based* active learning, and prove a new result precisely characterizing the scenarios in which disagreement-based active learning can be used to construct an activizer. In particular, we find that in many scenarios, disagreement-based active learning is not powerful enough to provide the desired improvements. In Section 4, we move beyond disagreement-based active learning, developing a new type of active learning algorithm based on *shatterable* sets of points. We apply this technique to construct a simple 3-stage procedure, which we then prove is a universal activizer for any concept space of finite VC dimension. In Section 5, we begin by reviewing the known results for bounding the label complexity of disagreement-based active learning in terms of the disagreement coefficient; we then develop a somewhat more involved procedure, again based on shatterable sets, which takes full advantage of the sequential nature of active learning. In addition to being an activizer, we show that this procedure often achieves dramatically superior label complexities than achievable by passive learning. In particular, we define a novel generalization of the disagreement coefficient, and use it to bound the label complexity of this procedure. This also provides us with concise sufficient conditions for obtaining exponential improvements over passive learning. Continuing in Section 6, we extend our framework to allow for label noise (the agnostic case), and discuss the possibility of extending the results from previous sections to these noisy learning problems. We first review the known results for noise-robust disagreement-based active learning, and characterizations of its label complexity in terms of the disagreement coefficient and Mammen-Tsybakov noise parameters. We then proceed to develop a new type of noise-robust active learning algorithm, again based on shatterable sets, and prove bounds on its label complexity in terms of our aforementioned generalization of the disagreement coefficient. Additionally, we present a general conjecture concerning the existence of activizers for certain passive learning algorithms in the agnostic case. We conclude in Section 7 with a host of enticing open problems for future investigation.

2. Definitions and Notation

For most of the paper, we consider the following formal setting. There is a measurable space $(\mathcal{X}, \mathcal{F}_{\mathcal{X}})$, where $\mathcal{X}$ is called the *instance space*; for simplicity, we suppose this is a standard Borel space (Srivastava, 1998) (e.g., $\mathbb{R}^m$ under the usual Borel σ-algebra), though most of the results generalize. A *classifier* is any measurable function $h : \mathcal{X} \to \{-1, +1\}$. There is a set $\mathbb{C}$ of classifiers called the *concept space*. In the *realizable case*, the learning problem is characterized as follows. There is a probability measure $\mathcal{P}$ on $\mathcal{X}$, and a sequence $\mathcal{Z}_X = \{X_1, X_2, \ldots\}$ of independent $\mathcal{X}$-valued random variables, each with distribution $\mathcal{P}$. We refer to these random variables as the sequence of *unlabeled examples*; although in practice, this sequence would typically be large but finite, to simplify the discussion and focus strictly on counting labels, we will suppose this sequence is inexhaustible. There is additionally a special element $f \in \mathbb{C}$, called the *target function*, and we denote by

$Y_i = f(X_i)$; we further denote by $\mathcal{Z} = \{(X_1,Y_1),(X_2,Y_2),\ldots\}$ the sequence of *labeled examples*, and for $m \in \mathbb{N}$ we denote by $\mathcal{Z}_m = \{(X_1,Y_1),(X_2,Y_2),\ldots,(X_m,Y_m)\}$ the finite subsequence consisting of the first m elements of $\mathcal{Z}$. For any classifier h, we define the *error rate* $\mathrm{er}(h) = \mathcal{P}(x : h(x) \neq f(x))$. Informally, the learning objective in the realizable case is to identify some h with small $\mathrm{er}(h)$ using elements from $\mathcal{Z}$, without direct access to f.

An *active learning algorithm* $\mathcal{A}$ is permitted direct access to the $\mathcal{Z}_X$ sequence (the unlabeled examples), but to gain access to the Y_i values it must request them one at a time, in a sequential manner. Specifically, given access to the $\mathcal{Z}_X$ values, the algorithm selects any index $i \in \mathbb{N}$, requests to observe the Y_i value, then having observed the value of Y_i, selects another index i', observes the value of $Y_{i'}$, etc. The algorithm is given as input an integer n, called the *label budget*, and is permitted to observe at most n labels total before eventually halting and returning a classifier $\hat{h}_n = \mathcal{A}(n)$; that is, by definition, an active learning algorithm never attempts to access more than the given budget n number of labels. We will then study the values of n sufficient to guarantee $\mathbb{E}[\mathrm{er}(\hat{h}_n)] \leq \varepsilon$, for any given value $\varepsilon \in (0,1)$. We refer to this as the *label complexity*. We will be particularly interested in the asymptotic dependence on ε in the label complexity, as $\varepsilon \to 0$. Formally, we have the following definition.

Definition 1 *An active learning algorithm $\mathcal{A}$ achieves label complexity $\Lambda(\cdot,\cdot,\cdot)$ if, for every target function f, distribution $\mathcal{P}$, $\varepsilon \in (0,1)$, and integer $n \geq \Lambda(\varepsilon, f, \mathcal{P})$, we have $\mathbb{E}[\mathrm{er}(\mathcal{A}(n))] \leq \varepsilon$.*

This definition of label complexity is similar to one originally studied by Balcan, Hanneke, and Vaughan (2010). It has a few features worth noting. First, the label complexity has an explicit dependence on the target function f and distribution $\mathcal{P}$. As noted by Dasgupta (2005), we need this dependence if we are to fully understand the range of label complexities achievable by active learning; we further illustrate this issue in the examples below. The second feature to note is that the label complexity, as defined here, is simply a sufficient budget size to achieve the specified accuracy. That is, here we are asking only how many label requests are required for the algorithm to achieve a given accuracy (in expectation). However, as noted by Balcan, Hanneke, and Vaughan (2010), this number might not be sufficiently large to *detect* that the algorithm has indeed achieved the required accuracy based only on the observed data. That is, because the number of labeled examples used in active learning can be quite small, we come across the problem that the number of labels needed to *learn* a concept might be significantly smaller than the number of labels needed to *verify* that we have successfully learned the concept. As such, this notion of label complexity is most useful in the *design* of effective learning algorithms, rather than for predicting the number of labels an algorithm should request in any particular application. Specifically, to design effective active learning algorithms, we should generally desire small label complexity values, so that (in the extreme case) if some algorithm $\mathcal{A}$ has smaller label complexity values than some other algorithm $\mathcal{A}'$ for *all* target functions and distributions, then (all other factors being equal) we should clearly prefer algorithm $\mathcal{A}$ over algorithm $\mathcal{A}'$; this is true regardless of whether we have a means to *detect* (verify) how large the improvements offered by algorithm $\mathcal{A}$ over algorithm $\mathcal{A}'$ are for any particular application. Thus, in our present context, performance guarantees in terms of this notion of label complexity play a role analogous to concepts such as *universal consistency* or *admissibility*, which are also generally useful in guiding the design of effective algorithms, but are not intended to be informative in the context of any particular application. See the work of Balcan, Hanneke, and Vaughan (2010) for a discussion of this issue, as it relates to a definition of label complexity similar

to that above, as well as other notions of label complexity from the active learning literature (some of which include a verification requirement).

We will be interested in the performance of active learning algorithms, relative to the performance of a given *passive learning algorithm*. In this context, a passive learning algorithm $\mathcal{A}$ takes as input a finite sequence of labeled examples $\mathcal{L} \in \bigcup_n (\mathcal{X} \times \{-1,+1\})^n$, and returns a classifier $\hat{h} = \mathcal{A}(\mathcal{L})$. We allow both active and passive learning algorithms to be randomized: that is, to have independent internal randomness, in addition to the given random data. We define the label complexity for a passive learning algorithm as follows.

Definition 2 *A passive learning algorithm $\mathcal{A}$ achieves label complexity $\Lambda(\cdot,\cdot,\cdot)$ if, for every target function f, distribution $\mathcal{P}$, $\varepsilon \in (0,1)$, and integer $n \geq \Lambda(\varepsilon, f, \mathcal{P})$, we have $\mathbb{E}[\mathrm{er}(\mathcal{A}(\mathcal{Z}_n))] \leq \varepsilon$.*

Although technically some algorithms may be able to achieve a desired accuracy without any observations, to make the general results easier to state (namely, those in Section 5), unless otherwise stated we suppose label complexities (both passive and active) take strictly positive values, among $\mathbb{N} \cup \{\infty\}$; note that label complexities (both passive and active) can be infinite, indicating that the corresponding algorithm might not achieve expected error rate ε for *any* $n \in \mathbb{N}$. Both the passive and active label complexities are defined as a number of labels sufficient to guarantee the *expected* error rate is at most ε. It is also common in the literature to discuss the number of label requests sufficient to guarantee the error rate is at most ε with *high probability* $1-\delta$ (e.g., Balcan, Hanneke, and Vaughan, 2010). In the present work, we formulate our results in terms of the expected error rate because it simplifies the discussion of asymptotics, in that we need only study the behavior of the label complexity as the single argument ε approaches 0, rather than the more complicated behavior of a function of ε and δ as both ε and δ approach 0 at various relative rates. However, we note that analogous results for these high-probability guarantees on the error rate can be extracted from the proofs below without much difficulty, and in several places we explicitly state results of this form.

Below we employ the standard notation from asymptotic analysis, including $O(\cdot)$, $o(\cdot)$, $\Omega(\cdot)$, $\omega(\cdot)$, $\Theta(\cdot)$, $\ll$, and $\gg$. In all contexts below not otherwise specified, the asymptotics are always considered as $\varepsilon \to 0$ when considering a function of ε, and as $n \to \infty$ when considering a function of n; also, in any expression of the form "$x \to 0$," we always mean the limit *from above* (i.e., $x \downarrow 0$). For instance, when considering nonnegative functions of ε, $\lambda_a(\varepsilon)$ and $\lambda_p(\varepsilon)$, the above notations are defined as follows. We say $\lambda_a(\varepsilon) = o(\lambda_p(\varepsilon))$ when $\lim_{\varepsilon\to 0} \frac{\lambda_a(\varepsilon)}{\lambda_p(\varepsilon)} = 0$, and this is equivalent to writing $\lambda_p(\varepsilon) = \omega(\lambda_a(\varepsilon))$, $\lambda_a(\varepsilon) \ll \lambda_p(\varepsilon)$, or $\lambda_p(\varepsilon) \gg \lambda_a(\varepsilon)$. We say $\lambda_a(\varepsilon) = O(\lambda_p(\varepsilon))$ when $\limsup_{\varepsilon\to 0} \frac{\lambda_a(\varepsilon)}{\lambda_p(\varepsilon)} < \infty$, which can equivalently be expressed as $\lambda_p(\varepsilon) = \Omega(\lambda_a(\varepsilon))$. Finally, we write $\lambda_a(\varepsilon) = \Theta(\lambda_p(\varepsilon))$ to mean that both $\lambda_a(\varepsilon) = O(\lambda_p(\varepsilon))$ and $\lambda_a(\varepsilon) = \Omega(\lambda_p(\varepsilon))$ are satisfied. We also use the standard notation for the limit of a sequence of sets, such as $\lim_{r\to 0} A_r$, defined by the property $\mathbb{1}_{\lim_{r\to 0} A_r} = \lim_{r\to 0} \mathbb{1}_{A_r}$ (if the latter exists), where $\mathbb{1}_A$ is the indicator function for the set A.

Define the class of functions Polylog$(1/\varepsilon)$ as those $g : (0,1) \to [0,\infty)$ such that, for some $k \in [0,\infty)$, $g(\varepsilon) = O(\log^k(1/\varepsilon))$. For a label complexity Λ, also define the set Nontrivial(Λ) as the collection of all pairs $(f,\mathcal{P})$ of a classifier and a distribution such that, $\forall \varepsilon > 0, \Lambda(\varepsilon, f, \mathcal{P}) < \infty$, and $\forall g \in \mathrm{Polylog}(1/\varepsilon), \Lambda(\varepsilon, f, \mathcal{P}) = \omega(g(\varepsilon))$.

In this context, an *active meta-algorithm* is a procedure $\mathcal{A}_a$ taking as input a passive algorithm $\mathcal{A}_p$ and a label budget n, such that for any passive algorithm $\mathcal{A}_p$, $\mathcal{A}_a(\mathcal{A}_p,\cdot)$ is an active learning algorithm. We define an *activizer* for a given passive algorithm as follows.

Definition 3 *We say an active meta-algorithm $\mathcal{A}_a$* activizes *a passive algorithm $\mathcal{A}_p$ for a concept space $\mathbb{C}$ if the following holds. For any label complexity Λ_p achieved by $\mathcal{A}_p$, the active learning algorithm $\mathcal{A}_a(\mathcal{A}_p,\cdot)$ achieves a label complexity Λ_a such that, for every $f\in\mathbb{C}$ and every distribution $\mathcal{P}$ on $\mathcal{X}$ with $(f,\mathcal{P})\in$ Nontrivial(Λ_p), there exists a constant $c\in[1,\infty)$ such that*

$$\Lambda_a(c\varepsilon,f,\mathcal{P}) = o\left(\Lambda_p(\varepsilon,f,\mathcal{P})\right).$$

In this case, $\mathcal{A}_a$ is called an activizer *for $\mathcal{A}_p$ with respect to $\mathbb{C}$, and the active learning algorithm $\mathcal{A}_a(\mathcal{A}_p,\cdot)$ is called the $\mathcal{A}_a$-*activized *$\mathcal{A}_p$.*

We also refer to any active meta-algorithm $\mathcal{A}_a$ that activizes *every* passive algorithm $\mathcal{A}_p$ for $\mathbb{C}$ as a *universal activizer* for $\mathbb{C}$. One of the main contributions of this work is establishing that such universal activizers do exist for any VC class $\mathbb{C}$.

A bit of explanation is in order regarding Definition 3. We might interpret it as follows: an *activizer* for $\mathcal{A}_p$ strongly improves (in a little-o sense) the label complexity for all *nontrivial* target functions and distributions. Here, we seek a meta-algorithm that, when given $\mathcal{A}_p$ as input, results in an active learning algorithm with strictly superior label complexities. However, there is a sense in which some distributions $\mathcal{P}$ or target functions f are *trivial* relative to $\mathcal{A}_p$. For instance, perhaps $\mathcal{A}_p$ has a *default* classifier that it is naturally biased toward (e.g., with minimal $\mathcal{P}(x:h(x)=+1)$, as in the Closure algorithm of Helmbold, Sloan, and Warmuth, 1990), so that when this default classifier is the target function, $\mathcal{A}_p$ achieves a constant label complexity. In these trivial scenarios, we cannot hope to *improve* over the behavior of the passive algorithm, but instead can only hope to *compete* with it. The *sense* in which we wish to compete may be a subject of some controversy, but the implication of Definition 3 is that the label complexity of the activized algorithm should be strictly better than every nontrivial upper bound on the label complexity of the passive algorithm. For instance, if $\Lambda_p(\varepsilon,f,\mathcal{P})\in$ Polylog$(1/\varepsilon)$, then we are guaranteed $\Lambda_a(\varepsilon,f,\mathcal{P})\in$ Polylog$(1/\varepsilon)$ as well, but if $\Lambda_p(\varepsilon,f,\mathcal{P})=O(1)$, we are still only guaranteed $\Lambda_a(\varepsilon,f,\mathcal{P})\in$ Polylog$(1/\varepsilon)$. This serves the purpose of defining a framework that can be studied without requiring too much obsession over small additive terms in trivial scenarios, thus focusing the analyst's efforts toward nontrivial scenarios where $\mathcal{A}_p$ has relatively *large* label complexity, which are precisely the scenarios for which active learning is truly needed. In our proofs, we find that in fact Polylog$(1/\varepsilon)$ can be replaced with $O(\log(1/\varepsilon))$, giving a slightly broader definition of "nontrivial," for which all of the results below still hold. Section 7 discusses open problems regarding this issue of trivial problems.

The definition of Nontrivial$(\cdot)$ also only requires the activized algorithm to be effective in scenarios where the passive learning algorithm has *reasonable* behavior (i.e., finite label complexities); this is only intended to keep with the reduction-based style of the framework, and in fact this restriction can easily be lifted using a trick from Balcan, Hanneke, and Vaughan (2010) (aggregating the activized algorithm with another algorithm that is always reasonable).

Finally, we also allow a constant factor c loss in the ε argument to Λ_a. We allow this to be an arbitrary constant, again in the interest of allowing the analyst to focus only on the most significant aspects of the problem; for most reasonable passive learning algorithms, we typically expect $\Lambda_p(\varepsilon,f,\mathcal{P})=$ Poly$(1/\varepsilon)$, in which case c can be set to 1 by adjusting the leading constant factors of

Λ_a. A careful inspection of our proofs reveals that c can always be set arbitrarily close to 1 without affecting the theorems below (and in fact, we can even get $c = (1 + o(1))$, a function of ε).

Throughout this work, we will adopt the usual notation for probabilities, such as $\mathbb{P}(\text{er}(\hat{h}) > \varepsilon)$, and as usual we interpret this as measuring the corresponding event in the (implicit) underlying probability space. In particular, we make the usual implicit assumption that all sets involved in the analysis are measurable; where this assumption does not hold, we may turn to outer probabilities, though we will not make further mention of these technical details. We will also use the notation $P^k(\cdot)$ to represent k-dimensional product measures; for instance, for a measurable set $A \subseteq \mathcal{X}^k$, $\mathcal{P}^k(A) = \mathbb{P}((X'_1, \ldots, X'_k) \in A)$, for independent $\mathcal{P}$-distributed random variables $X'_1, \ldots, X'_k$. Additionally, to simplify notation, we will adopt the convention that $\mathcal{X}^0 = \{\varnothing\}$ and $\mathcal{P}^0(\mathcal{X}^0) = 1$. Throughout, we will denote by $\mathbb{1}_A(z)$ the indicator function for a set A, which has the value 1 when $z \in A$ and 0 otherwise; additionally, at times it will be more convenient to use the bipolar indicator function, defined as $\mathbb{1}^{\pm}_A(z) = 2\mathbb{1}_A(z) - 1$.

We will require a few additional definitions for the discussion below. For any classifier $h : \mathcal{X} \to \{-1, +1\}$ and finite sequence of labeled examples $\mathcal{L} \in \bigcup_m (\mathcal{X} \times \{-1, +1\})^m$, define the *empirical error rate* $\text{er}_{\mathcal{L}}(h) = |\mathcal{L}|^{-1} \sum_{(x,y)\in\mathcal{L}} \mathbb{1}_{\{-y\}}(h(x))$; for completeness, define $\text{er}_{\emptyset}(h) = 0$. Also, for $\mathcal{L} = \mathcal{Z}_m$, the first m labeled examples in the data sequence, abbreviate this as $\text{er}_m(h) = \text{er}_{\mathcal{Z}_m}(h)$. For any probability measure P on $\mathcal{X}$, set of classifiers $\mathcal{H}$, classifier h, and $r > 0$, define $\text{B}_{\mathcal{H},P}(h, r) = \{g \in \mathcal{H} : P(x : h(x) \neq g(x)) \leq r\}$; when $P = \mathcal{P}$, the distribution of the unlabeled examples, and $\mathcal{P}$ is clear from the context, we abbreviate this as $\text{B}_{\mathcal{H}}(h, r) = \text{B}_{\mathcal{H},\mathcal{P}}(h, r)$; furthermore, when $P = \mathcal{P}$ and $\mathcal{H} = \mathbb{C}$, the concept space, and both $\mathcal{P}$ and $\mathbb{C}$ are clear from the context, we abbreviate this as $\text{B}(h, r) = \text{B}_{\mathbb{C},\mathcal{P}}(h, r)$. Also, for any set of classifiers $\mathcal{H}$, and any sequence of labeled examples $\mathcal{L} \in \bigcup_m (\mathcal{X} \times \{-1, +1\})^m$, define $\mathcal{H}[\mathcal{L}] = \{h \in \mathcal{H} : \text{er}_{\mathcal{L}}(h) = 0\}$; for any $(x, y) \in \mathcal{X} \times \{-1, +1\}$, abbreviate $\mathcal{H}[(x, y)] = \mathcal{H}[\{(x, y)\}] = \{h \in \mathcal{H} : h(x) = y\}$.

We also adopt the usual definition of "shattering" used in learning theory (e.g., Vapnik, 1998). Specifically, for any set of classifiers $\mathcal{H}$, $k \in \mathbb{N}$, and $S = (x_1, \ldots, x_k) \in \mathcal{X}^k$, we say $\mathcal{H}$ *shatters* S if, $\forall (y_1, \ldots, y_k) \in \{-1, +1\}^k$, $\exists h \in \mathcal{H}$ such that $\forall i \in \{1, \ldots, k\}$, $h(x_i) = y_i$; equivalently, $\mathcal{H}$ shatters S if $\exists \{h_1, \ldots, h_{2^k}\} \subseteq \mathcal{H}$ such that for each $i, j \in \{1, \ldots, 2^k\}$ with $i \neq j$, $\exists \ell \in \{1, \ldots, k\}$ with $h_i(x_\ell) \neq h_j(x_\ell)$. To simplify notation, we will also say that $\mathcal{H}$ shatters $\varnothing$ if and only if $\mathcal{H} \neq \{\}$. As usual, we define the *VC dimension* of $\mathbb{C}$, denoted d, as the largest integer k such that $\exists S \in \mathcal{X}^k$ shattered by $\mathbb{C}$ (Vapnik and Chervonenkis, 1971; Vapnik, 1998). To focus on nontrivial problems, we will only consider concept spaces $\mathbb{C}$ with $d > 0$ in the results below. Generally, any such concept space $\mathbb{C}$ with $d < \infty$ is called a *VC class*.

2.1 Motivating Examples

Throughout this paper, we will repeatedly refer to a few canonical examples. Although themselves quite toy-like, they represent the boiled-down essence of some important distinctions between various types of learning problems. In some sense, the process of grappling with the fundamental distinctions raised by these types of examples has been a driving force behind much of the recent progress in understanding the label complexity of active learning.

The first example is perhaps the most classic, and is clearly the first that comes to mind when considering the potential for active learning to provide strong improvements over passive learning.

Example 1 *In the problem of learning* threshold *classifiers, we consider* $\mathcal{X} = [0, 1]$ *and* $\mathbb{C} = \{h_z(x) = \mathbb{1}^{\pm}_{[z,1]}(x) : z \in (0, 1)\}$.

There is a simple universal activizer for threshold classifiers, based on a kind of binary search. Specifically, suppose $n \in \mathbb{N}$ and that $\mathcal{A}_p$ is any given passive learning algorithm. Consider the points in $\{X_1, X_2, \ldots, X_m\}$, for $m = 2^{n-1}$, and sort them in increasing order: $X_{(1)}, X_{(2)}, \ldots, X_{(m)}$. Also initialize $\ell = 0$ and $u = m+1$, and define $X_{(0)} = 0$ and $X_{(m+1)} = 1$. Now request the label of $X_{(i)}$ for $i = \lfloor (\ell + u)/2 \rfloor$ (i.e., the median point between ℓ and u); if the label is -1, let $\ell = i$, and otherwise let $u = i$; repeat this (requesting this median point, then updating ℓ or u accordingly) until we have $u = \ell + 1$. Finally, let $\hat{z} = X_{(u)}$, construct the labeled sequence $\mathcal{L} = \{(X_1, h_{\hat{z}}(X_1)), \ldots, (X_m, h_{\hat{z}}(X_m))\}$, and return the classifier $\hat{h} = \mathcal{A}_p(\mathcal{L})$.

Since each label request at least halves the set of integers between ℓ and u, the total number of label requests is at most $\log_2(m) + 1 = n$. Supposing $f \in \mathbb{C}$ is the target function, this procedure maintains the invariant that $f(X_{(\ell)}) = -1$ and $f(X_{(u)}) = +1$. Thus, once we reach $u = \ell + 1$, since f is a threshold, it must be some h_z with $z \in (\ell, u]$; therefore every $X_{(j)}$ with $j \leq \ell$ has $f(X_{(j)}) = -1$, and likewise every $X_{(j)}$ with $j \geq u$ has $f(X_{(j)}) = +1$; in particular, this means $\mathcal{L}$ equals $\mathcal{Z}_m$, the *true* labeled sequence. But this means $\hat{h} = \mathcal{A}_p(\mathcal{Z}_m)$. Since $n = \log_2(m) + 1$, this active learning algorithm will achieve an equivalent error rate to what $\mathcal{A}_p$ achieves with m labeled examples, but using only $\log_2(m) + 1$ label requests. In particular, this implies that if $\mathcal{A}_p$ achieves label complexity Λ_p, then this active learning algorithm achieves label complexity Λ_a such that $\Lambda_a(\varepsilon, f, \mathcal{P}) \leq \log_2 \Lambda_p(\varepsilon, f, \mathcal{P}) + 2$; as long as $1 \ll \Lambda_p(\varepsilon, f, \mathcal{P}) < \infty$, this is $o(\Lambda_p(\varepsilon, f, \mathcal{P}))$, so that this procedure activizes $\mathcal{A}_p$ for $\mathbb{C}$.

The second example we consider is almost equally simple (only increasing the VC dimension from 1 to 2), but is far more subtle in terms of how we must approach its analysis in active learning.

Example 2 *In the problem of learning* interval *classifiers, we consider* $\mathcal{X} = [0,1]$ *and* $\mathbb{C} = \{h_{[a,b]}(x) = \mathbb{1}^{\pm}_{[a,b]}(x) : 0 < a \leq b < 1\}$.

For the intervals problem, we can also construct a universal activizer, though slightly more complicated. Specifically, suppose again that $n \in \mathbb{N}$ and that $\mathcal{A}_p$ is any given passive learning algorithm. We first request the labels $\{Y_1, Y_2, \ldots, Y_{\lceil n/2 \rceil}\}$ of the first $\lceil n/2 \rceil$ examples in the sequence. If every one of these labels is -1, then we immediately return the all-negative constant classifier $\hat{h}(x) = -1$. Otherwise, consider the points $\{X_1, X_2, \ldots, X_m\}$, for $m = \max\left\{2^{\lfloor n/4 \rfloor - 1}, n\right\}$, and sort them in increasing order $X_{(1)}, X_{(2)}, \ldots, X_{(m)}$. For some value $i \in \{1, \ldots, \lceil n/2 \rceil\}$ with $Y_i = +1$, let j_+ denote the corresponding index j such that $X_{(j)} = X_i$. Also initialize $\ell_1 = 0$, $u_1 = \ell_2 = j_+$, and $u_2 = m + 1$, and define $X_{(0)} = 0$ and $X_{(m+1)} = 1$. Now if $\ell_1 + 1 < u_1$, request the label of $X_{(i)}$ for $i = \lfloor (\ell_1 + u_1)/2 \rfloor$ (the median point between ℓ_1 and u_1); if the label is -1, let $\ell_1 = i$, and otherwise let $u_1 = i$; repeat this (requesting this median point, then updating ℓ_1 or u_1 accordingly) until we have $u_1 = \ell_1 + 1$. Now if $\ell_2 + 1 < u_2$, request the label of $X_{(i)}$ for $i = \lfloor (\ell_2 + u_2)/2 \rfloor$ (the median point between ℓ_2 and u_2); if the label is -1, let $u_2 = i$, and otherwise let $\ell_2 = i$; repeat this (requesting this median point, then updating u_2 or ℓ_2 accordingly) until we have $u_2 = \ell_2 + 1$. Finally, let $\hat{a} = u_1$ and $\hat{b} = \ell_2$, construct the labeled sequence $\mathcal{L} = \left\{\left(X_1, h_{[\hat{a},\hat{b}]}(X_1)\right), \ldots, \left(X_m, h_{[\hat{a},\hat{b}]}(X_m)\right)\right\}$, and return the classifier $\hat{h} = \mathcal{A}_p(\mathcal{L})$.

Since each label request in the second phase halves the set of values between either ℓ_1 and u_1 or ℓ_2 and u_2, the total number of label requests is at most $\min\{m, \lceil n/2 \rceil + 2\log_2(m) + 2\} \leq n$. Suppose $f \in \mathbb{C}$ is the target function, and let $w(f) = \mathcal{P}(x : f(x) = +1)$. If $w(f) = 0$, then with probability 1 the algorithm will return the constant classifier $\hat{h}(x) = -1$, which has $\mathrm{er}(\hat{h}) = 0$ in this case. Otherwise, if $w(f) > 0$, then for any $n \geq \frac{2}{w(f)} \ln \frac{1}{\varepsilon}$, with probability at least $1 - \varepsilon$, there exists

$i \in \{1,\ldots,\lceil n/2 \rceil\}$ with $Y_i = +1$. Let H_+ denote the event that such an i exists. Supposing this is the case, the algorithm will make it into the second phase. In this case, the procedure maintains the invariant that $f(X_{(\ell_1)}) = -1$, $f(X_{(u_1)}) = f(X_{(\ell_2)}) = +1$, and $f(X_{(u_2)}) = -1$, where $\ell_1 < u_1 \leq \ell_2 < u_2$. Thus, once we have $u_1 = \ell_1 + 1$ and $u_2 = \ell_2 + 1$, since f is an interval, it must be some $h_{[a,b]}$ with $a \in (\ell_1, u_1]$ and $b \in [\ell_2, u_1)$; therefore, every $X_{(j)}$ with $j \leq \ell_1$ or $j \geq u_2$ has $f(X_{(j)}) = -1$, and likewise every $X_{(j)}$ with $u_1 \leq j \leq \ell_2$ has $f(X_{(j)}) = +1$; in particular, this means $\mathcal{L}$ equals $\mathcal{Z}_m$, the *true* labeled sequence. But this means $\hat{h} = \mathcal{A}_p(\mathcal{Z}_m)$. Supposing $\mathcal{A}_p$ achieves label complexity Λ_p, and that $n \geq \max\left\{8 + 4\log_2 \Lambda_p(\varepsilon, f, \mathcal{P}), \frac{2}{w(f)} \ln \frac{1}{\varepsilon}\right\}$, then $m \geq 2^{\lfloor n/4 \rfloor - 1} \geq \Lambda_p(\varepsilon, f, \mathcal{P})$ and $\mathbb{E}\left[\mathrm{er}(\hat{h})\right] \leq \mathbb{E}\left[\mathrm{er}(\hat{h})\mathbb{1}_{H_+}\right] + (1 - \mathbb{P}(H_+)) \leq \mathbb{E}\left[\mathrm{er}(\mathcal{A}_p(\mathcal{Z}_m))\right] + \varepsilon \leq 2\varepsilon$. In particular, this means this active learning algorithm achieves label complexity Λ_a such that, for any $f \in \mathbb{C}$ with $w(f) = 0$, $\Lambda_a(2\varepsilon, f, \mathcal{P}) = 0$, and for any $f \in \mathbb{C}$ with $w(f) > 0$, $\Lambda_a(2\varepsilon, f, \mathcal{P}) \leq \max\left\{8 + 4\log_2 \Lambda_p(\varepsilon, f, \mathcal{P}), \frac{2}{w(f)} \ln \frac{1}{\varepsilon}\right\}$. If $(f, \mathcal{P}) \in \text{Nontrivial}(\Lambda_p)$, then $\frac{2}{w(f)} \ln \frac{1}{\varepsilon} = o(\Lambda_p(\varepsilon, f, \mathcal{P}))$ and $8 + 4\log_2 \Lambda_p(\varepsilon, f, \mathcal{P}) = o(\Lambda_p(\varepsilon, f, \mathcal{P}))$, so that $\Lambda_a(2\varepsilon, f, \mathcal{P}) = o(\Lambda_p(\varepsilon, f, \mathcal{P}))$. Therefore, this procedure activizes $\mathcal{A}_p$ for $\mathbb{C}$.

This example also brings to light some interesting phenomena in the analysis of the label complexity of active learning. Note that unlike the thresholds example, we have a much stronger dependence on the target function in these label complexity bounds, via the $w(f)$ quantity. This issue is fundamental to the problem, and cannot be avoided. In particular, when $\mathcal{P}([0,x])$ is continuous, this is the very issue that makes the *minimax* label complexity for this problem (i.e., $\min_{\Lambda_a} \max_{f \in \mathbb{C}} \Lambda_a(\varepsilon, f, \mathcal{P})$) *no better* than passive learning (Dasgupta, 2005). Thus, this problem emphasizes the need for any informative label complexity analysis of active learning to explicitly describe the dependence of the label complexity on the target function, as advocated by Dasgupta (2005). This example also highlights the *unverifiability* phenomenon explored by Balcan, Hanneke, and Vaughan (2010), since in the case of $w(f) = 0$, the error rate of the returned classifier is *zero*, but (for nondegenerate $\mathcal{P}$) there is no way for the algorithm to verify this fact based only on the finite number of labels it observes. In fact, Balcan, Hanneke, and Vaughan (2010) have shown that under continuous $\mathcal{P}$, for any $f \in \mathbb{C}$ with $w(f) = 0$, the number of labels required to both *find* a classifier of small error rate *and verify* that the error rate is small based only on observable quantities is essentially *no better* than for passive learning.

These issues are present to a small degree in the intervals example, but were easily handled in a very natural way. The target-dependence shows up only in an initial phase of waiting for a positive example, and the always-negative classifiers were handled by setting a *default* return value. However, we can amplify these issues so that they show up in more subtle and involved ways. Specifically, consider the following example, studied by Balcan, Hanneke, and Vaughan (2010).

Example 3 *In the problem of learning* unions of i intervals, *we consider* $\mathcal{X} = [0,1]$ *and* $\mathbb{C} = \left\{h_{\mathbf{z}}(x) = \mathbb{1}^{\pm}_{\bigcup_{j=1}^{i}[z_{2j-1}, z_{2j}]}(x) : 0 < z_1 \leq z_2 \leq \ldots \leq z_{2i} < 1\right\}$.

The challenge of this problem is that, because sometimes $z_j = z_{j+1}$ for some j values, we do not know how many intervals are required to minimally represent the target function: only that it is at most i. This issue will be made clearer below. We can essentially think of any effective strategy here as having two components: one component that searches (perhaps randomly) with the purpose of identifying at least one example from each decision region, and another component that refines our estimates of the end-points of the regions the first component identifies. Later, we will go through the behavior of a universal activizer for this problem in detail.

3. Disagreement-Based Active Learning

At present, perhaps the best-understood active learning algorithms are those choosing their label requests based on disagreement among a set of remaining candidate classifiers. The canonical algorithm of this type, a version of which we discuss below in Section 5.1, was proposed by Cohn, Atlas, and Ladner (1994). Specifically, for any set $\mathcal{H}$ of classifiers, define the *region of disagreement*:

$$\mathrm{DIS}(\mathcal{H}) = \{x \in \mathcal{X} : \exists h_1, h_2 \in \mathcal{H} \text{ s.t. } h_1(x) \neq h_2(x)\}.$$

The basic idea of disagreement-based algorithms is that, at any given time in the algorithm, there is a subset $V \subseteq \mathbb{C}$ of remaining candidates, called the *version space*, which is guaranteed to contain the target f. When deciding whether to request a particular label Y_i, the algorithm simply checks whether $X_i \in \mathrm{DIS}(V)$: if so, the algorithm requests Y_i, and otherwise it does not. This general strategy is reasonable, since for any $X_i \notin \mathrm{DIS}(V)$, the label agreed upon by V must be $f(X_i)$, so that we would get no information by requesting Y_i; that is, for $X_i \notin \mathrm{DIS}(V)$, we can accurately *infer* Y_i based on information already available. This type of algorithm has recently received substantial attention, not only for its obvious elegance and simplicity, but also because (as we discuss in Section 6) there are natural ways to extend the technique to the general problem of learning with label noise and model misspecification (the *agnostic* setting). The details of disagreement-based algorithms can vary in how they update the set V and how frequently they do so, but it turns out almost all disagreement-based algorithms share many of the same fundamental properties, which we describe below.

3.1 A Basic Disagreement-Based Active Learning Algorithm

In Section 5.1, we discuss several known results on the label complexities achievable by these types of active learning algorithms. However, for now let us examine a very basic algorithm of this type. The following is intended to be a simple representative of the family of disagreement-based active learning algorithms. It has been stripped down to the bare essentials of what makes such algorithms work. As a result, although the gap between its label complexity and that achieved by passive learning is not necessarily as large as those achieved by the more sophisticated disagreement-based active learning algorithms of Section 5.1, it has the property that whenever those more sophisticated methods have label complexities asymptotically superior to those achieved by passive learning, that guarantee will also be true for this simpler method, and vice versa. The algorithm operates in only 2 phases. In the first, it uses one batch of label requests to reduce the version space V to a subset of $\mathbb{C}$; in the second, it uses another batch of label requests, this time only requesting labels for points in $\mathrm{DIS}(V)$. Thus, we have isolated precisely that aspect of disagreement-based active learning that involves improvements due to only requesting the labels of examples in the region of disagreement. The procedure is formally defined as follows, in terms of an estimator $\hat{P}_n(\mathrm{DIS}(V))$ specified below.

Meta-Algorithm 0
Input: passive algorithm $\mathcal{A}_p$, label budget n
Output: classifier $\hat{h}$

0. Request the first $\lfloor n/2 \rfloor$ labels $\{Y_1, \ldots, Y_{\lfloor n/2 \rfloor}\}$, and let $t \leftarrow \lfloor n/2 \rfloor$
1. Let $V = \{h \in \mathbb{C} : \text{er}_{\lfloor n/2 \rfloor}(h) = 0\}$
2. Let $\hat{\Delta} \leftarrow \hat{P}_n(\text{DIS}(V))$
3. Let $\mathcal{L} \leftarrow \{\}$
4. For $m = \lfloor n/2 \rfloor + 1, \ldots \lfloor n/2 \rfloor + \lfloor n/(4\hat{\Delta}) \rfloor$
5. If $X_m \in \text{DIS}(V)$ and $t < n$, request the label Y_m of X_m, and let $\hat{y} \leftarrow Y_m$ and $t \leftarrow t+1$
6. Else let $\hat{y} \leftarrow h(X_m)$ for an arbitrary $h \in V$
7. Let $\mathcal{L} \leftarrow \mathcal{L} \cup \{(X_m, \hat{y})\}$
8. Return $\mathcal{A}_p(\mathcal{L})$

Meta-Algorithm 0 depends on a data-dependent estimator $\hat{P}_n(\text{DIS}(V))$ of $\mathcal{P}(\text{DIS}(V))$, which we can define in a variety of ways using only *unlabeled* examples. In particular, for the theorems below, we will take the following definition for $\hat{P}_n(\text{DIS}(V))$, designed to be a confidence upper bound on $\mathcal{P}(\text{DIS}(V))$. Let $\mathcal{U}_n = \{X_{n^2+1}, \ldots, X_{2n^2}\}$. Then define

$$\hat{P}_n(\text{DIS}(V)) = \max\left\{ \frac{2}{n^2} \sum_{x \in \mathcal{U}_n} \mathbb{1}_{\text{DIS}(V)}(x), \frac{4}{n} \right\}. \tag{1}$$

Meta-Algorithm 0 is divided into two stages: one stage where we focus on reducing V, and a second stage where we construct the sample $\mathcal{L}$ for the passive algorithm. This might intuitively seem somewhat wasteful, as one might wish to use the requested labels from the first stage to augment those in the second stage when constructing $\mathcal{L}$, thus feeding all of the observed labels into the passive algorithm $\mathcal{A}_p$. Indeed, this can improve the label complexity in some cases (albeit only by a constant factor); however, in order to get the *general* property of being an activizer for *all* passive algorithms $\mathcal{A}_p$, we construct the sample $\mathcal{L}$ so that the conditional distribution of the $\mathcal{X}$ components in $\mathcal{L}$ given $|\mathcal{L}|$ is $\mathcal{P}^{|\mathcal{L}|}$, so that it is (conditionally) an i.i.d. sample, which is essential to our analysis. The choice of the number of (unlabeled) examples to process in the second stage guarantees (by a Chernoff bound) that the "$t < n$" constraint in Step 5 is redundant; this is a trick we will employ in several of the methods below. As explained above, because $f \in V$, this implies that every $(x,y) \in \mathcal{L}$ has $y = f(x)$.

To give some basic intuition for how this algorithm behaves, consider the example of learning threshold classifiers (Example 1); to simplify the explanation, for now we ignore the fact that $\hat{P}_n$ is only an estimate, as well as the "$t < n$" constraint in Step 5 (both of which will be addressed in the general analysis below). In this case, suppose the target function is $f = h_z$. Let $a = \max\{X_i : X_i < z, 1 \le i \le \lfloor n/2 \rfloor\}$ and $b = \min\{X_i : X_i \ge z, 1 \le i \le \lfloor n/2 \rfloor\}$. Then $V = \{h_{z'} : a < z' \le b\}$ and $\text{DIS}(V) = (a,b)$, so that the second phase of the algorithm only requests labels for a number of points in the region (a,b). With probability $1-\varepsilon$, the probability mass in this region is at most $O(\log(1/\varepsilon)/n)$, so that $|\mathcal{L}| \ge \ell_{n,\varepsilon} = \Omega(n^2/\log(1/\varepsilon))$; also, since the labels in $\mathcal{L}$ are all correct, and the X_m values in $\mathcal{L}$ are conditionally iid (with distribution $\mathcal{P}$) given $|\mathcal{L}|$, we see that the conditional distribution of $\mathcal{L}$ given $|\mathcal{L}| = \ell$ is the same as the (unconditional) distribution of $\mathcal{Z}_\ell$. In particular, if $\mathcal{A}_p$ achieves label complexity Λ_p, and $\hat{h}_n$ is the classifier returned by Meta-Algorithm 0 applied to

$\mathcal{A}_p$, then for any $n = \Omega\left(\sqrt{\Lambda_p(\varepsilon, f, \mathcal{P}) \log(1/\varepsilon)}\right)$ chosen so that $\ell_{n,\varepsilon} \geq \Lambda_p(\varepsilon, f, \mathcal{P})$, we have

$$\mathbb{E}\left[\mathrm{er}\left(\hat{h}_n\right)\right] \leq \varepsilon + \sup_{\ell \geq \ell_{n,\varepsilon}} \mathbb{E}\left[\mathrm{er}\left(\mathcal{A}_p(\mathcal{Z}_\ell)\right)\right] \leq \varepsilon + \sup_{\ell \geq \Lambda_p(\varepsilon, f, \mathcal{P})} \mathbb{E}\left[\mathrm{er}\left(\mathcal{A}_p(\mathcal{Z}_\ell)\right)\right] \leq 2\varepsilon.$$

This indicates the active learning algorithm achieves label complexity Λ_a with $\Lambda_a(2\varepsilon, f, \mathcal{P}) = O\left(\sqrt{\Lambda_p(\varepsilon, f, \mathcal{P}) \log(1/\varepsilon)}\right)$. In particular, if $\infty > \Lambda_p(\varepsilon, f, \mathcal{P}) = \omega(\log(1/\varepsilon))$, then $\Lambda_a(2\varepsilon, f, \mathcal{P}) = o(\Lambda_p(\varepsilon, f, \mathcal{P}))$. Therefore, Meta-Algorithm 0 is a universal activizer for the space of threshold classifiers.

In contrast, consider the problem of learning interval classifiers (Example 2). In this case, suppose the target function f has $\mathcal{P}(x : f(x) = +1) = 0$, and that $\mathcal{P}$ is uniform in $[0,1]$. Since (with probability one) every $Y_i = -1$, we have $V = \{h_{[a,b]} : \{X_1, \ldots, X_{\lfloor n/2 \rfloor}\} \cap [a,b] = \emptyset\}$. But this contains classifiers $h_{[a,a]}$ for every $a \in (0,1) \setminus \{X_1, \ldots, X_{\lfloor n/2 \rfloor}\}$, so that $\mathrm{DIS}(V) = (0,1) \setminus \{X_1, \ldots, X_{\lfloor n/2 \rfloor}\}$. Thus, $\mathcal{P}(\mathrm{DIS}(V)) = 1$, and $|\mathcal{L}| = O(n)$; that is, $\mathcal{A}_p$ gets run with no more labeled examples than simple passive learning would use. This indicates we should not expect Meta-Algorithm 0 to be a universal activizer for interval classifiers. Below, we formalize this by constructing a passive learning algorithm $\mathcal{A}_p$ that Meta-Algorithm 0 does not activize in scenarios of this type.

3.2 The Limiting Region of Disagreement

In this subsection, we generalize the examples from the previous subsection. Specifically, we prove that the performance of Meta-Algorithm 0 is intimately tied to a particular limiting set, referred to as the *disagreement core*. A similar definition was given by Balcan, Hanneke, and Vaughan (2010) (there referred to as the *boundary*, for reasons that will become clear below); it is also related to certain quantities in the work of Hanneke (2007b, 2011) described below in Section 5.1.

Definition 4 *Define the* disagreement core *of a classifier f with respect to a set of classifiers $\mathcal{H}$ and probability measure P as*

$$\partial_{\mathcal{H},P} f = \lim_{r \to 0} \mathrm{DIS}\left(\mathrm{B}_{\mathcal{H},P}(f, r)\right).$$

When $P = \mathcal{P}$, the data distribution on $\mathcal{X}$, and $\mathcal{P}$ is clear from the context, we abbreviate this as $\partial_{\mathcal{H}} f = \partial_{\mathcal{H},\mathcal{P}} f$; if additionally $\mathcal{H} = \mathbb{C}$, the full concept space, which is clear from the context, we further abbreviate this as $\partial f = \partial_{\mathbb{C}} f = \partial_{\mathbb{C},\mathcal{P}} f$.

As we will see, disagreement-based algorithms often tend to focus their label requests around the disagreement core of the target function. As such, the concept of the disagreement core will be essential in much of our discussion below. We therefore go through a few examples to build intuition about this concept and its properties. Perhaps the simplest example to start with is $\mathbb{C}$ as the class of *threshold* classifiers (Example 1), under $\mathcal{P}$ uniform on $[0,1]$. For any $h_z \in \mathbb{C}$ and sufficiently small $r > 0$, $\mathrm{B}(f, r) = \{h_{z'} : |z' - z| \leq r\}$, and $\mathrm{DIS}(\mathrm{B}(f, r)) = [z - r, z + r)$. Therefore, $\partial h_z = \lim_{r \to 0} \mathrm{DIS}(\mathrm{B}(h_z, r)) = \lim_{r \to 0} [z - r, z + r) = \{z\}$. Thus, in this case, the disagreement core of h_z with respect to $\mathbb{C}$ and $\mathcal{P}$ is precisely the decision boundary of the classifier. As a slightly more involved example, consider again the example of *interval* classifiers (Example 2), again under $\mathcal{P}$ uniform on $[0,1]$. Now for any $h_{[a,b]} \in \mathbb{C}$ with $b - a > 0$, for any sufficiently small $r > 0$, $\mathrm{B}(h_{[a,b]}, r) = \{h_{[a',b']} : |a - a'| + |b - b'| \leq r\}$, and $\mathrm{DIS}(\mathrm{B}(h_{[a,b]}, r)) = [a - r, a + r) \cup (b - r, b + r]$. Therefore, $\partial h_{[a,b]} = \lim_{r \to 0} \mathrm{DIS}(\mathrm{B}(h_{[a,b]}, r)) = \lim_{r \to 0} [a - r, a + r) \cup (b - r, b + r] = \{a, b\}$. Thus, in this case as well, the disagreement core of $h_{[a,b]}$ with respect to $\mathbb{C}$ and $\mathcal{P}$ is again the decision boundary of the classifier.

As the above two examples illustrate, ∂f often corresponds to the decision boundary of f in some geometric interpretation of $\mathcal{X}$ and f. Indeed, under fairly general conditions on $\mathbb{C}$ and $\mathcal{P}$, the disagreement core of f does correspond to (a subset of) the set of points dividing the two label regions of f; for instance, Friedman (2009) derives sufficient conditions, under which this is the case. In these cases, the behavior of disagreement-based active learning algorithms can often be interpreted in the intuitive terms of seeking label requests near the decision boundary of the target function, to refine an estimate of that boundary. However, in some more subtle scenarios this is no longer the case, for interesting reasons. To illustrate this, let us continue the example of interval classifiers from above, but now consider $h_{[a,a]}$ (i.e., $h_{[a,b]}$ with $a = b$). This time, for any $r \in (0,1)$ we have $\mathrm{B}(h_{[a,a]}, r) = \{h_{[a',b']} \in \mathbb{C} : b' - a' \leq r\}$, and $\mathrm{DIS}(\mathrm{B}(h_{[a,a]}, r)) = (0,1)$. Therefore, $\partial h_{[a,a]} = \lim_{r \to 0} \mathrm{DIS}(\mathrm{B}(h_{[a,a]}, r)) = \lim_{r \to 0} (0,1) = (0,1)$.

This example shows that in some cases, the disagreement core does not correspond to the decision boundary of the classifier, and indeed has $\mathcal{P}(\partial f) > 0$. Intuitively, as in the above example, this typically happens when the decision surface of the classifier is in some sense *simpler* than it could be. For instance, consider the space $\mathbb{C}$ of *unions of two intervals* (Example 3 with $i = 2$) under uniform $\mathcal{P}$. The classifiers $f \in \mathbb{C}$ with $\mathcal{P}(\partial f) > 0$ are precisely those representable (up to probability zero differences) as a single interval. The others (with $0 < z_1 < z_2 < z_3 < z_4 < 1$) have $\partial h_{\mathbf{z}} = \{z_1, z_2, z_3, z_4\}$. In these examples, the $f \in \mathbb{C}$ with $\mathcal{P}(\partial f) > 0$ are not only simpler than other nearby classifiers in $\mathbb{C}$, but they are also in some sense *degenerate* relative to the rest of $\mathbb{C}$; however, it turns out this is not always the case, as there exist scenarios $(\mathbb{C}, \mathcal{P})$, even with $d = 2$, and even with *countable* $\mathbb{C}$, for which *every* $f \in \mathbb{C}$ has $\mathcal{P}(\partial f) > 0$; in these cases, every classifier is in some important sense *simpler* than some other subset of nearby classifiers in $\mathbb{C}$.

In Section 3.3, we show that the label complexity of disagreement-based active learning is intimately tied to the disagreement core. In particular, scenarios where $\mathcal{P}(\partial f) > 0$, such as those mentioned above, lead to the conclusion that disagreement-based methods are sometimes insufficient for activized learning. This motivates the design of more sophisticated methods in Section 4, which overcome this deficiency, along with a corresponding refinement of the definition of "disagreement core " in Section 5.2 that eliminates the above issue with "simple" classifiers.

3.3 Necessary and Sufficient Conditions for Disagreement-Based Activized Learning

In the specific case of Meta-Algorithm 0, for large n we may intuitively expect it to focus its second batch of label requests in and around the disagreement core of the target function. Thus, whenever $\mathcal{P}(\partial f) = 0$, we should expect the label requests to be quite focused, and therefore the algorithm should achieve smaller label complexity compared to passive learning. On the other hand, if $\mathcal{P}(\partial f) > 0$, then the label requests will *not* become focused beyond a constant fraction of the space, so that the improvements achieved by Meta-Algorithm 0 over passive learning should be, at best, a constant factor. This intuition is formalized in the following general theorem, the proof of which is included in Appendix A.

Theorem 5 *For any VC class $\mathbb{C}$, Meta-Algorithm 0 is a universal activizer for $\mathbb{C}$ if and only if every $f \in \mathbb{C}$ and distribution $\mathcal{P}$ has $\mathcal{P}(\partial_{\mathbb{C},\mathcal{P}} f) = 0$.*

While the formal proof is given in Appendix A, the general idea is simple. As we always have $f \in V$, any $\hat{y}$ inferred in Step 6 must equal $f(x)$, so that all of the labels in $\mathcal{L}$ are correct. Also, as n grows large, classic results on passive learning imply the diameter of the set V will become small,

shrinking to zero as $n \to \infty$ (Vapnik and Chervonenkis, 1971; Vapnik, 1982; Blumer, Ehrenfeucht, Haussler, and Warmuth, 1989). Therefore, as $n \to \infty$, $\text{DIS}(V)$ should converge to a subset of ∂f, so that in the case $\mathcal{P}(\partial f) = 0$, we have $\hat{\Delta} \to 0$; thus $|\mathcal{L}| \gg n$, which implies an asymptotic strict improvement in label complexity over the passive algorithm $\mathcal{A}_p$ that $\mathcal{L}$ is fed into in Step 8. On the other hand, since ∂f is defined by classifiers arbitrarily close to f, it is unlikely that any finite sample of correctly labeled examples can contradict enough classifiers to make $\text{DIS}(V)$ significantly smaller than ∂f, so that we always have $\mathcal{P}(\text{DIS}(V)) \geq \mathcal{P}(\partial f)$. Therefore, if $\mathcal{P}(\partial f) > 0$, then $\hat{\Delta}$ converges to some nonzero constant, so that $|\mathcal{L}| = O(n)$, representing only a constant factor improvement in label complexity. In fact, as is implied from this sketch (and is proven in Appendix A), the targets f and distributions $\mathcal{P}$ for which Meta-Algorithm 0 achieves asymptotic strict improvements for all passive learning algorithms (for which f and $\mathcal{P}$ are nontrivial) are precisely those (and only those) for which $\mathcal{P}(\partial_{\mathbb{C},\mathcal{P}} f) = 0$.

There are some general conditions under which the zero-probability disagreement cores condition of Theorem 5 will hold. For instance, it is not difficult to show this will always hold when $\mathcal{X}$ is countable. Furthermore, with some effort one can show it will hold for most classes having VC dimension one (e.g., any countable $\mathbb{C}$ with $d = 1$). However, as we have seen, not all spaces $\mathbb{C}$ satisfy this zero-probability disagreement cores property. In particular, for the interval classifiers studied in Section 3.2, we have $\mathcal{P}(\partial h_{[a,a]}) = \mathcal{P}((0,1)) = 1$. Indeed, the aforementioned special cases aside, for *most* nontrivial spaces $\mathbb{C}$, one can construct distributions $\mathcal{P}$ that in some sense make $\mathbb{C}$ mimic the intervals problem, so that we should typically expect disagreement-based methods will *not* be activizers. For detailed discussions of various scenarios where the $\mathcal{P}(\partial_{\mathbb{C},\mathcal{P}} f) = 0$ condition is (or is not) satisfied for various $\mathbb{C}$, $\mathcal{P}$, and f, see the works of Hanneke (2009b), Hanneke (2007b), Hanneke (2011), Balcan, Hanneke, and Vaughan (2010), Friedman (2009), Wang (2009) and Wang (2011).

4. Beyond Disagreement: A Basic Activizer

Since the zero-probability disagreement cores condition of Theorem 5 is not always satisfied, we are left with the question of whether there could be other techniques for active learning, beyond simple disagreement-based methods, which could activize *every* passive learning algorithm for *every* VC class. In this section, we present an entirely new type of active learning algorithm, unlike anything in the existing literature, and we show that indeed it is a universal activizer for any class $\mathbb{C}$ of finite VC dimension.

4.1 A Basic Activizer

As mentioned, the case $\mathcal{P}(\partial f) = 0$ is already handled nicely by disagreement-based methods, since the label requests made in the second stage of Meta-Algorithm 0 will become focused into a small region, and $\mathcal{L}$ therefore grows faster than n. Thus, the primary question we are faced with is what to do when $\mathcal{P}(\partial f) > 0$. Since (loosely speaking) we have $\text{DIS}(V) \to \partial f$ in Meta-Algorithm 0, $\mathcal{P}(\partial f) > 0$ corresponds to scenarios where the label requests of Meta-Algorithm 0 will not become focused beyond a certain extent; specifically, as we show in Appendix B (Lemmas 35 and 36), $\mathcal{P}(\text{DIS}(V) \oplus \partial f) \to 0$ almost surely, where $\oplus$ is the symmetric difference, so that we expect Meta-Algorithm 0 will request labels for at least some constant fraction of the examples in $\mathcal{L}$.

On the one hand, this is definitely a major problem for disagreement-based methods, since it prevents them from improving over passive learning in those cases. On the other hand, if we do not

restrict ourselves to disagreement-based methods, we may actually be able to exploit properties of this scenario, so that it works to our *advantage*. In particular, in addition to the fact that $\mathcal{P}(\mathrm{DIS}(V) \oplus \partial_{\mathbb{C}} f) \to 0$, we show in Appendix B (Lemma 35) that $\mathcal{P}(\partial_V f \oplus \partial_{\mathbb{C}} f) = 0$ (almost surely) in Meta-Algorithm 0; this implies that for sufficiently large n, a random point x_1 in $\mathrm{DIS}(V)$ is likely to be in $\partial_V f$. We can exploit this fact by using x_1 to split V into two subsets: $V[(x_1,+1)]$ and $V[(x_1,-1)]$. Now, if $x_1 \in \partial_V f$, then (by definition of the disagreement core) $\inf_{h \in V[(x_1,+1)]} \mathrm{er}(h) = \inf_{h \in V[(x_1,-1)]} \mathrm{er}(h) = 0$. Therefore, for almost every point $x \notin \mathrm{DIS}(V[(x_1,+1)])$, the label agreed upon for x by classifiers in $V[(x_1,+1)]$ should be $f(x)$. Likewise, for almost every point $x \notin \mathrm{DIS}(V[(x_1,-1)])$, the label agreed upon for x by classifiers in $V[(x_1,-1)]$ should be $f(x)$. Thus, we can accurately *infer* the label of any point $x \notin \mathrm{DIS}(V[(x_1,+1)]) \cap \mathrm{DIS}(V[(x_1,-1)])$ (except perhaps a zero-probability subset). With these sets $V[(x_1,+1)]$ and $V[(x_1,-1)]$ in hand, there is no longer a need to request the labels of points for which either of them has agreement about the label, and we can focus our label requests to the region $\mathrm{DIS}(V[(x_1,+1)]) \cap \mathrm{DIS}(V[(x_1,-1)])$, which may be *much smaller* than $\mathrm{DIS}(V)$. Now if $\mathcal{P}(\mathrm{DIS}(V[(x_1,+1)]) \cap \mathrm{DIS}(V[(x_1,-1)])) \to 0$, then the label requests will become focused to a shrinking region, and by the same reasoning as for Theorem 5 we can asymptotically achieve strict improvements over passive learning by a method analogous to Meta-Algorithm 0 (with the above changes).

Already this provides a significant improvement over disagreement-based methods in many cases; indeed, in some cases (such as intervals) this fully addresses the nonzero-probability disagreement core issue in Theorem 5. In other cases (such as unions of two intervals), it does not completely address the issue, since for some targets we do not have $\mathcal{P}(\mathrm{DIS}(V[(x_1,+1)]) \cap \mathrm{DIS}(V[(x_1,-1)])) \to 0$. However, by repeatedly applying this same reasoning, we *can* address the issue in full generality. Specifically, if $\mathcal{P}(\mathrm{DIS}(V[(x_1,+1)]) \cap \mathrm{DIS}(V[(x_1,-1)])) \nrightarrow 0$, then $\mathrm{DIS}(V[(x_1,+1)]) \cap \mathrm{DIS}(V[(x_1,-1)])$ essentially converges to a region $\partial_{\mathbb{C}[(x_1,+1)]} f \cap \partial_{\mathbb{C}[(x_1,-1)]} f$, which has nonzero probability, and is nearly equivalent to $\partial_{V[(x_1,+1)]} f \cap \partial_{V[(x_1,-1)]} f$. Thus, for sufficiently large n, a random x_2 in $\mathrm{DIS}(V[(x_1,+1)]) \cap \mathrm{DIS}(V[(x_1,-1)])$ will likely be in $\partial_{V[(x_1,+1)]} f \cap \partial_{V[(x_1,-1)]} f$. In this case, we can repeat the above argument, this time splitting V into four sets $(V[(x_1,+1)][(x_2,+1)], V[(x_1,+1)][(x_2,-1)], V[(x_1,-1)][(x_2,+1)]$, and $V[(x_1,-1)][(x_2,-1)])$, each with infimum error rate equal zero, so that for a point x in the region of agreement of any of these four sets, the agreed-upon label will (almost surely) be $f(x)$, so that we can infer that label. Thus, we need only request the labels of those points in the *intersection* of all four regions of disagreement. We can further repeat this process as many times as needed, until we get a partition of V with shrinking probability mass in the intersection of the regions of disagreement, which (as above) can then be used to obtain asymptotic improvements over passive learning.

Note that the above argument can be written more concisely in terms of *shattering*. That is, any $x \in \mathrm{DIS}(V)$ is simply an x such that V shatters $\{x\}$; a point $x \in \mathrm{DIS}(V[(x_1,+1)]) \cap \mathrm{DIS}(V[(x_1,-1)])$ is simply one for which V shatters $\{x_1,x\}$, and for any $x \notin \mathrm{DIS}(V[(x_1,+1)]) \cap \mathrm{DIS}(V[(x_1,-1)])$, the label y we infer about x has the property that the set $V[(x,-y)]$ does not shatter $\{x_1\}$. This continues for each repetition of the above idea, with x in the intersection of the four regions of disagreement simply being one for which V shatters $\{x_1,x_2,x\}$, and so on. In particular, this perspective makes it clear that we need only repeat this idea at most d times to get a shrinking intersection region, since no set of $d+1$ points is shatterable. Note that there may be unobservable factors (e.g., the target function) determining the appropriate number of iterations of this idea sufficient to have a shrinking probability of requesting a label, while maintaining the accuracy of inferred labels. To address this,

we can simply try all $d+1$ possibilities, and then select one of the resulting $d+1$ classifiers via a kind of tournament of pairwise comparisons. Also, in order to reduce the probability of a mistaken inference due to $x_1 \notin \partial_V f$ (or similarly for later x_i), we can replace each single x_i with multiple samples, and then take a majority vote over whether to infer the label, and which label to infer if we do so; generally, we can think of this as estimating certain probabilities, and below we write these estimators as $\hat{P}_m$, and discuss the details of their implementation later. Combining Meta-Algorithm 0 with the above reasoning motivates a new type of active learning algorithm, referred to as Meta-Algorithm 1 below, and stated as follows.

Meta-Algorithm 1
Input: passive algorithm $\mathcal{A}_p$, label budget n
Output: classifier $\hat{h}$

---

0. Request the first $m_n = \lfloor n/3 \rfloor$ labels, $\{Y_1, \ldots, Y_{m_n}\}$, and let $t \leftarrow m_n$
1. Let $V = \{h \in \mathbb{C} : \mathrm{er}_{m_n}(h) = 0\}$
2. For $k = 1, 2, \ldots, d+1$
3. $\quad \hat{\Delta}^{(k)} \leftarrow \hat{P}_{m_n}\left(x : \hat{P}\left(S \in \mathcal{X}^{k-1} : V \text{ shatters } S \cup \{x\} | V \text{ shatters } S\right) \geq 1/2\right)$
4. $\quad$ Let $\mathcal{L}_k \leftarrow \{\}$
5. $\quad$ For $m = m_n + 1, \ldots, m_n + \lfloor n/(6 \cdot 2^k \hat{\Delta}^{(k)}) \rfloor$
6. $\quad\quad$ If $\hat{P}_m\left(S \in \mathcal{X}^{k-1} : V \text{ shatters } S \cup \{X_m\} | V \text{ shatters } S\right) \geq 1/2$ and $t < \lfloor 2n/3 \rfloor$
7. $\quad\quad\quad$ Request the label Y_m of X_m, and let $\hat{y} \leftarrow Y_m$ and $t \leftarrow t+1$
8. $\quad\quad$ Else, let $\hat{y} \leftarrow \underset{y \in \{-1,+1\}}{\mathrm{argmax}}\ \hat{P}_m\left(S \in \mathcal{X}^{k-1} : V[(X_m, -y)] \text{ does not shatter } S | V \text{ shatters } S\right)$
9. $\quad\quad$ Let $\mathcal{L}_k \leftarrow \mathcal{L}_k \cup \{(X_m, \hat{y})\}$
10. Return ActiveSelect$(\{\mathcal{A}_p(\mathcal{L}_1), \mathcal{A}_p(\mathcal{L}_2), \ldots, \mathcal{A}_p(\mathcal{L}_{d+1})\}, \lfloor n/3 \rfloor, \{X_{m_n + \max_k |\mathcal{L}_k| + 1}, \ldots\})$

Subroutine: ActiveSelect
Input: set of classifiers $\{h_1, h_2, \ldots, h_N\}$, label budget m, sequence of unlabeled examples $\mathcal{U}$
Output: classifier $\hat{h}$

---

0. For each $j, k \in \{1, 2, \ldots, N\}$ s.t. $j < k$,
1. $\quad$ Let R_{jk} be the first $\left\lfloor \frac{m}{j(N-j)\ln(eN)} \right\rfloor$ points in $\mathcal{U} \cap \{x : h_j(x) \neq h_k(x)\}$ (if such values exist)
2. $\quad$ Request the labels for R_{jk} and let Q_{jk} be the resulting set of labeled examples
3. $\quad$ Let $m_{kj} = \mathrm{er}_{Q_{jk}}(h_k)$
4. Return $h_{\hat{k}}$, where $\hat{k} = \max\left\{k \in \{1, \ldots, N\} : \max_{j<k} m_{kj} \leq 7/12\right\}$

Meta-Algorithm 1 is stated as a function of three types of estimated probabilities: namely,

$$\hat{P}_m\left(S \in \mathcal{X}^{k-1} : V \text{ shatters } S \cup \{x\} \middle| V \text{ shatters } S\right),$$
$$\hat{P}_m\left(S \in \mathcal{X}^{k-1} : V[(x, -y)] \text{ does not shatter } S \middle| V \text{ shatters } S\right),$$
$$\text{and } \hat{P}_m\left(x : \hat{P}\left(S \in \mathcal{X}^{k-1} : V \text{ shatters } S \cup \{x\} \middle| V \text{ shatters } S\right) \geq 1/2\right).$$

These can be defined in a variety of ways to make this a universal activizer for $\mathbb{C}$. Generally, the only requirement seems to be that they converge to the appropriate respective probabilities at a

sufficiently fast rate. For the theorem stated below regarding Meta-Algorithm 1, we will take the specific definitions stated in Appendix B.1.

Meta-Algorithm 1 requests labels in three batches: one to initially prune down the version space V, a second one to construct the labeled samples $\mathcal{L}_k$, and a third batch to select among the $d+1$ classifiers $\mathcal{A}_p(\mathcal{L}_k)$ in the ActiveSelect subroutine. As before, the choice of the number of (unlabeled) examples to process in the second batch guarantees (by a Chernoff bound) that the "$t < \lfloor 2n/3 \rfloor$" constraint in Step 6 is redundant. The mechanism for requesting labels in the second batch is motivated by the reasoning outlined above, using the shatterable sets S to split V into 2^{k-1} subsets, each of which approximates the target with high probability (for large n), and then checking whether the new point x is in the regions of disagreement for all 2^{k-1} subsets (by testing shatterability of $S \cup \{x\}$). To increase confidence in this test, we use many such S sets, and let them vote on whether or not to request the label (Step 6). As mentioned, if x is not in the region of disagreement for one of these 2^{k-1} subsets (call it V'), the agreed-upon label y has the property that $V[(x,-y)]$ does not shatter S (since $V[(x,-y)]$ does not intersect with V', which represents one of the 2^{k-1} labelings required to shatter S). Therefore, we infer that this label y is the correct label of x, and again we vote over many such S sets to increase confidence in this choice (Step 8). As mentioned, this reasoning leads to correctly inferred labels in Step 8 as long as n is sufficiently large *and* $\mathcal{P}^{k-1}(S \in \mathcal{X}^{k-1} : V \text{ shatters } S) \nrightarrow 0$. In particular, we are primarily interested in the largest value of k for which this reasoning holds, since this is the value at which the probability of requesting a label (Step 7) shrinks to zero as $n \to \infty$. However, since we typically cannot predict a priori what this largest valid k value will be (as it is target-dependent), we try all $d+1$ values of k, to generate $d+1$ hypotheses, and then use a simple pairwise testing procedure to select among them; note that we need at most try $d+1$ values, since V definitely cannot shatter any $S \in \mathcal{X}^{d+1}$. We will see that the ActiveSelect subroutine is guaranteed to select a classifier with error rate never significantly larger than the best among the classifiers given to it (say within a factor of 2, with high probability). Therefore, in the present context, we need only consider whether some k has a set $\mathcal{L}_k$ with correct labels *and* $|\mathcal{L}_k| \gg n$.

4.2 Examples

In the next subsection, we state a general result for Meta-Algorithm 1. But first, to illustrate how this procedure operates, we walk through its behavior on our usual examples; as we did for the examples of Meta-Algorithm 0, to simplify the explanation, for now we will ignore the fact that the $\hat{P}_m$ values are estimates, as well as the "$t < \lfloor 2n/3 \rfloor$" constraint of Step 6, and the issue of effectiveness of ActiveSelect; in the proofs of the general results below, we will show that these issues do not fundamentally change the analysis. For now, we merely focus on showing that some k has $\mathcal{L}_k$ correctly labeled and $|\mathcal{L}_k| \gg n$.

For threshold classifiers (Example 1), we have $d = 1$. In this case, the $k = 1$ round of the algorithm is essentially identical to Meta-Algorithm 0 (recall our conventions that $\mathcal{X}^0 = \{\varnothing\}$, $\mathcal{P}(\mathcal{X}^0) = 1$, and V shatters $\varnothing$ iff $V \neq \{\}$), and we therefore have $|\mathcal{L}_1| \gg n$, as discussed previously, so that Meta-Algorithm 1 is a universal activizer for threshold classifiers.

Next consider interval classifiers (Example 2), with $\mathcal{P}$ uniform on $[0,1]$; in this case, we have $d = 2$. If $f = h_{[a,b]}$ for $a < b$, then again the $k = 1$ round behaves essentially the same as Meta-Algorithm 0, and since we have seen $\mathcal{P}(\partial h_{[a,b]}) = 0$ in this case, we have $|\mathcal{L}_1| \gg n$. However, the behavior becomes far more interesting when $f = h_{[a,a]}$, which was precisely the case that prevented

Meta-Algorithm 0 from improving over passive learning. In this case, as we know from above, the $k = 1$ round will have $|\mathcal{L}_1| = O(n)$, so that we need to consider larger values of k to identify improvements. In this case, the $k = 2$ round behaves as follows. With probability 1, the initial $\lfloor n/3 \rfloor$ labels used to define V will all be negative. Thus, V is precisely the set of intervals that do not contain any of the initial $\lfloor n/3 \rfloor$ points. Now consider any $S = \{x_1\} \in \mathcal{X}^1$, with x_1 not equal to any of these initial $\lfloor n/3 \rfloor$ points, and consider any $x \notin \{x_1, X_1, \ldots, X_{\lfloor n/3 \rfloor}\}$. First note that V shatters S, since we can optionally put a small interval around x_1 using an element of V. If there is a point x' among the initial $\lfloor n/3 \rfloor$ *between* x and x_1, then any $h_{[a,b]} \in V$ with $x \in [a,b]$ cannot also have $x_1 \in [a,b]$, as it would also contain the observed negative point between them. Thus, V does *not* shatter $\{x_1, x\} = S \cup \{x\}$, so that this S will vote to infer (rather than request) the label of x in Step 6. Furthermore, we see that $V[(x,+1)]$ does not shatter S, while $V[(x,-1)]$ does shatter S, so that this S would also vote for the label $\hat{y} = -1$ in Step 8. For sufficiently large n, with high probability, any given x not equal one of the initial $\lfloor n/3 \rfloor$ should have *most* (probability at least $1 - O(n^{-1} \log n)$) of the possible x_1 values separated from it by at least one of the initial $\lfloor n/3 \rfloor$ points, so that the outcome of the vote in Step 6 will be a decision to infer (not request) the label, and the vote in Step 8 will be for -1. Since, with probability one, every $X_m \neq a$, we have every $Y_m = -1$, so that every point in $\mathcal{L}_2$ is labeled correctly. This also indicates that, for sufficiently large n, we have $\mathcal{P}(x : \mathcal{P}^1(S \in \mathcal{X}^1 : V \text{ shatters } S \cup \{x\} | V \text{ shatters } S) \geq 1/2) = 0$, so that the size of $\mathcal{L}_2$ is only limited by the precision of estimation in $\hat{P}_{m_n}$ in Step 3. Thus, as long as we implement $\hat{P}_{m_n}$ so that its value is at most $o(1)$ larger than the true probability, we can guarantee $|\mathcal{L}_2| \gg n$.

The unions of i intervals example (Example 3), again under $\mathcal{P}$ uniform on $[0,1]$, is slightly more involved; in this case, the appropriate value of k to consider for any given target depends on the minimum number of intervals necessary to represent the target function (up to zero-probability differences). If j intervals are required for this, then the appropriate value is $k = i - j + 1$. Specifically, suppose the target is minimally representable as a union of $j \in \{1, \ldots, i\}$ intervals of nonzero width: $[z_1, z_2] \cup [z_3, z_4] \cup \cdots \cup [z_{2j-1}, z_{2j}]$: that is, $z_1 < z_2 < \ldots < z_{2j-1} < z_{2j}$. Every target in $\mathbb{C}$ has distance zero to some classifier of this type, and will agree with that classifier on all samples with probability one, so we lose no generality by assuming all j intervals have nonzero width. Then consider any $x \in (0,1)$ and $S = \{x_1, \ldots, x_{i-j}\} \in \mathcal{X}^{i-j}$ such that, between any pair of elements of $S \cup \{x\} \cup \{z_1, \ldots, z_{2j}\}$, there is at least one of the initial $\lfloor n/3 \rfloor$ points, and none of $S \cup \{x\}$ are themselves equal to any of those initial points. First note that V shatters S, since for any x_ℓ not in one of the $[z_{2p-1}, z_{2p}]$ intervals (i.e., negative), we may optionally add an interval $[x_\ell, x_\ell]$ while staying in V, and for any x_ℓ in one of the $[z_{2p-1}, z_{2p}]$ intervals (i.e., positive), we may optionally split $[z_{2p-1}, z_{2p}]$ into two intervals to barely exclude the point x_ℓ (and a small neighborhood around it), by adding at most one interval to the representation; thus, in total we need to add at most $i - j$ intervals to the representation, so that the largest number of intervals used by any of these 2^{i-j} classifiers involved in shattering is i, as required; furthermore, note that one of these 2^{i-j} classifiers actually requires i intervals. Now for any such x and S as above, since one of the 2^{i-j} classifiers in V used to shatter S requires i intervals to represent it, and x is separated from each element of $S \cup \{z_1, \ldots, z_{2j}\}$ by a labeled example, we see that V cannot shatter $S \cup \{x\}$. Furthermore, if $f(x) = y$, then any labeled example to the immediate left or right of x is also labeled y, and in particular among the 2^{i-j} classifiers h from V that shatter S, the one h that requires i intervals to represent must also have $h(x) = y$, so that $V[(x,-y)]$ does not shatter S. Thus, any set S satisfying this separation property will vote to infer (rather than request) the label of x in Step 6, and will vote for the label $f(x)$ in Step 8. Furthermore, for sufficiently large n, for any given x separated from $\{z_1, \ldots, z_{2j}\}$ by $\{X_1, \ldots, X_{\lfloor n/3 \rfloor}\}$,

with high probability most of the sets $S \in \mathcal{X}^{i-j}$ will satisfy this pairwise separation property, and therefore so will most of the shatterable sets $S \in \mathcal{X}^{i-j}$, so that the overall outcome of the votes will favor inferring the label of x, and in particular inferring the label $f(x)$ for x. On the other hand, for x not satisfying this property (i.e., not separated from some z_p by any of the initial $\lfloor n/3 \rfloor$ examples), for any set S as above, V *can* shatter $S \cup \{x\}$, since we can optionally increase or decrease this z_p to include or exclude x from the associated interval, in addition to optionally adding the extra intervals to shatter S; therefore, by the same reasoning as above, for sufficiently large n, any such x *will* satisfy the condition in Step 6, and thus have its label requested. Thus, for sufficiently large n, every example in $\mathcal{L}_{i-j+1}$ will be labeled correctly. Finally, note that with probability one, the set of points x separated from each of the z_p values by at least one of the $\lfloor n/3 \rfloor$ initial points has probability approaching one as $n \to \infty$, so that again we have $|\mathcal{L}_{i-j+1}| \gg n$.

The above examples give some intuition about the operation of this procedure. Next, we turn to general results showing that this type of improvement generally holds.

4.3 General Results on Activized Learning

Returning to the abstract setting, we have the following general theorem, representing one of the main results of this paper. Its proof is included in Appendix B.

Theorem 6 *For any VC class $\mathbb{C}$, Meta-Algorithm 1 is a universal activizer for $\mathbb{C}$.*

This result is interesting both for its strength and generality. Recall that it means that given any passive learning algorithm $\mathcal{A}_p$, the active learning algorithm obtained by providing $\mathcal{A}_p$ as input to Meta-Algorithm 1 achieves a label complexity that strongly dominates that of $\mathcal{A}_p$ for all nontrivial distributions $\mathcal{P}$ and target functions $f \in \mathbb{C}$. Results of this type were not previously known. The specific technical advance over existing results (namely, those of Balcan, Hanneke, and Vaughan, 2010) is the fact that Meta-Algorithm 1 has no direct dependence on the distribution $\mathcal{P}$; as mentioned earlier, the (very different) approach proposed by Balcan, Hanneke, and Vaughan (2010) has a strong direct dependence on the distribution, to the extent that the distribution-dependence in that approach cannot be removed by merely replacing certain calculations with data-dependent estimators (as we did in Meta-Algorithm 1). In the proof, we actually show a somewhat more general result: namely, that Meta-Algorithm 1 achieves these asymptotic improvements for any target function f in the *closure* of $\mathbb{C}$ (i.e., any f such that $\forall r > 0, \mathrm{B}(f,r) \neq \emptyset$).

The following corollary is one concrete implication of Theorem 6.

Corollary 7 *For any VC class $\mathbb{C}$, there exists an active learning algorithm achieving a label complexity Λ_a such that, for all target functions $f \in \mathbb{C}$ and distributions $\mathcal{P}$,*

$$\Lambda_a(\varepsilon, f, \mathcal{P}) = o(1/\varepsilon).$$

Proof The *one-inclusion graph* passive learning algorithm of Haussler, Littlestone, and Warmuth (1994) is known to achieve label complexity at most d/ε, for every target function $f \in \mathbb{C}$ and distribution $\mathcal{P}$. Thus, Theorem 6 implies that the (Meta-Algorithm 1)-activized one-inclusion graph algorithm satisfies the claim. ■

As a byproduct, Theorem 6 also establishes the basic fact that there *exist* activizers. In some sense, this observation opens up a new realm for exploration: namely, characterizing the *properties*

that activizers can possess. This topic includes a vast array of questions, many of which deal with whether activizers are capable of *preserving* various properties of the given passive algorithm (e.g., margin-based dimension-independence, minimaxity, admissibility, etc.). Section 7 describes a variety of enticing questions of this type. In the sections below, we will consider quantifying how large the gap in label complexity between the given passive learning algorithm and the resulting activized algorithm can be. We will additionally study the effects of label noise on the possibility of activized learning.

4.4 Implementation and Efficiency

Meta-Algorithm 1 typically also has certain desirable efficiency guarantees. Specifically, suppose that for any m labeled examples Q, there is an algorithm with $\text{poly}(d \cdot m)$ running time that finds some $h \in \mathbb{C}$ with $\text{er}_Q(h) = 0$ if one exists, and otherwise returns a value indicating that no such h exists in $\mathbb{C}$; for many concept spaces there are known methods with this capability (e.g., linear or polynomial separators, rectangles, k-DNF) (Khachiyan, 1979; Karmarkar, 1984; Valiant, 1984; Kearns and Vazirani, 1994), while for others this is known to be hard (e.g., k-term DNF, bounded-size decision trees) (Pitt and Valiant, 1988; Alekhnovich, Braverman, Feldman, Klivans, and Pitassi, 2004). Given such a subroutine, we can create an efficient implementation of the main body of Meta-Algorithm 1. Specifically, rather than explicitly representing V in Step 1, we can simply store the set $Q_0 = \{(X_1, Y_1), \ldots, (X_{m_n}, Y_{m_n})\}$. Then for any step in the algorithm where we need to test whether V shatters a set R, we can simply try all $2^{|R|}$ possible labelings of R, and for each one temporarily add these $|R|$ additional labeled examples to Q_0 and check whether there is an $h \in \mathbb{C}$ consistent with all of the labels. At first, it might seem that these 2^k evaluations would be prohibitive; however, supposing $\hat{P}_{m_n}$ is implemented so that it is $\Omega(1/\text{poly}(n))$ (as it is in Appendix B.1), note that the loop beginning at Step 5 executes a nonzero number of times only if $n/\hat{\Delta}^{(k)} > 2^k$, so that $2^k \leq \text{poly}(n)$; we can easily add a condition that skips the step of calculating $\hat{\Delta}^{(k)}$ if 2^k exceeds this $\text{poly}(n)$ lower bound on $n/\hat{\Delta}^{(k)}$, so that even those shatterability tests can be skipped in this case. Thus, for the actual occurrences of it in the algorithm, testing whether V shatters R requires only $\text{poly}(n) \cdot \text{poly}(d \cdot (|Q_0| + |R|))$ time. The total number of times this test is performed in calculating $\hat{\Delta}^{(k)}$ (from Appendix B.1) is itself only $\text{poly}(n)$, and the number of iterations of the loop in Step 5 is at most $n/\hat{\Delta}^{(k)} = \text{poly}(n)$. Determining the label $\hat{y}$ in Step 8 can be performed in a similar fashion. So in general, the total running time of the main body of Meta-Algorithm 1 is $\text{poly}(d \cdot n)$.

The only remaining question is the efficiency of the final step. Of course, we can require $\mathcal{A}_p$ to have running time polynomial in the size of its input set (and d). But beyond this, we must consider the efficiency of the ActiveSelect subroutine. This actually turns out to have some subtleties involved. The way it is stated above is simple and elegant, but not always efficient. Specifically, we have no a priori bound on the number of unlabeled examples the algorithm must process before finding a point X_m where $h_j(X_m) \neq h_k(X_m)$. Indeed, if $\mathcal{P}(x : h_j(x) \neq h_k(x)) = 0$, we may effectively need to examine the entire infinite sequence of X_m values to determine this. Fortunately, these problems can be corrected without difficulty, simply by truncating the search at a predetermined number of points. Specifically, rather than taking the next $\lfloor m/\binom{N}{2} \rfloor$ examples for which h_j and h_k disagree, simply restrict ourselves to at most this number, or at most the number of such points among the next M unlabeled examples. In Appendix B, we show that ActiveSelect, as originally stated, has a high-probability $(1 - \exp\{-\Omega(m)\})$ guarantee that the classifier it selects has error rate at most twice the best of the N it is given. With the modification to truncate the search at M unlabeled exam-

ples, this guarantee is increased to $\min_k \text{er}(h_k) + \max\{\text{er}(h_k), m/M\}$. For the concrete guarantee of Corollary 7, it suffices to take $M \gg m^2$. However, to guarantee the modified ActiveSelect can still be used in Meta-Algorithm 1 while maintaining (the stronger) Theorem 6, we need M at least as big as $\Omega\left(\min\left\{\exp\left\{m^c\right\}, m/\min_k \text{er}(h_k)\right\}\right)$, for any constant $c > 0$. In general, if we have a $1/\text{poly}(n)$ lower bound on the error rate of the classifier produced by $\mathcal{A}_p$ for a given number of labeled examples as input, we can set M as above using this lower bound in place of $\min_k \text{er}(h_k)$, resulting in an efficient version of ActiveSelect that still guarantees Theorem 6. However, it is presently not known whether there always exist universal activizers for $\mathbb{C}$ that are efficient (either $\text{poly}(d \cdot n)$ or $\text{poly}(d/\varepsilon)$ running time) when the above assumptions on efficiency of $\mathcal{A}_p$ and finding $h \in \mathbb{C}$ with $\text{er}_Q(h) = 0$ hold.

5. The Magnitudes of Improvements

In the previous section, we saw that we can always improve the label complexity of a passive learning algorithm by activizing it. However, there remains the question of how large the gap is between the passive algorithm's label complexity and the activized algorithm's label complexity. In the present section, we refine the above procedures to take greater advantage of the sequential nature of active learning. For each, we characterize the improvements it achieves relative to any given passive algorithm.

As a byproduct, this provides concise sufficient conditions for *exponential* gains, addressing an open problem of Balcan, Hanneke, and Vaughan (2010). Specifically, consider the following definition, essentially similar to one explored by Balcan, Hanneke, and Vaughan (2010).

Definition 8 *For a concept space $\mathbb{C}$ and distribution $\mathcal{P}$, we say that $(\mathbb{C}, \mathcal{P})$ is* learnable at an exponential rate *if there exists an active learning algorithm achieving label complexity Λ such that, $\forall f \in \mathbb{C}$, $\Lambda(\varepsilon, f, \mathcal{P}) \in \text{Polylog}(1/\varepsilon)$. We further say $\mathbb{C}$ is learnable at an exponential rate if there exists an active learning algorithm achieving label complexity Λ such that, for all distributions $\mathcal{P}$ and all $f \in \mathbb{C}$, $\Lambda(\varepsilon, f, \mathcal{P}) \in \text{Polylog}(1/\varepsilon)$.*

5.1 The Label Complexity of Disagreement-Based Active Learning

As before, to establish a foundation to build upon, we begin by studying the label complexity gains achievable by disagreement-based active learning. From above, we already know that disagreement-based active learning is not sufficient to achieve the best possible gains; but as before, it will serve as a suitable starting place to gain intuition for how we might approach the problem of improving Meta-Algorithm 1 and quantifying the improvements achievable over passive learning by the resulting more sophisticated methods.

The upper bounds on the label complexity of disagreement-based learning in this subsection are essentially already known and available in the published literature (though in a slightly less general form). Specifically, we review (a modified version of) the method of Cohn, Atlas, and Ladner (1994), referred to as Meta-Algorithm 2 below, which was historically the original disagreement-based active learning algorithm. We then state the known results on the label complexities achievable by this method, in terms of a quantity known as the disagreement coefficient; that result is due to Hanneke (2011, 2007b). We further provide a novel lower bound on the label complexity of this method, again in terms of the disagreement coefficient; in particular, this shows that the stated upper bounds represent a fairly tight analysis of this method.

5.1.1 THE CAL ACTIVE LEARNING ALGORITHM

To begin, we consider the following simple disagreement-based method, typically referred to as CAL after its discoverers Cohn, Atlas, and Ladner (1994), though the version here is slightly modified compared to the original (see below). It essentially represents a refinement of Meta-Algorithm 0 to take greater advantage of the sequential aspect of active learning. That is, rather than requesting only two batches of labels, as in Meta-Algorithm 0, this method updates the version space after every label request, thus focusing the region of disagreement (and therefore the region in which it requests labels) after each label request.

Meta-Algorithm 2
Input: passive algorithm $\mathcal{A}_p$, label budget n
Output: classifier $\hat{h}$

---

0. $V \leftarrow \mathbb{C}, t \leftarrow 0, m \leftarrow 0, \mathcal{L} \leftarrow \{\}$
1. While $t < \lceil n/2 \rceil$ and $m \leq 2^n$
2. $\quad m \leftarrow m+1$
3. $\quad$ If $X_m \in \mathrm{DIS}(V)$
4. $\quad\quad$ Request the label Y_m of X_m and let $t \leftarrow t+1$
5. $\quad\quad$ Let $V \leftarrow V[(X_m, Y_m)]$
6. Let $\hat{\Delta} \leftarrow \hat{P}_m(\mathrm{DIS}(V))$
7. Do $\lfloor n/(6\hat{\Delta}) \rfloor$ times
8. $\quad m \leftarrow m+1$
9. $\quad$ If $X_m \in \mathrm{DIS}(V)$ and $t < n$
10. $\quad\quad$ Request the label Y_m of X_m and let $\hat{y} \leftarrow Y_m$ and $t \leftarrow t+1$
11. $\quad$ Else let $\hat{y} = h(X_m)$ for an arbitrary $h \in V$
12. $\quad$ Let $\mathcal{L} \leftarrow \mathcal{L} \cup \{(X_m, \hat{y})\}$ and $V \leftarrow V[(X_m, \hat{y})]$
13. Return $\mathcal{A}_p(\mathcal{L})$

The procedure is specified in terms of an estimator $\hat{P}_m$; for our purposes, we define this as in (13) of Appendix B.1 (with $k = 1$ there). Every example X_m added to the set $\mathcal{L}$ in Step 12 either has its label requested (Step 10) or inferred (Step 11). By the same Chernoff bound argument mentioned for the previous methods, we are guaranteed (with high probability) that the "$t < n$" constraint in Step 9 is always satisfied when $X_m \in \mathrm{DIS}(V)$. Since we assume $f \in \mathbb{C}$, an inductive argument shows that we will always have $f \in V$ as well; thus, every label requested *or* inferred will agree with f, and therefore the labels in $\mathcal{L}$ are all correct.

As with Meta-Algorithm 0, this method has two stages to it: one in which we focus on reducing the version space V, and a second in which we focus on constructing a set of labeled examples to feed into the passive algorithm. The original algorithm of Cohn, Atlas, and Ladner (1994) essentially used only the first stage, and simply returned any classifier in V after exhausting its budget for label requests. Here we have added the second stage (Steps 6-13) so that we can guarantee a certain conditional independence (given $|\mathcal{L}|$) among the examples fed into the passive algorithm, which is important for the general results (Theorem 10 below). Hanneke (2011) showed that the original (simpler) algorithm achieves the (less general) label complexity bound of Corollary 11 below.

5.1.2 EXAMPLES

Not surprisingly, by essentially the same argument as Meta-Algorithm 0, one can show Meta-Algorithm 2 satisfies the claim in Theorem 5. That is, Meta-Algorithm 2 is a universal activizer for $\mathbb{C}$ if and only if $\mathcal{P}(\partial f) = 0$ for every $\mathcal{P}$ and $f \in \mathbb{C}$. However, there are further results known on the label complexity achieved by Meta-Algorithm 2. Specifically, to illustrate the types of improvements achievable by Meta-Algorithm 2, consider our usual toy examples; as before, to simplify the explanation, for these examples we ignore the fact that $\hat{P}_m$ is only an estimate, as well as the "$t < n$" constraint in Step 9 (both of which will be addressed in the general results below).

First, consider threshold classifiers (Example 1) under a uniform $\mathcal{P}$ on $[0,1]$, and suppose $f = h_z \in \mathbb{C}$. Suppose the given passive algorithm has label complexity Λ_p. To get expected error at most ε in Meta-Algorithm 2, it suffices to have $|\mathcal{L}| \geq \Lambda_p(\varepsilon/2, f, \mathcal{P})$ with probability at least $1-\varepsilon/2$. Starting from any particular V set obtained in the algorithm, call it V_0, the set $\mathrm{DIS}(V_0)$ is simply the region between the largest negative example observed so far (say z_ℓ) and the smallest positive example observed so far (say z_r). With probability at least $1-\varepsilon/n$, at least one of the next $O(\log(n/\varepsilon))$ examples in this $[z_\ell, z_r]$ region will be in $[z_\ell + (1/3)(z_r - z_\ell), z_r - (1/3)(z_r - z_\ell)]$, so that after processing that example, we definitely have $\mathcal{P}(\mathrm{DIS}(V)) \leq (2/3)\mathcal{P}(\mathrm{DIS}(V_0))$. Thus, upon reaching Step 6, since we have made $n/2$ label requests, a union bound implies that with probability $1-\varepsilon/2$, we have $\mathcal{P}(\mathrm{DIS}(V)) \leq \exp\{-\Omega(n/\log(n/\varepsilon))\}$, and therefore $|\mathcal{L}| \geq \exp\{\Omega(n/\log(n/\varepsilon))\}$. Thus, for some value $\Lambda_a(\varepsilon, f, \mathcal{P}) = O(\log(\Lambda_p(\varepsilon/2, f, \mathcal{P}))\log(\log(\Lambda_p(\varepsilon/2, f, \mathcal{P}))/\varepsilon))$, any $n \geq \Lambda_a(\varepsilon, f, \mathcal{P})$ gives $|\mathcal{L}| \geq \Lambda_p(\varepsilon/2, f, \mathcal{P})$ with probability at least $1-\varepsilon/2$, so that the activized algorithm achieves label complexity $\Lambda_a(\varepsilon, f, \mathcal{P}) \in \mathrm{Polylog}(\Lambda_p(\varepsilon/2, f, \mathcal{P})/\varepsilon)$.

Consider also the intervals problem (Example 2) under a uniform $\mathcal{P}$ on $[0,1]$, and suppose $f = h_{[a,b]} \in \mathbb{C}$, for $b > a$. In this case, as with any disagreement-based algorithm, until the algorithm observes the first positive example (i.e., the first $X_m \in [a,b]$), it will request the label of every example (see the reasoning above for Meta-Algorithm 0). However, at every time after observing this first positive point, say x, the region $\mathrm{DIS}(V)$ is restricted to the region between the largest negative point less than x and smallest positive point, and the region between the largest positive point and the smallest negative point larger than x. For each of these two regions, the same arguments used for the threshold problem above can be applied to show that, with probability $1 - O(\varepsilon)$, the region of disagreement is reduced by at least a constant fraction every $O(\log(n/\varepsilon))$ label requests, so that $|\mathcal{L}| \geq \exp\{\Omega(n/\log(n/\varepsilon))\}$. Thus, again the label complexity is of the form $O(\log(\Lambda_p(\varepsilon/2, f, \mathcal{P}))\log(\log(\Lambda_p(\varepsilon/2, f, \mathcal{P}))/\varepsilon))$, which is $\mathrm{Polylog}(\Lambda_p(\varepsilon/2, f, \mathcal{P})/\varepsilon)$, though this time there is a significant (additive) target-dependent term (roughly $\propto \frac{1}{b-a}\log(1/\varepsilon)$), accounting for the length of the initial phase before observing any positive examples. On the other hand, as with *any* disagreement-based algorithm, when $f = h_{[a,a]}$, because the algorithm never observes a positive example, it requests the label of every example it considers; in this case, by the same argument given for Meta-Algorithm 0, upon reaching Step 6 we have $\mathcal{P}(\mathrm{DIS}(V)) = 1$, so that $|\mathcal{L}| = O(n)$, and we observe no improvements for some passive algorithms $\mathcal{A}_p$.

A similar analysis can be performed for unions of i intervals under $\mathcal{P}$ uniform on $[0,1]$. In that case, we find that any $h_{\mathbf{z}} \in \mathbb{C}$ not representable (up to zero-probability differences) by a union of $i-1$ or fewer intervals allows for the exponential improvements of the type observed in the previous two examples; this time, the phase of exponentially decreasing $\mathcal{P}(\mathrm{DIS}(V))$ only occurs after observing an example in each of the i intervals and each of the $i-1$ negative regions separating the intervals, resulting in an additive term of roughly $\propto \frac{1}{\min_{1 \leq j < 2i} z_{j+1} - z_j}\log(i/\varepsilon)$ in the label complexity. However,

any $h_{\mathbf{z}} \in \mathbb{C}$ representable (up to zero-probability differences) by a union of $i-1$ or fewer intervals has $\mathcal{P}(\partial h_{\mathbf{z}}) = 1$, which means $|\mathcal{L}| = O(n)$, and therefore (as with any disagreement-based algorithm) Meta-Algorithm 2 will not provide improvements for some passive algorithms $\mathcal{A}_p$.

5.1.3 THE DISAGREEMENT COEFFICIENT

Toward generalizing the arguments from the above examples, consider the following definition of Hanneke (2007b).

Definition 9 *For $\varepsilon \geq 0$, the* disagreement coefficient *of a classifier f with respect to a concept space $\mathbb{C}$ under a distribution $\mathcal{P}$ is defined as*

$$\theta_f(\varepsilon) = 1 \vee \sup_{r>\varepsilon} \frac{\mathcal{P}\left(\mathrm{DIS}(\mathrm{B}(f,r))\right)}{r}.$$

Also abbreviate $\theta_f = \theta_f(0)$.

Informally, the disagreement coefficient describes the rate of collapse of the region of disagreement, relative to the distance from f. It has been useful in characterizing the label complexities achieved by several disagreement-based active learning algorithms (Hanneke, 2007b, 2011; Dasgupta, Hsu, and Monteleoni, 2007; Beygelzimer, Dasgupta, and Langford, 2009; Wang, 2009; Koltchinskii, 2010; Beygelzimer, Hsu, Langford, and Zhang, 2010), and itself has been studied and bounded for various families of learning problems (Hanneke, 2007b, 2011; Balcan, Hanneke, and Vaughan, 2010; Friedman, 2009; Beygelzimer, Dasgupta, and Langford, 2009; Mahalanabis, 2011; Wang, 2011). See the paper of Hanneke (2011) for a detailed discussion of the disagreement coefficient, including its relationships to several related quantities, as well as a variety of general properties that it satisfies. In particular, below we use the fact that, for any constant $c \in [1,\infty)$, $\theta_f(\varepsilon) \leq \theta_f(\varepsilon/c) \leq c\theta_f(\varepsilon)$. Also note that $\mathcal{P}(\partial f) = 0$ if and only if $\theta_f(\varepsilon) = o(1/\varepsilon)$. See the papers of Friedman (2009) and Mahalanabis (2011) for some general conditions on $\mathbb{C}$ and $\mathcal{P}$, under which every $f \in \mathbb{C}$ has $\theta_f < \infty$, which (as we explain below) has particularly interesting implications for active learning (Hanneke, 2007b, 2011).

To build intuition about the behavior of the disagreement coefficient, we briefly go through its calculation for our usual toy examples from above. The first two of these calculations are taken from Hanneke (2007b), and the last is from Balcan, Hanneke, and Vaughan (2010). First, consider the thresholds problem (Example 1), and for simplicity suppose the distribution $\mathcal{P}$ is uniform on $[0,1]$. In this case, as in Section 3.2, $\mathrm{B}(h_z,r) = \{h_{z'} \in \mathbb{C} : |z'-z| \leq r\}$, and $\mathrm{DIS}(\mathrm{B}(h_z,r)) \subseteq [z-r, z+r)$ with equality for sufficiently small r. Therefore, $\mathcal{P}(\mathrm{DIS}(\mathrm{B}(h_z,r))) \leq 2r$ (with equality for small r), and $\theta_{h_z}(\varepsilon) \leq 2$ with equality for sufficiently small ε. In particular, $\theta_{h_z} = 2$.

On the other hand, consider the intervals problem (Example 2), again under $\mathcal{P}$ uniform on $[0,1]$. This time, for $h_{[a,b]} \in \mathbb{C}$ with $b-a > 0$, we have for $0 < r < b-a$, $\mathrm{B}(h_{[a,b]},r) = \{h_{[a',b']} \in \mathbb{C} : |a-a'|+|b-b'| \leq r\}$, $\mathrm{DIS}(\mathrm{B}(h_{[a,b]},r)) \subseteq [a-r,a+r) \cup (b-r,b+r]$, and $\mathcal{P}(\mathrm{DIS}(\mathrm{B}(h_{[a,b]},r))) \leq 4r$ (with equality for sufficiently small r). But for $0 < b-a \leq r$, we have $\mathrm{B}(h_{[a,b]},r) \supseteq \{h_{[a',a']} : a' \in (0,1)\}$, so that $\mathrm{DIS}(\mathrm{B}(h_{[a,b]},r)) = (0,1)$ and $\mathcal{P}(\mathrm{DIS}(\mathrm{B}(h_{[a,b]},r))) = 1$. Thus, we generally have $\theta_{h_{[a,b]}}(\varepsilon) \leq \max\left\{\frac{1}{b-a}, 4\right\}$, with equality for sufficiently small ε. However, this last reasoning also indicates $\forall r > 0, \mathrm{B}(h_{[a,a]},r) \supseteq \{h_{[a',a']} : a' \in (0,1)\}$, so that $\mathrm{DIS}(\mathrm{B}(h_{[a,a]},r)) = (0,1)$ and $\mathcal{P}(\mathrm{DIS}(\mathrm{B}(h_{[a,a]},r))) = 1$; therefore, $\theta_{h_{[a,a]}}(\varepsilon) = \frac{1}{\varepsilon}$, the largest possible value for the disagreement coefficient; in particular, this also means $\theta_{h_{[a,a]}} = \infty$.

Finally, consider the unions of i intervals problem (Example 3), again under $\mathcal{P}$ uniform on $[0,1]$. First take any $h_{\mathbf{z}} \in \mathbb{C}$ such that any $h_{\mathbf{z}'} \in \mathbb{C}$ representable as a union of $i-1$ intervals has $\mathcal{P}(\{x : h_{\mathbf{z}}(x) \neq h_{\mathbf{z}'}(x)\}) > 0$. Then for $0 < r < \min_{1 \leq j < 2i} z_{j+1} - z_j$, $\mathrm{B}(h_{\mathbf{z}}, r) = \{h_{\mathbf{z}'} \in \mathbb{C} : \sum_{1 \leq j \leq 2i} |z_j - z'_j| \leq r\}$, so that $\mathcal{P}(\mathrm{DIS}(\mathrm{B}(h_{\mathbf{z}}, r))) \leq 4ir$, with equality for sufficiently small r. For $r > \min_{1 \leq j < 2i} z_{j+1} - z_j$, $\mathrm{B}(h_{\mathbf{z}}, r)$ contains a set of classifiers that flips the labels (compared to $h_{\mathbf{z}}$) in that smallest region and uses the resulting extra interval to disagree with $h_{\mathbf{z}}$ on a tiny region at an arbitrary location (either by encompassing some point with a small interval, or by splitting an interval into two intervals separated by a small gap). Thus, $\mathrm{DIS}(\mathrm{B}(h_{\mathbf{z}}, r)) = (0,1)$, and $\mathcal{P}(\mathrm{DIS}(h_{\mathbf{z}}, r)) = 1$. So in total, $\theta_{h_{\mathbf{z}}}(\varepsilon) \leq \max\left\{\frac{1}{\min_{1 \leq j < 2i} z_{j+1} - z_j}, 4i\right\}$, with equality for sufficiently small ε. On the other hand, if $h_{\mathbf{z}} \in \mathbb{C}$ can be represented by a union of $i-1$ (or fewer) intervals, then we can use the extra interval to disagree with $h_{\mathbf{z}}$ on a tiny region at an arbitrary location, while still remaining in $\mathrm{B}(h_{\mathbf{z}}, r)$, so that $\mathrm{DIS}(\mathrm{B}(h_{\mathbf{z}}, r)) = (0,1)$, $\mathcal{P}(\mathrm{DIS}(\mathrm{B}(h_{\mathbf{z}}, r))) = 1$, and $\theta_{h_{\mathbf{z}}}(\varepsilon) = \frac{1}{\varepsilon}$; in particular, in this case we have $\theta_{h_{\mathbf{z}}} = \infty$.

5.1.4 General Upper Bounds on the Label Complexity of Meta-Algorithm 2

As mentioned, the disagreement coefficient has implications for the label complexities achievable by disagreement-based active learning. The intuitive reason for this is that, as the number of label requests increases, the *diameter* of the version space shrinks at a predictable rate. The disagreement coefficient then relates the diameter of the version space to the size of its region of disagreement, which in turn describes the probability of requesting a label. Thus, the expected frequency of label requests in the data sequence decreases at a predictable rate related to the disagreement coefficient, so that $|\mathcal{L}|$ in Meta-Algorithm 2 can be lower bounded by a function of the disagreement coefficient. Specifically, the following result was essentially established by Hanneke (2011, 2007b), though actually the result below is slightly more general than the original.

Theorem 10 *For any VC class $\mathbb{C}$, and any passive learning algorithm $\mathcal{A}_p$ achieving label complexity Λ_p, the active learning algorithm obtained by applying Meta-Algorithm 2 with $\mathcal{A}_p$ as input achieves a label complexity Λ_a that, for any distribution $\mathcal{P}$ and classifier $f \in \mathbb{C}$, satisfies*

$$\Lambda_a(\varepsilon, f, \mathcal{P}) = O\left(\theta_f\left(\Lambda_p(\varepsilon/2, f, \mathcal{P})^{-1}\right) \log^2 \frac{\Lambda_p(\varepsilon/2, f, \mathcal{P})}{\varepsilon}\right).$$

The proof of Theorem 10 is similar to the original result of Hanneke (2011, 2007b), with only minor modifications to account for using $\mathcal{A}_p$ instead of returning an arbitrary element of V. The formal details are implicit in the proof of Theorem 16 below (since Meta-Algorithm 2 is essentially identical to the $k = 1$ round of Meta-Algorithm 3, defined below). We also have the following simple corollaries.

Corollary 11 *For any VC class $\mathbb{C}$, there exists a passive learning algorithm $\mathcal{A}_p$ such that, for every $f \in \mathbb{C}$ and distribution $\mathcal{P}$, the active learning algorithm obtained by applying Meta-Algorithm 2 with $\mathcal{A}_p$ as input achieves label complexity*

$$\Lambda_a(\varepsilon, f, \mathcal{P}) = O\left(\theta_f(\varepsilon) \log^2(1/\varepsilon)\right).$$

Proof The one-inclusion graph algorithm of Haussler, Littlestone, and Warmuth (1994) is a passive learning algorithm achieving label complexity $\Lambda_p(\varepsilon, f, \mathcal{P}) \leq d/\varepsilon$. Plugging this into Theorem 10, using the fact that $\theta_f(\varepsilon/2d) \leq 2d\theta_f(\varepsilon)$, and simplifying, we arrive at the result. In fact, we will see in the proof of Theorem 16 that incurring this extra constant factor of d is not actually necessary. ■

Corollary 12 *For any VC class $\mathbb{C}$ and distribution $\mathcal{P}$, if $\forall f \in \mathbb{C}$, $\theta_f < \infty$, then $(\mathbb{C}, \mathcal{P})$ is learnable at an exponential rate. If this is true for all $\mathcal{P}$, then $\mathbb{C}$ is learnable at an exponential rate.*

Proof The first claim follows directly from Corollary 11, since $\theta_f(\varepsilon) \leq \theta_f$. The second claim then follows from the fact that Meta-Algorithm 2 is adaptive to $\mathcal{P}$ (has no direct dependence on $\mathcal{P}$ except via the data). ■

Aside from the disagreement coefficient and Λ_p terms, the other constant factors hidden in the big-O in Theorem 10 are only $\mathbb{C}$-dependent (i.e., independent of f and $\mathcal{P}$). As mentioned, if we are only interested in achieving the label complexity bound of Corollary 11, we can obtain this result more directly by the simpler original algorithm of Cohn, Atlas, and Ladner (1994) via the analysis of Hanneke (2011, 2007b).

5.1.5 GENERAL LOWER BOUNDS ON THE LABEL COMPLEXITY OF META-ALGORITHM 2

It is also possible to prove a kind of *lower bound* on the label complexity of Meta-Algorithm 2 in terms of the disagreement coefficient, so that the dependence on the disagreement coefficient in Theorem 10 is unavoidable. Specifically, there are two simple observations that intuitively explain the possibility of such lower bounds. The first observation is that the expected number of label requests Meta-Algorithm 2 makes among the first $\lceil 1/\varepsilon \rceil$ unlabeled examples is at least $\theta_f(\varepsilon)/2$ (assuming it does not halt first). Similarly, the second observation is that, to arrive at a region of disagreement with expected probability mass less than $\mathcal{P}(\mathrm{DIS}(\mathrm{B}(f, \varepsilon)))/2$, Meta-Algorithm 2 requires a budget n of size at least $\theta_f(\varepsilon)/2$. These observations are formalized in Appendix C as Lemmas 47 and 48. The relevance of these observations in the context of deriving lower bounds based on the disagreement coefficient is clear. In particular, we can use the latter of these insights to arrive at the following theorem, which essentially complements Theorem 10, showing that it cannot generally be improved beyond reducing the constants and logarithmic factors, without altering the algorithm or introducing additional $\mathcal{A}_p$-dependent quantities in the label complexity bound. The proof is included in Appendix C.

Theorem 13 *For any set of classifiers $\mathbb{C}$, $f \in \mathbb{C}$, distribution $\mathcal{P}$, and nonincreasing function $\lambda : (0,1) \to \mathbb{N}$, there exists a passive learning algorithm $\mathcal{A}_p$ achieving a label complexity Λ_p with $\Lambda_p(\varepsilon, f, \mathcal{P}) = \lambda(\varepsilon)$ for all $\varepsilon > 0$, such that if Meta-Algorithm 2, with $\mathcal{A}_p$ as its argument, achieves label complexity Λ_a, then*

$$\Lambda_a(\varepsilon, f, \mathcal{P}) = \Omega\left(\theta_f\left(\Lambda_p(2\varepsilon, f, \mathcal{P})^{-1}\right)\right).$$

Recall that there are many natural learning problems for which $\theta_f = \infty$, and indeed where $\theta_f(\varepsilon) = \Omega(1/\varepsilon)$: for instance, intervals with $f = h_{[a,a]}$ under uniform $\mathcal{P}$, or unions of i intervals under uniform $\mathcal{P}$ with f representable as $i-1$ or fewer intervals. Thus, since we have just seen that the

improvements gained by disagreement-based methods are well-characterized by the disagreement coefficient, if we would like to achieve exponential improvements over passive learning for these problems, we will need to move beyond these disagreement-based methods. In the subsections that follow, we will use an alternative algorithm and analysis, and prove a general result that is always at least as good as Theorem 10 (in a big-O sense), and often significantly better (in a little-o sense). In particular, it leads to a sufficient condition for learnability at an exponential rate, strictly more general than that of Corollary 12.

5.2 An Improved Activizer

In this subsection, we define a new active learning method based on shattering, as in Meta-Algorithm 1, but which also takes fuller advantage of the sequential aspect of active learning, as in Meta-Algorithm 2. We will see that this algorithm can be analyzed in a manner analogous to the disagreement coefficient analysis of Meta-Algorithm 2, leading to a new and often dramatically-improved label complexity bound. Specifically, consider the following meta-algorithm.

Meta-Algorithm 3
Input: passive algorithm $\mathcal{A}_p$, label budget n
Output: classifier $\hat{h}$

---

0. $V \leftarrow V_0 = \mathbb{C}$, $T_0 \leftarrow \lceil 2n/3 \rceil$, $t \leftarrow 0$, $m \leftarrow 0$
1. For $k = 1, 2, \ldots, d+1$
2. Let $\mathcal{L}_k \leftarrow \{\}$, $T_k \leftarrow T_{k-1} - t$, and let $t \leftarrow 0$
3. While $t < \lceil T_k/4 \rceil$ and $m \leq k \cdot 2^n$
4. $m \leftarrow m+1$
5. If $\hat{P}_m\left(S \in \mathcal{X}^{k-1} : V \text{ shatters } S \cup \{X_m\} | V \text{ shatters } S\right) \geq 1/2$
6. Request the label Y_m of X_m, and let $\hat{y} \leftarrow Y_m$ and $t \leftarrow t+1$
7. Else let $\hat{y} \leftarrow \underset{y \in \{-1,+1\}}{\text{argmax}}\ \hat{P}_m\left(S \in \mathcal{X}^{k-1} : V[(X_m, -y)] \text{ does not shatter } S | V \text{ shatters } S\right)$
8. Let $V \leftarrow V_m = V_{m-1}[(X_m, \hat{y})]$
9. $\hat{\Delta}^{(k)} \leftarrow \hat{P}_m\left(x : \hat{P}\left(S \in \mathcal{X}^{k-1} : V \text{ shatters } S \cup \{x\} | V \text{ shatters } S\right) \geq 1/2\right)$
10. Do $\lfloor T_k/(3\hat{\Delta}^{(k)}) \rfloor$ times
11. $m \leftarrow m+1$
12. If $\hat{P}_m\left(S \in \mathcal{X}^{k-1} : V \text{ shatters } S \cup \{X_m\} | V \text{ shatters } S\right) \geq 1/2$ and $t < \lfloor 3T_k/4 \rfloor$
13. Request the label Y_m of X_m, and let $\hat{y} \leftarrow Y_m$ and $t \leftarrow t+1$
14. Else, let $\hat{y} \leftarrow \underset{y \in \{-1,+1\}}{\text{argmax}}\ \hat{P}_m\left(S \in \mathcal{X}^{k-1} : V[(X_m, -y)] \text{ does not shatter } S | V \text{ shatters } S\right)$
15. Let $\mathcal{L}_k \leftarrow \mathcal{L}_k \cup \{(X_m, \hat{y})\}$ and $V \leftarrow V_m = V_{m-1}[(X_m, \hat{y})]$
16. Return ActiveSelect$(\{\mathcal{A}_p(\mathcal{L}_1), \mathcal{A}_p(\mathcal{L}_2), \ldots, \mathcal{A}_p(\mathcal{L}_{d+1})\}, \lfloor n/3 \rfloor, \{X_{m+1}, X_{m+2}, \ldots\})$

As before, the procedure is specified in terms of estimators $\hat{P}_m$. Again, these can be defined in a variety of ways, as long as they converge (at a fast enough rate) to their respective true probabilities. For the results below, we will use the definitions given in Appendix B.1: that is, the same definitions used in Meta-Algorithm 1. Following the same argument as for Meta-Algorithm 1, one can show that Meta-Algorithm 3 is a universal activizer for $\mathbb{C}$, for any VC class $\mathbb{C}$. However, we can also obtain more detailed results in terms of a generalization of the disagreement coefficient given below.

As with Meta-Algorithm 1, this procedure has three main components: one in which we focus on reducing the version space V, one in which we focus on collecting a (conditionally) i.i.d. sample to feed into $\mathcal{A}_p$, and one in which we select from among the $d+1$ executions of $\mathcal{A}_p$. However, unlike Meta-Algorithm 1, here the first stage is also broken up based on the value of k, so that each k has its own first and second stages, rather than sharing a single first stage. Again, the choice of the number of (unlabeled) examples processed in each second stage guarantees (by a Chernoff bound) that the "$t < \lfloor 3T_k/4 \rfloor$" constraint in Step 12 is redundant. Depending on the type of label complexity result we wish to prove, this multistage architecture is sometimes avoidable. In particular, as with Corollary 11 above, to directly achieve the label complexity bound in Corollary 17 below, we can use a much simpler approach that replaces Steps 9-16, instead simply returning an arbitrary element of V upon termination.

Within each value of k, Meta-Algorithm 3 behaves analogous to Meta-Algorithm 2, requesting the label of an example only if it cannot infer the label from known information, and updating the version space V after every label request; however, unlike Meta-Algorithm 2, for values of $k > 1$, the mechanism for inferring a label is based on shatterable sets, as in Meta-Algorithm 1, and is motivated by the same argument of splitting V into subsets containing arbitrarily good classifiers (see the discussion in Section 4.1). Also unlike Meta-Algorithm 2, even the inferred labels can be used to reduce the set V (Steps 8 and 15), since they are not only correct but also potentially informative in the sense that $x \in \mathrm{DIS}(V)$. As with Meta-Algorithm 1, the key to obtaining improvement guarantees is that some value of k has $|\mathcal{L}_k| \gg n$, while maintaining that all of the labels in $\mathcal{L}_k$ are correct; ActiveSelect then guarantees the overall performance is not too much worse than that obtained by $\mathcal{A}_p(\mathcal{L}_k)$ for this value of k.

To build intuition about the behavior of Meta-Algorithm 3, let us consider our usual toy examples, again under a uniform distribution $\mathcal{P}$ on $[0,1]$; as before, for simplicity we ignore the fact that $\hat{P}_m$ is only an estimate, as well as the constraint on t in Step 12 and the effectiveness of ActiveSelect, all of which will be addressed in the general analysis. First, for the behavior of the algorithm for thresholds and nonzero-width intervals, we may simply refer to the discussion of Meta-Algorithm 2, since the $k = 1$ round of Meta-Algorithm 3 is essentially identical to Meta-Algorithm 2; in this case, we have already seen that $|\mathcal{L}_1|$ grows as $\exp\{\Omega(n/\log(n/\varepsilon))\}$ for thresholds, and does so for nonzero-width intervals after some initial period of slow growth related to the width of the target interval (i.e., the period before finding the first positive example). As with Meta-Algorithm 1, for zero-width intervals, we must look to the $k = 2$ round of Meta-Algorithm 3 to find improvements. Also as with Meta-Algorithm 1, for sufficiently large n, every X_m processed in the $k = 2$ round will have its label inferred (correctly) in Step 7 or 14 (i.e., it does not request any labels). But this means we reach Step 9 with $m = 2 \cdot 2^n + 1$; furthermore, in these circumstances the definition of $\hat{P}_m$ from Appendix B.1 guarantees (for sufficiently large n) that $\hat{\Delta}^{(2)} = 2/m$, so that $|\mathcal{L}_2| \propto n \cdot m = \Omega(n \cdot 2^n)$. Thus, we expect the label complexity gains to be *exponentially improved* compared to $\mathcal{A}_p$.

For a more involved example, consider unions of 2 intervals (Example 3), under uniform $\mathcal{P}$ on $[0,1]$, and suppose $f = h_{(a,b,a,b)}$ for $b - a > 0$; that is, the target function is representable as a single nonzero-width interval $[a,b] \subset (0,1)$. As we have seen, $\partial f = (0,1)$ in this case, so that disagreement-based methods are ineffective at improving over passive. This also means the $k = 1$ round of Meta-Algorithm 3 will not provide improvements (i.e., $|\mathcal{L}_1| = O(n)$). However, consider the $k = 2$ round. As discussed in Section 4.2, for sufficiently large n, after the first round ($k = 1$) the set V is such that any label we infer in the $k = 2$ round will be correct. Thus, it suffices to determine how large the set $\mathcal{L}_2$ becomes. By the same reasoning as in Section 4.2, for sufficiently large n, the

examples X_m whose labels are requested in Step 6 are precisely those *not* separated from both a and b by at least one of the $m-1$ examples already processed (since V is consistent with the labels of all $m-1$ of those examples). But this is the same set of points Meta-Algorithm 2 would query for the *intervals* example in Section 5.1; thus, the same argument used there implies that in this problem we have $|\mathcal{L}_2| \geq \exp\{\Omega(n/\log(n/\varepsilon))\}$ with probability $1-\varepsilon/2$, which means we should expect a label complexity of $O\left(\log(\Lambda_p(\varepsilon/2,f,\mathcal{P}))\log(\log(\Lambda_p(\varepsilon/2,f,\mathcal{P}))/\varepsilon)\right)$, where Λ_p is the label complexity of $\mathcal{A}_p$. For the case $f = h_{(a,a,a,a)}$, $k=3$ is the relevant round, and the analysis goes similarly to the $h_{[a,a]}$ scenario for intervals above. Unions of $i > 2$ intervals can be studied analogously, with the appropriate value of k to analyze being determined by the number of intervals required to represent the target up to zero-probability differences (see the discussion in Section 4.2).

5.3 Beyond the Disagreement Coefficient

In this subsection, we introduce a new quantity, a generalization of the disagreement coefficient, which we will later use to provide a general characterization of the improvements achievable by Meta-Algorithm 3, analogous to how the disagreement coefficient characterized the improvements achievable by Meta-Algorithm 2 in Theorem 10. First, let us define the following generalization of the disagreement core.

Definition 14 *For an integer $k \geq 0$, define the* k-dimensional shatter core *of a classifier f with respect to a set of classifiers $\mathcal{H}$ and probability measure P as*

$$\partial^k_{\mathcal{H},P} f = \lim_{r\to 0}\left\{S \in \mathcal{X}^k : \mathrm{B}_{\mathcal{H},P}(f,r) \text{ shatters } S\right\}.$$

As before, when $P = \mathcal{P}$, and $\mathcal{P}$ is clear from the context, we will abbreviate $\partial^k_{\mathcal{H}} f = \partial^k_{\mathcal{H},\mathcal{P}} f$, and when we also intend $\mathcal{H} = \mathbb{C}$, the *full* concept space, and $\mathbb{C}$ is clearly defined in the given context, we further abbreviate $\partial^k f = \partial^k_{\mathbb{C}} f = \partial^k_{\mathbb{C},\mathcal{P}} f$. We have the following definition, which will play a key role in the label complexity bounds below.

Definition 15 *For any concept space $\mathbb{C}$, distribution $\mathcal{P}$, and classifier f, $\forall k \in \mathbb{N}$, $\forall \varepsilon \geq 0$, define*

$$\theta^{(k)}_f(\varepsilon) = 1 \vee \sup_{r>\varepsilon} \frac{\mathcal{P}^k\left(S \in \mathcal{X}^k : \mathrm{B}(f,r) \text{ shatters } S\right)}{r}.$$

Then define

$$\tilde{d}_f = \min\left\{k \in \mathbb{N} : \mathcal{P}^k\left(\partial^k f\right) = 0\right\}$$

and

$$\tilde{\theta}_f(\varepsilon) = \theta^{(\tilde{d}_f)}_f(\varepsilon).$$

Also abbreviate $\theta^{(k)}_f = \theta^{(k)}_f(0)$ and $\tilde{\theta}_f = \tilde{\theta}_f(0)$.

We might refer to the quantity $\theta^{(k)}_f(\varepsilon)$ as the order-k (or k-dimensional) disagreement coefficient, as it represents a direct generalization of the disagreement coefficient $\theta_f(\varepsilon)$. However, rather than merely measuring the rate of collapse of the probability of *disagreement* (one-dimensional shatterability), $\theta^{(k)}_f(\varepsilon)$ measures the rate of collapse of the probability of *k-dimensional shatterability*. In particular, we have $\tilde{\theta}_f(\varepsilon) = \theta^{(\tilde{d}_f)}_f(\varepsilon) \leq \theta^{(1)}_f(\varepsilon) = \theta_f(\varepsilon)$, so that this new quantity is never

larger than the disagreement coefficient. However, unlike the disagreement coefficient, we *always* have $\tilde{\theta}_f(\varepsilon) = o(1/\varepsilon)$ for VC classes $\mathbb{C}$. In fact, we could equivalently define $\tilde{\theta}_f(\varepsilon)$ as the value of $\theta_f^{(k)}(\varepsilon)$ for the smallest k with $\theta_f^{(k)}(\varepsilon) = o(1/\varepsilon)$. Additionally, we will see below that there are many interesting cases where $\theta_f = \infty$ (even $\theta_f(\varepsilon) = \Omega(1/\varepsilon)$) but $\tilde{\theta}_f < \infty$ (e.g., intervals with a zero-width target, or unions of i intervals where the target is representable as a union of $i-1$ or fewer intervals). As was the case for θ_f, we will see that showing $\tilde{\theta}_f < \infty$ for a given learning problem has interesting implications for the label complexity of active learning (Corollary 18 below). In the process, we have also defined the quantity $\tilde{d}_f$, which may itself be of independent interest in the asymptotic analysis of learning in general. For VC classes, $\tilde{d}_f$ always exists, and in fact is at most $d+1$ (since $\mathbb{C}$ cannot shatter any $d+1$ points). When $d = \infty$, the quantity $\tilde{d}_f$ might not be defined (or defined as ∞), in which case $\tilde{\theta}_f(\varepsilon)$ is also not defined; in this work we restrict our discussion to VC classes, so that this issue never comes up; Section 7 discusses possible extensions to classes of infinite VC dimension.

We should mention that the restriction of $\tilde{\theta}_f(\varepsilon) \geq 1$ in the definition is only for convenience, as it simplifies the theorem statements and proofs below. It is not fundamental to the definition, and can be removed (at the expense of slightly more complicated theorem statements). In fact, this only makes a difference to the value of $\tilde{\theta}_f(\varepsilon)$ in some (seemingly unusual) degenerate cases. The same is true of $\theta_f(\varepsilon)$ in Definition 9.

The process of calculating $\tilde{\theta}_f(\varepsilon)$ is quite similar to that for the disagreement coefficient; we are interested in describing $\mathrm{B}(f,r)$, and specifically the variety of behaviors of elements of $\mathrm{B}(f,r)$ on points in $\mathcal{X}$, in this case with respect to shattering. To illustrate the calculation of $\tilde{\theta}_f(\varepsilon)$, consider our usual toy examples, again under $\mathcal{P}$ uniform on $[0,1]$. For the thresholds example (Example 1), we have $\tilde{d}_f = 1$, so that $\tilde{\theta}_f(\varepsilon) = \theta_f^{(1)}(\varepsilon) = \theta_f(\varepsilon)$, which we have seen is equal 2 for small ε. Similarly, for the intervals example (Example 2), any $f = h_{[a,b]} \in \mathbb{C}$ with $b-a > 0$ has $\tilde{d}_f = 1$, so that $\tilde{\theta}_f(\varepsilon) = \theta_f^{(1)}(\varepsilon) = \theta_f(\varepsilon)$, which for sufficiently small ε, is equal $\max\left\{\frac{1}{b-a}, 4\right\}$. Thus, for these two examples, $\tilde{\theta}_f(\varepsilon) = \theta_f(\varepsilon)$. However, continuing the intervals example, consider $f = h_{[a,a]} \in \mathbb{C}$. In this case, we have seen $\partial^1 f = \partial f = (0,1)$, so that $\mathcal{P}(\partial^1 f) = 1 > 0$. For any $x_1, x_2 \in (0,1)$ with $0 < |x_1 - x_2| \leq r$, $\mathrm{B}(f,r)$ can shatter (x_1,x_2), specifically using the classifiers $\{h_{[x_1,x_2]}, h_{[x_1,x_1]}, h_{[x_2,x_2]}, h_{[x_3,x_3]}\}$ for any $x_3 \in (0,1) \setminus \{x_1,x_2\}$. However, for any $x_1, x_2 \in (0,1)$ with $|x_1 - x_2| > r$, no element of $\mathrm{B}(f,r)$ classifies both as $+1$ (as it would need width greater than r, and thus would have distance from $h_{[a,a]}$ greater than r). Therefore, $\{S \in \mathcal{X}^2 : \mathrm{B}(f,r) \text{ shatters } S\} = \{(x_1,x_2) \in (0,1)^2 : 0 < |x_1 - x_2| \leq r\}$; this latter set has probability $2r(1-r) + r^2 = (2-r) \cdot r$, which shrinks to 0 as $r \to 0$. Therefore, $\tilde{d}_f = 2$. Furthermore, this shows $\tilde{\theta}_f(\varepsilon) = \theta_f^{(2)}(\varepsilon) = \sup_{r > \varepsilon}(2-r) = 2 - \varepsilon \leq 2$. Contrasting this with $\theta_f(\varepsilon) = 1/\varepsilon$, we see $\tilde{\theta}_f(\varepsilon)$ is significantly smaller than the disagreement coefficient; in particular, $\tilde{\theta}_f = 2 < \infty$, while $\theta_f = \infty$.

Consider also the space of unions of i intervals (Example 3) under $\mathcal{P}$ uniform on $[0,1]$. In this case, we have already seen that, for any $f = h_{\mathbf{z}} \in \mathbb{C}$ not representable (up to zero-probability differences) by a union of $i-1$ or fewer intervals, we have $\mathcal{P}(\partial^1 f) = \mathcal{P}(\partial f) = 0$, so that $\tilde{d}_f = 1$, and $\tilde{\theta}_f = \theta_f^{(1)} = \theta_f = \max\left\{\frac{1}{\min_{1 \leq p < 2i} z_{p+1} - z_p}, 4i\right\}$. To generalize this, suppose $f = h_{\mathbf{z}}$ is minimally representable as a union of any number $j \leq i$ of intervals of nonzero width: $[z_1, z_2] \cup [z_3, z_4] \cup \cdots \cup [z_{2j-1}, z_{2j}]$, with $0 < z_1 < z_2 < \cdots < z_{2j} < 1$. For our purposes, this is fully general, since every element of $\mathbb{C}$ has distance zero to some $h_{\mathbf{z}}$ of this type, and $\tilde{\theta}_h = \tilde{\theta}_{h'}$ for any h, h' with $\mathcal{P}(x : h(x) \neq h'(x)) = 0$. Now for any $k < i - j + 1$, and any $S = (x_1, \ldots, x_k) \in \mathcal{X}^k$ with all elements distinct, the

set $\mathrm{B}(f,r)$ can shatter S, as follows. Begin with the intervals $[z_{2p-1},z_{2p}]$ as above, and modify the classifier in the following way for each labeling of S. For any of the x_ℓ values we wish to label $+1$, if it is already in an interval $[z_{2p-1},z_{2p}]$, we do nothing; if it is not in one of the $[z_{2p-1},z_{2p}]$ intervals, we add the interval $[x_\ell,x_\ell]$ to the classifier. For any of the x_ℓ values we wish to label -1, if it is not in any interval $[z_{2p-1},z_{2p}]$, we do nothing; if it is in some interval $[z_{2p-1},z_{2p}]$, we split the interval by setting to -1 the labels in a small region $(x_\ell-\gamma,x_\ell+\gamma)$, for $\gamma < r/k$ chosen small enough so that $(x_\ell-\gamma,x_\ell+\gamma)$ does not contain any other element of S. These operations add at most k new intervals to the minimal representation of the classifier as a union of intervals, which therefore has at most $j+k\leq i$ intervals. Furthermore, the classifier disagrees with f on a set of size at most r, so that it is contained in $\mathrm{B}(f,r)$. We therefore have $\mathcal{P}^k(S\in\mathcal{X}^k:\mathrm{B}(f,r)\text{ shatters }S)=1$. However, note that for $0<r<\min\limits_{1\leq p<2j} z_{p+1}-z_p$, for any k and $S\in\mathcal{X}^k$ with all elements of $S\cup\{z_p:1\leq p\leq 2j\}$ separated by a distance greater than r, classifying the points in S opposite to f while remaining r-close to f requires us to increase to a minimum of $j+k$ intervals. Thus, for $k=i-j+1$, any $S=(x_1,\ldots,x_k)\in\mathcal{X}^k$ with $\min\limits_{y_1,y_2\in S\cup\{z_p\}_p:y_1\neq y_2}|y_1-y_2|>r$ is *not* shatterable by $\mathrm{B}(f,r)$. We therefore have $\{S\in\mathcal{X}^k:\mathrm{B}(f,r)\text{ shatters }S\}\subseteq\left\{S\in\mathcal{X}^k:\min\limits_{y_1,y_2\in S\cup\{z_p\}_p:y_1\neq y_2}|y_1-y_2|\leq r\right\}$. For $r<\min\limits_{1\leq p<2j} z_{p+1}-z_p$, we can bound the probability of this latter set by considering sampling the points x_ℓ sequentially; the probability the ℓ^{th} point is within r of one of $x_1,\ldots,x_{\ell-1},z_1,\ldots,z_{2j}$ is at most $2r(2j+\ell-1)$, so (by a union bound) the probability any of the k points $x_1,\ldots,x_k$ is within r of any other or any of $z_1,\ldots,z_{2j}$ is at most $\sum_{\ell=1}^k 2r(2j+\ell-1)=2r\left(2jk+\binom{k}{2}\right)=(1+i-j)(i+3j)r$. Since this approaches zero as $r\to 0$, we have $\tilde{d}_f=i-j+1$. Furthermore, this analysis shows $\tilde{\theta}_f=\theta_f^{(i-j+1)}\leq\max\left\{\frac{1}{\min\limits_{1\leq p<2j} z_{p+1}-z_p},(1+i-j)(i+3j)\right\}$. In fact, careful further inspection reveals that this upper bound is tight (i.e., this is the exact value of $\tilde{\theta}_f$). Recalling that $\theta_f(\varepsilon)=1/\varepsilon$ for $j<i$, we see that again $\tilde{\theta}_f(\varepsilon)$ is significantly smaller than the disagreement coefficient; in particular, $\tilde{\theta}_f<\infty$ while $\theta_f=\infty$.

Of course, for the quantity $\tilde{\theta}_f(\varepsilon)$ to be truly useful, we need to be able to describe its behavior for families of learning problems beyond these simple toy problems. Fortunately, as with the disagreement coefficient, for learning problems with simple "geometric" interpretations, one can typically bound the value of $\tilde{\theta}_f$ without too much difficulty. For instance, consider $\mathcal{X}$ the surface of a unit hypersphere in p-dimensional Euclidean space (with $p\geq 3$), with $\mathcal{P}$ uniform on $\mathcal{X}$, and $\mathbb{C}$ the space of linear separators: $\mathbb{C}=\{h_{\mathbf{w},b}(\mathbf{x})=\mathbb{1}^{\pm}_{[0,\infty)}(\mathbf{w}\cdot\mathbf{x}+b):\mathbf{w}\in\mathbb{R}^p,b\in\mathbb{R}\}$. Balcan, Hanneke, and Vaughan (2010) proved that $(\mathbb{C},\mathcal{P})$ is learnable at an exponential rate, by a specialized argument for this space. In the process, they established that for any $f\in\mathbb{C}$ with $\mathcal{P}(x:f(x)=+1)\in(0,1)$, $\theta_f<\infty$; in fact, a similar argument shows $\theta_f\leq 4\pi\sqrt{p}/\min_y\mathcal{P}(x:f(x)=y)$. Thus, in this case, $\tilde{d}_f=1$, and $\tilde{\theta}_f=\theta_f<\infty$. However, consider $f\in\mathbb{C}$ with $\mathcal{P}(x:f(x)=y)=1$ for some $y\in\{-1,+1\}$. In this case, every $h\in\mathbb{C}$ with $\mathcal{P}(x:h(x)=-y)\leq r$ has $\mathcal{P}(x:h(x)\neq f(x))\leq r$ and is therefore contained in $\mathrm{B}(f,r)$. In particular, for any $x\in\mathcal{X}$, there is such an h that disagrees with f on only a small spherical cap containing x, so that $\mathrm{DIS}(\mathrm{B}(f,r))=\mathcal{X}$ for all $r>0$. But this means $\partial f=\mathcal{X}$, which implies $\theta_f(\varepsilon)=1/\varepsilon$ and $\tilde{d}_f>1$. However, let us examine the value of $\theta_f^{(2)}$. Let $A_p=\frac{2\pi^{p/2}}{\Gamma\left(\frac{p}{2}\right)}$ denote the surface area of the unit sphere in $\mathbb{R}^p$, and let $C_p(z)=\frac{1}{2}A_pI_{2z-z^2}\left(\frac{p-1}{2},\frac{1}{2}\right)$ denote the surface area of a spherical cap of height $z\in(0,1)$ (Li, 2011), where $I_x(a,b)=\frac{\Gamma(a+b)}{\Gamma(a)\Gamma(b)}\int_0^x t^{a-1}(1-t)^{b-1}\mathrm{d}t$

is the regularized incomplete beta function. In particular, since $\sqrt{\frac{p}{12}} \leq \frac{\Gamma\left(\frac{p}{2}\right)}{\Gamma\left(\frac{p-1}{2}\right)\Gamma\left(\frac{1}{2}\right)} \leq \frac{1}{2}\sqrt{p-2}$, the probability mass $\frac{C_p(z)}{A_p} = \frac{1}{2}\frac{\Gamma\left(\frac{p}{2}\right)}{\Gamma\left(\frac{p-1}{2}\right)\Gamma\left(\frac{1}{2}\right)}\int_0^{2z-z^2} t^{\frac{p-3}{2}}(1-t)^{-\frac{1}{2}}\mathrm{d}t$ contained in a spherical cap of height z satisfies

$$\frac{C_p(z)}{A_p} \geq \frac{1}{2}\sqrt{\frac{p}{12}}\int_0^{2z-z^2} t^{\frac{p-3}{2}}\mathrm{d}t = \sqrt{\frac{p}{12}}\frac{\left(2z-z^2\right)^{\frac{p-1}{2}}}{p-1} \geq \frac{\left(2z-z^2\right)^{\frac{p-1}{2}}}{\sqrt{12p}}, \tag{2}$$

and letting $z = \min\{z, 1/2\}$, also satisfies

$$\begin{aligned}\frac{C_p(z)}{A_p} &\leq \frac{2C_p(z)}{A_p} \leq \frac{1}{2}\sqrt{p-2}\int_0^{2z-z^2} t^{\frac{p-3}{2}}(1-t)^{-\frac{1}{2}}\mathrm{d}t \\ &\leq \sqrt{p-2}\int_0^{2z-z^2} t^{\frac{p-3}{2}}\mathrm{d}t = \frac{2\sqrt{p-2}}{p-1}\left(2z-z^2\right)^{\frac{p-1}{2}} \leq \frac{\left(2z-z^2\right)^{\frac{p-1}{2}}}{\sqrt{p/6}} \leq \frac{\left(2z\right)^{\frac{p-1}{2}}}{\sqrt{p/6}}. \end{aligned} \tag{3}$$

Consider any linear separator $h \in \mathrm{B}(f,r)$ for $r < 1/2$, and let $z(h)$ denote the height of the spherical cap where $h(x) = -y$. Then (2) indicates the probability of this region is at least $\frac{\left(2z(h)-z(h)^2\right)^{\frac{p-1}{2}}}{\sqrt{12p}}$. Since $h \in \mathrm{B}(f,r)$, we know this probability mass is at most r, and we therefore have $2z(h) - z(h)^2 \leq \left(\sqrt{12p}r\right)^{\frac{2}{p-1}}$. Now for any $x_1 \in \mathcal{X}$, the set of $x_2 \in \mathcal{X}$ for which $\mathrm{B}(f,r)$ shatters (x_1,x_2) is equivalent to the set $\mathrm{DIS}(\{h \in \mathrm{B}(f,r) : h(x_1) = -y\})$. But if $h(x_1) = -y$, then x_1 is in the aforementioned spherical cap associated with h. A little trigonometry reveals that, for any spherical cap of height $z(h)$, any two points on the surface of this cap are within distance $2\sqrt{2z(h)-z(h)^2} \leq 2\left(\sqrt{12p}r\right)^{\frac{1}{p-1}}$ of each other. Thus, for any point x_2 further than $2\left(\sqrt{12p}r\right)^{\frac{1}{p-1}}$ from x_1, it must be outside the spherical cap associated with h, which means $h(x_2) = y$. But this is true for every $h \in \mathrm{B}(f,r)$ with $h(x_1) = -y$, so that $\mathrm{DIS}(\{h \in \mathrm{B}(f,r) : h(x_1) = -y\})$ is contained in the spherical cap of all elements of $\mathcal{X}$ within distance $2\left(\sqrt{12p}r\right)^{\frac{1}{p-1}}$ of x_1; a little more trigonometry reveals that the height of this spherical cap is $2\left(\sqrt{12p}r\right)^{\frac{2}{p-1}}$. Then (3) indicates the probability mass in this region is at most $\frac{2^{p-1}\sqrt{12p}r}{\sqrt{p/6}} = 2^p\sqrt{18}r$. Thus, $\mathcal{P}^2((x_1,x_2) : \mathrm{B}(f,r) \text{ shatters } (x_1,x_2)) = \int \mathcal{P}(\mathrm{DIS}(\{h \in \mathrm{B}(f,r) : h(x_1) = -y\}))\mathcal{P}(\mathrm{d}x_1) \leq 2^p\sqrt{18}r$. In particular, since this approaches zero as $r \to 0$, we have $\tilde{d}_f = 2$. This also shows that $\tilde{\theta}_f = \theta_f^{(2)} \leq 2^p\sqrt{18}$, a finite constant (albeit a rather large one). Following similar reasoning, using the opposite inequalities as appropriate, and taking r sufficiently small, one can also show $\tilde{\theta}_f \geq 2^p/(12\sqrt{2})$.

5.4 Bounds on the Label Complexity of Activized Learning

We have seen above that in the context of several examples, Meta-Algorithm 3 can offer significant advantages in label complexity over any given passive learning algorithm, and indeed also over disagreement-based active learning in many cases. In this subsection, we present a general result characterizing the magnitudes of these improvements over passive learning, in terms of $\tilde{\theta}_f(\varepsilon)$. Specifically, we have the following general theorem, along with two immediate corollaries. The proof is included in Appendix D.

Theorem 16 *For any VC class $\mathbb{C}$, and any passive learning algorithm $\mathcal{A}_p$ achieving label complexity Λ_p, the (Meta-Algorithm 3)-activized $\mathcal{A}_p$ algorithm achieves a label complexity Λ_a that, for any distribution $\mathcal{P}$ and classifier $f \in \mathbb{C}$, satisfies*

$$\Lambda_a(\varepsilon, f, \mathcal{P}) = O\left(\tilde{\theta}_f\left(\Lambda_p(\varepsilon/4, f, \mathcal{P})^{-1}\right)\log^2 \frac{\Lambda_p(\varepsilon/4, f, \mathcal{P})}{\varepsilon}\right).$$

Corollary 17 *For any VC class $\mathbb{C}$, there exists a passive learning algorithm $\mathcal{A}_p$ such that, the (Meta-Algorithm 3)-activized $\mathcal{A}_p$ algorithm achieves a label complexity Λ_a that, for any distribution $\mathcal{P}$ and classifier $f \in \mathbb{C}$, satisfies*

$$\Lambda_a(\varepsilon, f, \mathcal{P}) = O\left(\tilde{\theta}_f(\varepsilon)\log^2(1/\varepsilon)\right).$$

Proof The one-inclusion graph algorithm of Haussler, Littlestone, and Warmuth (1994) is a passive learning algorithm achieving label complexity $\Lambda_p(\varepsilon, f, \mathcal{P}) \leq d/\varepsilon$. Plugging this into Theorem 16, using the fact that $\tilde{\theta}_f(\varepsilon/4d) \leq 4d\tilde{\theta}_f(\varepsilon)$, and simplifying, we arrive at the result. In fact, we will see in the proof of Theorem 16 that incurring this extra constant factor of d is not actually necessary. ■

Corollary 18 *For any VC class $\mathbb{C}$ and distribution $\mathcal{P}$, if $\forall f \in \mathbb{C}, \tilde{\theta}_f < \infty$, then $(\mathbb{C}, \mathcal{P})$ is learnable at an exponential rate. If this is true for all $\mathcal{P}$, then $\mathbb{C}$ is learnable at an exponential rate.*

Proof The first claim follows directly from Corollary 17, since $\tilde{\theta}_f(\varepsilon) \leq \tilde{\theta}_f$. The second claim then follows from the fact that Meta-Algorithm 3 is adaptive to $\mathcal{P}$ (has no direct dependence on $\mathcal{P}$ except via the data). ■

Actually, in the proof we arrive at a somewhat more general result, in that the bound of Theorem 16 actually holds for any target function f in the "closure" of $\mathbb{C}$: that is, any f such that $\forall r > 0, \mathrm{B}(f, r) \neq \emptyset$. As previously mentioned, if our goal is only to obtain the label complexity bound of Corollary 17 by a direct approach, then we can use a simpler procedure (which cuts out Steps 9-16, instead returning an arbitrary element of V), analogous to how the analysis of the original algorithm of Cohn, Atlas, and Ladner (1994) by Hanneke (2011) obtains the label complexity bound of Corollary 11 (see also Algorithm 5 below). However, the general result of Theorem 16 is interesting in that it applies to any passive algorithm.

Inspecting the proof, we see that it is also possible to state a result that separates the probability of success from the achieved error rate, similar to the PAC model of Valiant (1984) and the analysis of active learning by Balcan, Hanneke, and Vaughan (2010). Specifically, suppose $\mathcal{A}_p$ is a passive learning algorithm such that, $\forall \varepsilon, \delta \in (0,1)$, there is a value $\lambda(\varepsilon, \delta, f, \mathcal{P}) \in \mathbb{N}$ such that $\forall n \geq \lambda(\varepsilon, \delta, f, \mathcal{P}), \mathbb{P}(\mathrm{er}(\mathcal{A}_p(\mathcal{Z}_n)) > \varepsilon) \leq \delta$. Suppose $\hat{h}_n$ is the classifier returned by the (Meta-Algorithm 3)-activized $\mathcal{A}_p$ with label budget n. Then for some $(\mathbb{C}, \mathcal{P}, f)$-dependent constant $c \in [1, \infty)$, $\forall \varepsilon, \delta \in (0, e^{-3})$, letting $\lambda = \lambda(\varepsilon/2, \delta/2, f, \mathcal{P})$,

$$\forall n \geq c\tilde{\theta}_f\left(\lambda^{-1}\right)\log^2(\lambda/\delta), \;\; \mathbb{P}\left(\mathrm{er}\left(\hat{h}_n\right) > \varepsilon\right) \leq \delta.$$

For instance, if $\mathcal{A}_p$ is an empirical risk minimization algorithm, then this is $\propto \tilde{\theta}_f(\varepsilon)\mathrm{polylog}\left(\frac{1}{\varepsilon\delta}\right)$.

5.5 Limitations and Potential Improvements

Theorem 16 and its corollaries represent significant improvements over most known results for the label complexity of active learning, and in particular over Theorem 10 and its corollaries. As for whether this also represents the best possible label complexity gains achievable by any active learning algorithm, the answer is mixed. As with most algorithms and analyses, Meta-Algorithm 3, Theorem 16, and corollaries, represent one set of solutions in a spectrum that trades strength of performance guarantees with simplicity. As such, there are several possible modifications one might make, which could potentially improve the performance guarantees. Here we sketch a few such possibilities. This subsection can be skipped by the casual reader without loss of continuity.

Even with Meta-Algorithm 3 as-is, various improvements to the bound of Theorem 16 should be possible, simply by being more careful in the analysis. For instance, as mentioned, Meta-Algorithm 3 is a *universal activizer* for any VC class $\mathbb{C}$, so in particular we know that whenever $\tilde{\theta}_f(\varepsilon) \neq o\left(1/\left(\varepsilon \log^2(1/\varepsilon)\right)\right)$, the above bound is not tight (see the work of Balcan, Hanneke, and Vaughan, 2010 for a construction leading to such $\tilde{\theta}_f(\varepsilon)$ values), and indeed any bound of the form $\tilde{\theta}_f(\varepsilon)\text{polylog}(1/\varepsilon)$ will not be tight in some cases of this type. A more refined analysis may close this gap.

Another type of potential improvement is in the constant factors. Specifically, in the case when $\tilde{\theta}_f < \infty$, if we are only interested in *asymptotic* label complexity guarantees in Corollary 17, we can replace "$\sup_{r>0}$" in Definition 15 with "$\limsup_{r\to 0}$," which can sometimes be significantly smaller and/or easier to study. This is true for the disagreement coefficient in Corollary 11 as well. Additionally, the proof (in Appendix D) reveals that there are significant $(\mathbb{C}, \mathcal{P}, f)$-dependent constant factors other than $\tilde{\theta}_f(\varepsilon)$, and it is quite likely that these can be improved by a more careful analysis of Meta-Algorithm 3 (or in some cases, possibly an improved definition of the estimators $\hat{P}_m$).

However, even with such refinements to improve the results, the approach of using $\tilde{\theta}_f$ to prove learnability at an exponential rate has limits. For instance, it is known that any *countable* $\mathbb{C}$ is learnable at an exponential rate (Balcan, Hanneke, and Vaughan, 2010). However, there are countable VC classes $\mathbb{C}$ for which $\tilde{\theta}_f = \infty$ for some elements of $\mathbb{C}$ (e.g., take the tree-paths concept space of Balcan, Hanneke, and Vaughan (2010), except instead of all infinite-depth paths from the root, take all of the finite-depth paths from the root, but keep one infinite-depth path f; for this modified space $\mathbb{C}$, which is countable, every $h \in \mathbb{C}$ has $\tilde{d}_h = 1$, and for that one infinite-depth f we have $\tilde{\theta}_f = \infty$).

Inspecting the proof reveals that it is possible to make the results slightly sharper by replacing $\tilde{\theta}_f(r_0)$ (for $r_0 = \Lambda_p(\varepsilon/4, f, \mathcal{P})^{-1}$) with a somewhat more complicated quantity: namely,

$$\min_{k<\tilde{d}_f} \sup_{r>r_0} r^{-1} \cdot \mathcal{P}\left(x \in \mathcal{X} : \mathcal{P}^k\left(S \in \mathcal{X}^k : \mathrm{B}(f,r) \text{ shatters } S \cup \{x\}\right) \geq \mathbb{P}\left(\partial^k f\right)/16\right). \tag{4}$$

This quantity can be bounded in terms of $\tilde{\theta}_f(r_0)$ via Markov's inequality, but is sometimes smaller.

As for improving Meta-Algorithm 3 itself, there are several possibilities. One immediate improvement one can make is to replace the condition in Steps 5 and 12 by $\min_{1\leq j\leq k} \hat{P}_m(S \in \mathcal{X}^{j-1} : V \text{ shatters } S \cup \{X_m\} | V \text{ shatters } S) \geq 1/2$, likewise replacing the corresponding quantity in Step 9, and substituting in Steps 7 and 14 the quantity $\max_{1\leq j\leq k} \hat{P}_m(S \in \mathcal{X}^{j-1} : V[(X_m, -y)] \text{ does not shatter } S | V \text{ shatters } S)$; in particular, the results stated for Meta-Algorithm 3 remain valid with this substitution, requiring only minor modifications to the proofs. However, it is not clear what gains in theoretical guarantees this achieves.

Additionally, there are various quantities in this procedure that can be altered almost arbitrarily, allowing room for fine-tuning. Specifically, the $2/3$ in Step 0 and $1/3$ in Step 16 can be set to arbitrary constants summing to 1. Likewise, the $1/4$ in Step 3, $1/3$ in Step 10, and $3/4$ in Step 12 can be changed to any constants in $(0,1)$, possibly depending on k, such that the sum of the first two is strictly less than the third. Also, the $1/2$ in Steps 5, 9, and 12 can be set to any constant in $(0,1)$. Furthermore, the $k \cdot 2^n$ in Step 3 only prevents infinite looping, and can be set to any function growing superlinearly in n, though to get the largest possible improvements it should at least grow exponentially in n; typically, *any* active learning algorithm capable of exponential improvements over reasonable passive learning algorithms will require access to a number of unlabeled examples exponential in n, and Meta-Algorithm 3 is no exception to this.

One major issue in the design of the procedure is an inherent trade-off between the achieved label complexity and the number of unlabeled examples used by the algorithm. This is noteworthy both because of the practical concerns of gathering such large quantities of unlabeled data, and also for computational efficiency reasons. In contrast to disagreement-based methods, the design of the estimators used in Meta-Algorithm 3 introduces such a trade-off, though in contrast to the splitting index analysis of Dasgupta (2005), the trade-off here seems only in the constant factors. The choice of these $\hat{P}_m$ estimators, both in their definition in Appendix B.1, and indeed in the very quantities they estimate, is such that we can (if desired) limit the number of unlabeled examples the main body of the algorithm uses (the actual number it needs to achieve Theorem 16 can be extracted from the proofs in Appendix D.1). However, if the number of unlabeled examples used by the algorithm is not a limiting factor, we can suggest more effective quantities. Specifically, following the original motivation for using shatterable sets, we might consider a greedily-constructed distribution over the set $\{S \in \mathcal{X}^j : V \text{ shatters } S, 1 \leq j < k, \text{ and either } j = k-1 \text{ or } \mathcal{P}(s : V \text{ shatters } S \cup \{s\}) = 0\}$. We can construct the distribution implicitly, via the following generative model. First we set $S = \{\}$. Then repeat the following. If $|S| = k-1$ or $\mathcal{P}(s \in \mathcal{X} : V \text{ shatters } S \cup \{s\}) = 0$, output S; otherwise, sample s according to the conditional distribution of X given that V shatters $S \cup \{X\}$. If we denote this distribution (over S) as $\tilde{\mathcal{P}}_k$, then replacing the estimator $\hat{P}_m\left(S \in \mathcal{X}^{k-1} : V \text{ shatters } S \cup \{X_m\} | V \text{ shatters } S\right)$ in Meta-Algorithm 3 with an appropriately constructed estimator of $\tilde{\mathcal{P}}_k(S : V \text{ shatters } S \cup \{X_m\})$ (and similarly replacing the other estimators) can lead to some improvements in the constant factors of the label complexity. However, such a modification can also dramatically increase the number of unlabeled examples required by the algorithm, since rejection-sampling to get a point from the conditional distribution of X given V shatters $S \cup \{X\}$ can be costly, as can determining whether $\mathcal{P}(s \in \mathcal{X} : V \text{ shatters } S \cup \{s\}) \approx 0$.

Unlike Meta-Algorithm 1, there remain serious efficiency concerns about Meta-Algorithm 3. If we knew the value of $\tilde{d}_f$ and $\tilde{d}_f \leq c \log_2(d)$ for some constant c, then we could potentially design an efficient version of Meta-Algorithm 3 still achieving Corollary 17. Specifically, suppose we can find a classifier in $\mathbb{C}$ consistent with any given sample, or determine that no such classifier exists, in time polynomial in the sample size (and d), and also that $\mathcal{A}_p$ efficiently returns a classifier in $\mathbb{C}$ consistent with the sample it is given. Then restricting the loop of Step 1 to those $k \leq \tilde{d}_f$ and returning $\mathcal{A}_p(\mathcal{L}_{\tilde{d}_f})$, the algorithm becomes efficient, in the sense that with high probability, its running time is $\text{poly}(d/\varepsilon)$, where ε is the error rate guarantee from inverting the label complexity at the value of n given to the algorithm. To be clear, in some cases we may obtain values $m \propto \exp\{\Omega(n)\}$, but the error rate guaranteed by $\mathcal{A}_p$ is $\tilde{O}(1/m)$ in these cases, so that we still have m polynomial in d/ε. However, in the absence of this access to $\tilde{d}_f$, the values of $k > \tilde{d}_f$ in Meta-Algorithm 3 may reach values of m much larger than $\text{poly}(d/\varepsilon)$, since the error rates obtained from these $\mathcal{A}_p(\mathcal{L}_k)$

evaluations are not guaranteed to be better than the $\mathcal{A}_p(\mathcal{L}_{\tilde{d}_f})$ evaluations, and yet we may have $|\mathcal{L}_k| \gg |\mathcal{L}_{\tilde{d}_f}|$. Thus, there remains a challenging problem of obtaining the results above (Theorem 16 and Corollary 17) via an efficient algorithm, adaptive to the value of $\tilde{d}_f$.

6. Toward Agnostic Activized Learning

The previous sections addressed learning in the *realizable* case, where there is a perfect classifier $f \in \mathbb{C}$ (i.e., $\text{er}(f) = 0$). To move beyond these scenarios, to problems in which f is not a perfect classifier (i.e., stochastic labels) or not well-approximated by $\mathbb{C}$, requires a change in technique to make the algorithms more robust to such issues. As we will see in Section 6.2, the results we can prove in this more general setting are not quite as strong as those of the previous sections, but in some ways they are more interesting, both from a practical perspective, as we expect real learning problems to involve imperfect teachers or underspecified instance representations, and also from a theoretical perspective, as the class of problems addressed is significantly more general than those encompassed by the realizable case above.

In this context, we will be largely interested in more general versions of the same types of questions as above, such as whether one can activize a given passive learning algorithm, in this case guaranteeing strictly improved label complexities for all nontrivial joint distributions over $\mathcal{X} \times \{-1,+1\}$. In Section 6.3, we present a general conjecture regarding this type of strong domination. To approach such questions, we will explore techniques for making the above algorithms robust to label noise. Specifically, we will use a natural generalization of a technique developed for noise-robust disagreement-based active learning. Toward this end, as well as for the sake of comparison, we will review the known techniques and results for disagreement-based agnostic active learning in Section 6.5. We then extend these techniques in Section 6.6 to develop a new type of agnostic active learning algorithm, based on shatterable sets, which relates to the disagreement-based agnostic active learning algorithms in a way analogous to how Meta-Algorithm 3 relates to Meta-Algorithm 2. Furthermore, we present a bound on the label complexities achieved by this method, representing a natural generalization of both Corollary 17 and the known results on disagreement-based agnostic active learning (Hanneke, 2011).

Although we present several new results, in some sense this section is less about what we know and more about what we do not yet know. As such, we will focus less on presenting a complete and elegant theory, and more on identifying potentially promising directions for exploration. In particular, Section 6.8 sketches out some interesting directions, which could potentially lead to a resolution of the aforementioned general conjecture from Section 6.3.

6.1 Definitions and Notation

In this setting, there is a joint distribution $\mathcal{P}_{XY}$ on $\mathcal{X} \times \{-1,+1\}$, with marginal distribution $\mathcal{P}$ on $\mathcal{X}$. For any classifier h, we denote by $\text{er}(h) = \mathcal{P}_{XY}((x,y) : h(x) \neq y)$. Also, denote by $\nu^*(\mathcal{P}_{XY}) = \inf_{h:\mathcal{X}\to\{-1,+1\}} \text{er}(h)$ the *Bayes error rate*, or simply ν^* when $\mathcal{P}_{XY}$ is clear from the context; also define the conditional label distribution $\eta(x;\mathcal{P}_{XY}) = \mathbb{P}(Y = +1|X = x)$, where $(X,Y) \sim \mathcal{P}_{XY}$, or $\eta(x) = \eta(x;\mathcal{P}_{XY})$ when $\mathcal{P}_{XY}$ is clear from the context. For a given concept space $\mathbb{C}$, denote $\nu(\mathbb{C};\mathcal{P}_{XY}) = \inf_{h\in\mathbb{C}} \text{er}(h)$, called the *noise rate* of $\mathbb{C}$; when $\mathbb{C}$ and/or $\mathcal{P}_{XY}$ is clear from the context, we may abbreviate $\nu = \nu(\mathbb{C}) = \nu(\mathbb{C};\mathcal{P}_{XY})$. For $\mathcal{H} \subseteq \mathbb{C}$, the *diameter* is defined as $\text{diam}(\mathcal{H};\mathcal{P}) = \sup_{h_1,h_2\in\mathcal{H}} \mathcal{P}(x : h_1(x) \neq$

$h_2(x)$). Also, for any $\varepsilon > 0$, define the ε-minimal set $\mathbb{C}(\varepsilon;\mathcal{P}_{XY}) = \{h \in \mathbb{C} : \text{er}(h) \leq \nu + \varepsilon\}$. For any set of classifiers $\mathcal{H}$, define the *closure*, denoted $\text{cl}(\mathcal{H};\mathcal{P})$, as the set of all measurable $h : \mathcal{X} \to \{-1,+1\}$ such that $\forall r > 0, \text{B}_{\mathcal{H},\mathcal{P}}(h,r) \neq \emptyset$. When $\mathcal{P}_{XY}$ is clear from the context, we will simply refer to $\mathbb{C}(\varepsilon) = \mathbb{C}(\varepsilon;\mathcal{P}_{XY})$, and when $\mathcal{P}$ is clear, we write $\text{diam}(\mathcal{H}) = \text{diam}(\mathcal{H};\mathcal{P})$ and $\text{cl}(\mathcal{H}) = \text{cl}(\mathcal{H};\mathcal{P})$.

In the noisy setting, rather than being a *perfect* classifier, we will let f denote an arbitrary element of $\text{cl}(\mathbb{C};\mathcal{P})$ with $\text{er}(f) = \nu(\mathbb{C};\mathcal{P}_{XY})$: that is, $f \in \bigcap_{\varepsilon>0} \text{cl}\left(\mathbb{C}(\varepsilon;\mathcal{P}_{XY});\mathcal{P}\right)$. Such a classifier must exist, since $\text{cl}(\mathbb{C})$ is *compact* in the pseudo-metric $\rho(h,g) = \int |h-g| \text{d}\mathcal{P} \propto \mathcal{P}(x : h(x) \neq g(x))$ (in the usual sense of the equivalence classes being compact in the ρ-induced metric). This can be seen by recalling that $\mathbb{C}$ is totally bounded (Haussler, 1992), and thus so is $\text{cl}(\mathbb{C})$, and that $\text{cl}(\mathbb{C})$ is a closed subset of $\mathcal{L}^1(\mathcal{P})$, which is complete (Dudley, 2002), so $\text{cl}(\mathbb{C})$ is also complete (Munkres, 2000). Total boundedness and completeness together imply compactness (Munkres, 2000), and this implies the existence of f since monotone sequences of nonempty closed subsets of a compact space have a nonempty limit set (Munkres, 2000).

As before, in the learning problem there is a sequence $\mathcal{Z} = \{(X_1,Y_1),(X_2,Y_2),\ldots\}$, where the (X_i,Y_i) are independent and identically distributed, and we denote by $\mathcal{Z}_m = \{(X_i,Y_i)\}_{i=1}^m$. As before, the $X_i \sim \mathcal{P}$, but rather than having each Y_i value determined as a function of X_i, instead we have each pair $(X_i,Y_i) \sim \mathcal{P}_{XY}$. The learning protocol is defined identically as above; that is, the algorithm has direct access to the X_i values, but must request the Y_i (label) values one at a time, sequentially, and can request at most n total labels, where n is a budget provided as input to the algorithm. The label complexity is now defined just as before (Definition 1), but generalized by replacing $(f,\mathcal{P})$ with the joint distribution $\mathcal{P}_{XY}$. Specifically, we have the following formal definition, which will be used throughout this section (and the corresponding appendices).

Definition 19 *An active learning algorithm $\mathcal{A}$ achieves label complexity $\Lambda(\cdot,\cdot)$ if, for any joint distribution $\mathcal{P}_{XY}$, for any $\varepsilon \in (0,1)$ and any integer $n \geq \Lambda(\varepsilon,\mathcal{P}_{XY})$, we have $\mathbb{E}\left[\text{er}\left(\mathcal{A}(n)\right)\right] \leq \varepsilon$.*

However, because there may not be any classifier with error rate less than any arbitrary $\varepsilon \in (0,1)$, our objective changes here to achieving error rate at most $\nu + \varepsilon$ for any given $\varepsilon \in (0,1)$. Thus, we are interested in the quantity $\Lambda(\nu+\varepsilon,\mathcal{P}_{XY})$, and will be particularly interested in this quantity's asymptotic dependence on ε, as $\varepsilon \to 0$. In particular, $\Lambda(\varepsilon,\mathcal{P}_{XY})$ may often be infinite for $\varepsilon < \nu$.

The label complexity for passive learning can be generalized analogously, again replacing $(f,\mathcal{P})$ by $\mathcal{P}_{XY}$ in Definition 2 as follows.

Definition 20 *A passive learning algorithm $\mathcal{A}$ achieves label complexity $\Lambda(\cdot,\cdot)$ if, for any joint distribution $\mathcal{P}_{XY}$, for any $\varepsilon \in (0,1)$ and any integer $n \geq \Lambda(\varepsilon,\mathcal{P}_{XY})$, we have $\mathbb{E}\left[\text{er}\left(\mathcal{A}\left(\mathcal{Z}_n\right)\right)\right] \leq \varepsilon$.*

For any label complexity Λ in the agnostic case, define the set $\text{Nontrivial}(\Lambda;\mathbb{C})$ as the set of all distributions $\mathcal{P}_{XY}$ on $\mathcal{X} \times \{-1,+1\}$ such that $\forall \varepsilon > 0, \Lambda(\nu+\varepsilon,\mathcal{P}_{XY}) < \infty$, and $\forall g \in \text{Polylog}(1/\varepsilon)$, $\Lambda(\nu+\varepsilon,\mathcal{P}_{XY}) = \omega(g(\varepsilon))$. In this context, we can define an *activizer* for a given passive algorithm as follows.

Definition 21 *We say an active meta-algorithm $\mathcal{A}_a$* activizes *a passive algorithm $\mathcal{A}_p$ for $\mathbb{C}$ in the agnostic case if the following holds. For any label complexity Λ_p achieved by $\mathcal{A}_p$, the active learning algorithm $\mathcal{A}_a(\mathcal{A}_p,\cdot)$ achieves a label complexity Λ_a such that, for every distribution $\mathcal{P}_{XY} \in \text{Nontrivial}(\Lambda_p;\mathbb{C})$, there exists a constant $c \in [1,\infty)$ such that*

$$\Lambda_a(\nu + c\varepsilon,\mathcal{P}_{XY}) = o\left(\Lambda_p(\nu+\varepsilon,\mathcal{P}_{XY})\right).$$

In this case, $\mathcal{A}_a$ is called an activizer *for $\mathcal{A}_p$ with respect to $\mathbb{C}$ in the agnostic case, and the active learning algorithm $\mathcal{A}_a(\mathcal{A}_p,\cdot)$ is called the $\mathcal{A}_a$-*activized *$\mathcal{A}_p$.*

6.2 A Negative Result

First, the bad news: we cannot generally hope for universal activizers for VC classes in the agnostic case. In fact, there even exist passive algorithms that *cannot be activized*, even by any specialized active learning algorithm.

Specifically, consider again Example 1, where $\mathcal{X} = [0,1]$ and $\mathbb{C}$ is the class of threshold classifiers, and let $\check{\mathcal{A}}_p$ be a passive learning algorithm that behaves as follows. Given n points $\mathcal{Z}_n = \{(X_1,Y_1),(X_2,Y_2),\ldots,(X_n,Y_n)\}$, $\check{\mathcal{A}}_p(\mathcal{Z}_n)$ returns the classifier $h_{\hat{z}} \in \mathbb{C}$, where $\hat{z} = \frac{1-2\hat{\eta}_0}{1-\hat{\eta}_0}$ and $\hat{\eta}_0 = \left(\frac{|\{i\in\{1,\ldots,n\}:X_i=0,Y_i=+1\}|}{|\{i\in\{1,\ldots,n\}:X_i=0\}|} \vee \frac{1}{8}\right) \wedge \frac{3}{8}$, taking $\hat{\eta}_0 = 1/8$ if $\{i \in \{1,\ldots,n\} : X_i = 0\} = \emptyset$. For most distributions $\mathcal{P}_{XY}$, this algorithm clearly would not behave "reasonably," in that its error rate would be quite large; in particular, in the realizable case, the algorithm's worst-case expected error rate does not converge to zero as $n \to \infty$. However, for certain distributions $\mathcal{P}_{XY}$ engineered specifically for this algorithm, it has near-optimal behavior in a strong sense. Specifically, we have the following result, the proof of which is included in Appendix E.1.

Theorem 22 *There is no activizer for $\check{\mathcal{A}}_p$ with respect to the space of threshold classifiers in the agnostic case.*

Recall that threshold classifiers were, in some sense, one of the simplest scenarios for activized learning in the realizable case. Also, since threshold-like problems are embedded in most "geometric" concept spaces, this indicates we should generally not expect there to exist activizers for arbitrary passive algorithms in the agnostic case. However, this leaves open the question of whether certain families of passive learning algorithms can be activized in the agnostic case, a topic we turn to next.

6.3 A Conjecture: Activized Empirical Risk Minimization

The counterexample above is interesting, in that it exposes the limits on generality in the agnostic setting. However, the passive algorithm that cannot be activized there is in many ways not very reasonable, in that it has suboptimal worst-case expected excess error rate (among other deficiencies). It may therefore be more interesting to ask whether some family of "reasonable" passive learning algorithms can be activized in the agnostic case. It seems that, unlike $\check{\mathcal{A}}_p$ above, certain passive learning algorithms should not have too peculiar a dependence on the label noise, so that they use Y_i to help determine $f(X_i)$ and that is all. In such cases, any Y_i value for which we can already infer the value $f(X_i)$ should simply be ignored as redundant information, so that we needn't request such values. While this discussion is admittedly vague, consider the following formal conjecture.

Recall that an *empirical risk minimization* algorithm for $\mathbb{C}$ is a type of passive learning algorithm $\mathcal{A}$, characterized by the fact that for any set $\mathcal{L} \in \bigcup_m (\mathcal{X} \times \{-1,+1\})^m$, $\mathcal{A}(\mathcal{L}) \in \operatorname*{argmin}_{h\in\mathbb{C}} \mathrm{er}_{\mathcal{L}}(h)$.

Conjecture 23 *For any VC class, there exists an active meta-algorithm $\mathcal{A}_a$ and an empirical risk minimization algorithm $\mathcal{A}_p$ for $\mathbb{C}$ such that $\mathcal{A}_a$* activizes *$\mathcal{A}_p$ for $\mathbb{C}$ in the agnostic case.*

Resolution of this conjecture would be interesting for a variety of reasons. If the conjecture is correct, it means that the vast (and growing) literature on the label complexity of empirical risk

minimization has direct implications for the potential performance of active learning under the same conditions. We might also expect activized empirical risk minimization to be quite effective in practical applications.

While this conjecture remains open at this time, the remainder of this section might be viewed as partial evidence in its favor, as we show that active learning is able to achieve improvements over the known bounds on the label complexity of passive learning in many cases.

6.4 Low Noise Conditions

In the subsections below, we will be interested in stating bounds on the label complexity of active learning, analogous to those of Theorem 10 and Theorem 16, but for learning with label noise. As in the realizable case, we should expect such bounds to have some explicit dependence on the distribution $\mathcal{P}_{XY}$. Initially, one might hope that we could state interesting label complexity bounds purely in terms of a simple quantity such as $\nu(\mathbb{C};\mathcal{P}_{XY})$. However, it is known that any label complexity bound for a nontrivial $\mathbb{C}$ (for either passive or active) depending on $\mathcal{P}_{XY}$ only via $\nu(\mathbb{C};\mathcal{P}_{XY})$ will be $\Omega\left(\varepsilon^{-2}\right)$ when $\nu(\mathbb{C};\mathcal{P}_{XY}) > 0$ (Kääriäinen, 2006). Since passive learning can achieve a $\mathcal{P}_{XY}$-independent $O\left(\varepsilon^{-2}\right)$ label complexity bound for any VC class (Alexander, 1984), we will need to discuss label complexity bounds that depend on $\mathcal{P}_{XY}$ via more detailed quantities than merely $\nu(\mathbb{C};\mathcal{P}_{XY})$ if we are to characterize the improvements of active learning over passive.

In this subsection, we review an index commonly used to describe certain properties of $\mathcal{P}_{XY}$ relative to $\mathbb{C}$: namely, the Mammen-Tsybakov margin conditions (Mammen and Tsybakov, 1999; Tsybakov, 2004; Koltchinskii, 2006). Specifically, we have the following formal condition from Koltchinskii (2006).

Condition 1 *There exist constants* $\mu, \kappa \in [1,\infty)$ *such that* $\forall \varepsilon > 0$, $\operatorname{diam}(\mathbb{C}(\varepsilon;\mathcal{P}_{XY});\mathcal{P}) \leq \mu \cdot \varepsilon^{\frac{1}{\kappa}}$.

This condition has recently been studied in depth in the passive learning literature, as it can be used to characterize scenarios where the label complexity of passive learning is between the worst-case $\Theta(1/\varepsilon^2)$ and the realizable case $\Theta(1/\varepsilon)$ (e.g., Mammen and Tsybakov, 1999; Tsybakov, 2004; Massart and Nédélec, 2006; Koltchinskii, 2006). The condition can equivalently be stated as

$$\exists \mu' \in (0,1], \kappa \in [1,\infty) \text{ s.t. } \forall h \in \mathbb{C}, \operatorname{er}(h) - \nu(\mathbb{C};\mathcal{P}_{XY}) \geq \mu' \cdot \mathcal{P}(x : h(x) \neq f(x))^{\kappa}.$$

The condition is implied by a variety of interesting special cases. For instance, it is satisfied when $\nu(\mathbb{C};\mathcal{P}_{XY}) = \nu^*(\mathcal{P}_{XY})$ and

$$\exists \mu'', \alpha \in (0,\infty) \text{ s.t. } \forall \varepsilon > 0, \mathcal{P}(x : |\eta(x;\mathcal{P}_{XY}) - 1/2| \leq \varepsilon) \leq \mu'' \cdot \varepsilon^{\alpha},$$

where κ and μ are functions of α and μ'' (Mammen and Tsybakov, 1999; Tsybakov, 2004); in particular, $\kappa = (1+\alpha)/\alpha$. This can intuitively be interpreted as saying that very noisy points are relatively rare. Special cases of this condition have also been studied in depth; for instance, *bounded noise* conditions, wherein $\nu(\mathbb{C};\mathcal{P}_{XY}) = \nu^*(\mathcal{P}_{XY})$ and $\forall x, |\eta(x;\mathcal{P}_{XY}) - 1/2| > c$ for some constant $c > 0$ (e.g., Giné and Koltchinskii, 2006; Massart and Nédélec, 2006), are a special case of Condition 1 with $\kappa = 1$.

Condition 1 can be interpreted in a variety of ways, depending on the context. For instance, in certain concept spaces with a geometric interpretation, it can often be realized as a kind of *large*

margin condition, under some condition relating the noisiness of a point's label to its distance from the optimal decision surface. That is, if the magnitude of noise $(1/2 - |\eta(x;\mathcal{P}_{XY}) - 1/2|)$ for a given point depends inversely on its distance from the optimal decision surface, so that points closer to the decision surface have noisier labels, a small value of κ in Condition 1 will occur if the distribution $\mathcal{P}$ has *low density* near the optimal decision surface (assuming $\nu(\mathbb{C};\mathcal{P}_{XY}) = \nu^*(\mathcal{P}_{XY})$) (e.g., Dekel, Gentile, and Sridharan, 2010). On the other hand, when there is *high* density near the optimal decision surface, the value of κ may be determined by how quickly $\eta(x;\mathcal{P}_{XY})$ changes as x approaches the decision boundary (Castro and Nowak, 2008). See the works of Mammen and Tsybakov (1999), Tsybakov (2004), Koltchinskii (2006), Massart and Nédélec (2006), Castro and Nowak (2008), Dekel, Gentile, and Sridharan (2010) and Bartlett, Jordan, and McAuliffe (2006) for further interpretations of Condition 1.

In the context of passive learning, one natural method to study is that of *empirical risk minimization*. Recall that a passive learning algorithm $\mathcal{A}$ is called an empirical risk minimization algorithm for $\mathbb{C}$ if it returns a classifier from $\mathbb{C}$ making the minimum number of mistakes on the labeled sample it is given as input. It is known that for any VC class $\mathbb{C}$, for any $\mathcal{P}_{XY}$ satisfying Condition 1 for finite μ and κ, every empirical risk minimization algorithm for $\mathbb{C}$ achieves a label complexity

$$\Lambda(\nu+\varepsilon,\mathcal{P}_{XY}) = O\left(\varepsilon^{\frac{1}{\kappa}-2}\cdot\log\frac{1}{\varepsilon}\right). \tag{5}$$

This follows from the works of Koltchinskii (2006) and Massart and Nédélec (2006). Furthermore, for nontrivial concept spaces, one can show that $\inf_{\Lambda}\sup_{\mathcal{P}_{XY}}\Lambda(\nu+\varepsilon;\mathcal{P}_{XY}) = \Omega\left(\varepsilon^{\frac{1}{\kappa}-2}\right)$, where the supremum ranges over all $\mathcal{P}_{XY}$ satisfying Condition 1 for the given μ and κ values, and the infimum ranges over all label complexities achievable by passive learning algorithms (Castro and Nowak, 2008; Hanneke, 2011); that is, the bound (5) cannot be significantly improved by any passive algorithm, without allowing the label complexity to have a more refined dependence on $\mathcal{P}_{XY}$ than afforded by Condition 1.

In the context of active learning, a variety of results are presently known, which in some cases show improvements over (5). Specifically, for any VC class $\mathbb{C}$ and any $\mathcal{P}_{XY}$ satisfying Condition 1, a certain noise-robust disagreement-based active learning algorithm achieves label complexity

$$\Lambda(\nu+\varepsilon,\mathcal{P}_{XY}) = O\left(\theta_f\left(\varepsilon^{\frac{1}{\kappa}}\right)\cdot\varepsilon^{\frac{2}{\kappa}-2}\cdot\log^2\frac{1}{\varepsilon}\right).$$

This general result was established by Hanneke (2011) (analyzing the algorithm of Dasgupta, Hsu, and Monteleoni, 2007), generalizing earlier $\mathbb{C}$-specific results by Castro and Nowak (2008) and Balcan, Broder, and Zhang (2007), and was later simplified and refined in some cases by Koltchinskii (2010). Comparing this to (5), when $\theta_f < \infty$ this is an improvement over passive learning by a factor of $\varepsilon^{\frac{1}{\kappa}}\cdot\log(1/\varepsilon)$. Note that this generalizes the label complexity bound of Corollary 11 above, since the realizable case entails Condition 1 with $\kappa = \mu/2 = 1$. It is also known that this type of improvement is essentially the best we can hope for when we describe $\mathcal{P}_{XY}$ purely in terms of the parameters of Condition 1. Specifically, for any nontrivial concept space $\mathbb{C}$, $\inf_{\Lambda}\sup_{\mathcal{P}_{XY}}\Lambda(\nu+\varepsilon,\mathcal{P}_{XY}) = \Omega\left(\max\left\{\varepsilon^{\frac{2}{\kappa}-2},\log\frac{1}{\varepsilon}\right\}\right)$, where the supremum ranges over all $\mathcal{P}_{XY}$ satisfying Condition 1 for the given μ and κ values, and the infimum ranges over all label complexities achievable by active learning algorithms (Hanneke, 2011; Castro and Nowak, 2008).

In the following subsection, we review the established techniques and results for disagreement-based agnostic active learning; the algorithm presented here is slightly different from that originally

analyzed by Hanneke (2011), but the label complexity bounds of Hanneke (2011) hold for this new algorithm as well. We follow this in Section 6.7 with a new agnostic active learning method that goes beyond disagreement-based learning, again generalizing the notion of disagreement to the notion of shatterability; this can be viewed as analogous to the generalization of Meta-Algorithm 2 represented by Meta-Algorithm 3, and as in that case the resulting label complexity bound replaces $\theta_f(\cdot)$ with $\tilde{\theta}_f(\cdot)$.

For both passive and active learning, results under Condition 1 are also known for more general scenarios than VC classes: namely, under entropy conditions (Mammen and Tsybakov, 1999; Tsybakov, 2004; Koltchinskii, 2006, 2011; Massart and Nédélec, 2006; Castro and Nowak, 2008; Hanneke, 2011; Koltchinskii, 2010). For a nonparametric class known as *boundary fragments*, Castro and Nowak (2008) find that active learning sometimes offers advantages over passive learning, under a special case of Condition 1. Furthermore, Hanneke (2011) shows a general result on the label complexity achievable by disagreement-based agnostic active learning, which sometimes exhibits an improved dependence on the parameters of Condition 1 under conditions on the disagreement coefficient and certain entropy conditions for $(\mathbb{C}, \mathcal{P})$ (see also Koltchinskii, 2010). These results will not play a role in the discussion below, as in the present work we restrict ourselves strictly to VC classes, leaving more general results for future investigations.

6.5 Disagreement-Based Agnostic Active Learning

Unlike the realizable case, here in the agnostic case we cannot eliminate a classifier from the version space after making merely a single mistake, since even the best classifier is potentially imperfect. Rather, we take a collection of samples with labels, and eliminate those classifiers making significantly more mistakes relative to some others in the version space. This is the basic idea underlying most of the known agnostic active learning algorithms, including those discussed in the present work. The precise meaning of "significantly more," sufficient to guarantee the version space always contains some good classifier, is typically determined by established bounds on the deviation of excess empirical error rates from excess true error rates, taken from the passive learning literature.

The following disagreement-based algorithm is slightly different from any in the existing literature, but is similar in style to a method of Beygelzimer, Dasgupta, and Langford (2009); it also bares resemblance to the algorithms of Koltchinskii (2010); Dasgupta, Hsu, and Monteleoni (2007); Balcan, Beygelzimer, and Langford (2006a, 2009). It should be considered as representative of the family of disagreement-based agnostic active learning algorithms, and all results below concerning it have analogous results for variants of these other disagreement-based methods.

Algorithm 4
Input: label budget n, confidence parameter δ
Output: classifier $\hat{h}$

---

0. $m \leftarrow 0, i \leftarrow 0, V_0 \leftarrow \mathbb{C}, \mathcal{L}_1 \leftarrow \{\}$
1. While $t < n$ and $m \leq 2^n$
2. $\quad m \leftarrow m+1$
3. $\quad$ If $X_m \in \mathrm{DIS}(V_i)$
4. $\quad\quad$ Request the label Y_m of X_m, and let $\mathcal{L}_{i+1} \leftarrow \mathcal{L}_{i+1} \cup \{(X_m, Y_m)\}$ and $t \leftarrow t+1$
5. $\quad$ Else let $\hat{y}$ be the label agreed upon by classifiers in V_i, and $\mathcal{L}_{i+1} \leftarrow \mathcal{L}_{i+1} \cup \{(X_m, \hat{y})\}$
6. $\quad$ If $m = 2^{i+1}$
7. $\quad\quad V_{i+1} \leftarrow \left\{ h \in V_i : \mathrm{er}_{\mathcal{L}_{i+1}}(h) - \min_{h' \in V_i} \mathrm{er}_{\mathcal{L}_{i+1}}(h') \leq \hat{U}_{i+1}(V_i, \delta) \right\}$
8. $\quad\quad i \leftarrow i+1$, and then $\mathcal{L}_{i+1} \leftarrow \{\}$
9. Return any $\hat{h} \in V_i$

The algorithm is specified in terms of an estimator, $\hat{U}_i$. The definition of $\hat{U}_i$ should typically be based on generalization bounds known for passive learning. Inspired by the work of Koltchinskii (2006) and applications thereof in active learning (Hanneke, 2011; Koltchinskii, 2010), we will take a definition of $\hat{U}_i$ based on a data-dependent Rademacher complexity, as follows. Let $\xi_1, \xi_2, \ldots$ denote a sequence of independent Rademacher random variables (i.e., uniform in $\{-1,+1\}$), also independent from all other random variables in the algorithm (i.e., $\mathcal{Z}$). Then for any set $\mathcal{H} \subseteq \mathbb{C}$, define

$$\hat{R}_i(\mathcal{H}) = \sup_{h_1,h_2 \in \mathcal{H}} 2^{-i} \sum_{m=2^{i-1}+1}^{2^i} \xi_m \cdot (h_1(X_m) - h_2(X_m)),$$

$$\hat{D}_i(\mathcal{H}) = \sup_{h_1,h_2 \in \mathcal{H}} 2^{-i} \sum_{m=2^{i-1}+1}^{2^i} |h_1(X_m) - h_2(X_m)|,$$

$$\hat{U}_i(\mathcal{H}, \delta) = 12\hat{R}_i(\mathcal{H}) + 34\sqrt{\hat{D}_i(\mathcal{H}) \frac{\ln(32i^2/\delta)}{2^{i-1}}} + \frac{752 \ln(32i^2/\delta)}{2^{i-1}}. \tag{6}$$

Algorithm 4 operates by repeatedly doubling the sample size $|\mathcal{L}_{i+1}|$, while only requesting the labels of the points in the region of disagreement of the version space. Each time it doubles the size of the sample $\mathcal{L}_{i+1}$, it updates the version space by eliminating any classifiers that make significantly more mistakes on $\mathcal{L}_{i+1}$ relative to others in the version space. Since the labels of the examples we infer in Step 5 are agreed upon by all elements of the version space, the *difference* of empirical error rates in Step 7 is identical to the difference of empirical error rates under the *true* labels. This allows us to use established results on deviations of excess empirical error rates from excess true error rates to judge suboptimality of some of the classifiers in the version space in Step 7, thus reducing the version space.

As with Meta-Algorithm 2, for computational feasibility, the sets V_i and $\mathrm{DIS}(V_i)$ in Algorithm 4 can be represented implicitly by a set of constraints imposed by previous rounds of the loop. Also, the update to $\mathcal{L}_{i+1}$ in Step 5 is included only to make Step 7 somewhat simpler or more intuitive; it can be be removed without altering the behavior of the algorithm, as long as we compensate by multiplying $\mathrm{er}_{\mathcal{L}_{i+1}}$ by an appropriate renormalization constant in Step 7: namely, $2^{-i}|\mathcal{L}_{i+1}|$.

We have the following result about the label complexity of Algorithm 4; it is representative of the type of theorem one can prove about disagreement-based active learning under Condition 1.

Lemma 24 *Let $\mathbb{C}$ be a VC class and suppose the joint distribution $\mathcal{P}_{XY}$ on $\mathcal{X} \times \{-1,+1\}$ satisfies Condition 1 for finite parameters μ and κ. There is a $(\mathbb{C}, \mathcal{P}_{XY})$-dependent constant $c \in (0,\infty)$ such that, for any $\varepsilon, \delta \in (0, e^{-3})$, and any integer*

$$n \geq c \cdot \theta_f\left(\varepsilon^{\frac{1}{\kappa}}\right) \cdot \varepsilon^{\frac{2}{\kappa}-2} \cdot \log^2 \frac{1}{\varepsilon\delta},$$

if $\hat{h}_n$ is the output of Algorithm 4 when run with label budget n and confidence parameter δ, then on an event of probability at least $1-\delta$,

$$\mathrm{er}\left(\hat{h}_n\right) \leq \nu + \varepsilon.$$

The proof of this result is essentially similar to the proof by Hanneke (2011), combined with some simplifying ideas from Koltchinskii (2010). It is also implicit in the proof of Lemma 26 below (by replacing "$\tilde{d}_f$" with "1" in the proof). The details are omitted. This result leads immediately to the following implication concerning the label complexity.

Theorem 25 *Let $\mathbb{C}$ be a VC class and suppose the joint distribution $\mathcal{P}_{XY}$ on $\mathcal{X} \times \{-1,+1\}$ satisfies Condition 1 for finite parameters $\mu, \kappa \in (1,\infty)$. With an appropriate (n,κ)-dependent setting of δ, Algorithm 4 achieves a label complexity Λ_a with*

$$\Lambda_a(\nu+\varepsilon, \mathcal{P}_{XY}) = O\left(\theta_f\left(\varepsilon^{\frac{1}{\kappa}}\right) \cdot \varepsilon^{\frac{2}{\kappa}-2} \cdot \log^2 \frac{1}{\varepsilon}\right).$$

Proof Taking $\delta = n^{-\frac{\kappa}{2\kappa-2}}$, the result follows by simple algebra. ■

We should note that it is possible to design a kind of wrapper to adaptively determine an appropriate δ value, so that the algorithm achieves the label complexity guarantee of Theorem 25 without requiring any explicit dependence on the noise parameter κ. Specifically, one can use an idea similar to the model selection procedure of Hanneke (2011) for this purpose. However, as our focus in this work is on moving beyond disagreement-based active learning, we do not include the details of such a procedure here.

Note that Theorem 25 represents an improvement over the known results for passive learning (namely, (5)) whenever $\theta_f(\varepsilon)$ is small, and in particular this gap can be large when $\theta_f < \infty$. The results of Lemma 24 and Theorem 25 represent the state-of-the-art (up to logarithmic factors) in our understanding of the label complexity of agnostic active learning for VC classes. Thus, any significant improvement over these would advance our understanding of the fundamental capabilities of active learning in the presence of label noise. Next, we provide such an improvement.

6.6 A New Type of Agnostic Active Learning Algorithm Based on Shatterable Sets

Algorithm 4 and Theorem 25 represent natural extensions of Meta-Algorithm 2 and Theorem 10 to the agnostic setting. As such, they not only benefit from the advantages of those methods (small $\theta_f(\varepsilon)$ implies improved label complexity), but also suffer the same disadvantages ($\mathcal{P}(\partial f) > 0$ implies no strong improvements over passive). It is therefore natural to investigate whether the improvements offered by Meta-Algorithm 3 and the corresponding Theorem 16 can be extended to the

agnostic setting in a similar way. In particular, as was possible for Theorem 16 with respect to Theorem 10, we might wonder whether it is possible to replace $\theta_f\left(\varepsilon^{\frac{1}{\kappa}}\right)$ in Theorem 25 with $\tilde{\theta}_f\left(\varepsilon^{\frac{1}{\kappa}}\right)$ by a modification of Algorithm 4 analogous to the modification of Meta-Algorithm 2 embodied in Meta-Algorithm 3. As we have seen, $\tilde{\theta}_f\left(\varepsilon^{\frac{1}{\kappa}}\right)$ is often significantly smaller in its asymptotic dependence on ε, compared to $\theta_f\left(\varepsilon^{\frac{1}{\kappa}}\right)$, in many cases even bounded by a finite constant when $\theta_f\left(\varepsilon^{\frac{1}{\kappa}}\right)$ is not. This would therefore represent a significant improvement over the known results for active learning under Condition 1. Toward this end, consider the following algorithm.

Algorithm 5
Input: label budget n, confidence parameter δ
Output: classifier $\hat{h}$

---

0. $m \leftarrow 0, i_0 \leftarrow 0, V_0 \leftarrow \mathbb{C}$
1. For $k = 1, 2, \ldots, d+1$
2. $\quad t \leftarrow 0, i_k \leftarrow i_{k-1}, m \leftarrow 2^{i_k}, V_{i_k+1} \leftarrow V_{i_k}, \mathcal{L}_{i_k+1} \leftarrow \{\}$
3. $\quad$ While $t < \left\lfloor 2^{-k} n \right\rfloor$ and $m \leq k \cdot 2^n$
4. $\quad\quad m \leftarrow m+1$
5. $\quad\quad$ If $\hat{P}_{4m}\left(S \in \mathcal{X}^{k-1} : V_{i_k+1} \text{ shatters } S \cup \{X_m\} | V_{i_k+1} \text{ shatters } S\right) \geq 1/2$
6. $\quad\quad\quad$ Request the label Y_m of X_m, and let $\mathcal{L}_{i_k+1} \leftarrow \mathcal{L}_{i_k+1} \cup \{(X_m, Y_m)\}$ and $t \leftarrow t+1$
7. $\quad\quad$ Else $\hat{y} \leftarrow \underset{y \in \{-1,+1\}}{\text{argmax}}\, \hat{P}_{4m}\left(S \in \mathcal{X}^{k-1} : V_{i_k+1}[(X_m, -y)] \text{ does not shatter } S | V_{i_k+1} \text{ shatters } S\right)$
8. $\quad\quad\quad \mathcal{L}_{i_k+1} \leftarrow \mathcal{L}_{i_k+1} \cup \{(X_m, \hat{y})\}$ and $V_{i_k+1} \leftarrow V_{i_k+1}[(X_m, \hat{y})]$
9. $\quad\quad$ If $m = 2^{i_k+1}$
10. $\quad\quad\quad V_{i_k+1} \leftarrow \left\{ h \in V_{i_k+1} : \text{er}_{\mathcal{L}_{i_k+1}}(h) - \min_{h' \in V_{i_k+1}} \text{er}_{\mathcal{L}_{i_k+1}}(h') \leq \hat{U}_{i_k+1}\left(V_{i_k}, \delta\right) \right\}$
11. $\quad\quad\quad i_k \leftarrow i_k + 1$, then $V_{i_k+1} \leftarrow V_{i_k}$, and $\mathcal{L}_{i_k+1} \leftarrow \{\}$
12. Return any $\hat{h} \in V_{i_{d+1}+1}$

For the argmax in Step 7, we break ties in favor of a $\hat{y}$ value with $V_{i_k+1}[(X_m, \hat{y})] \neq \emptyset$ to maintain the invariant that $V_{i_k+1} \neq \emptyset$ (see the proof of Lemma 59); when both y values satisfy this, we may break ties arbitrarily. The procedure is specified in terms of several estimators. The $\hat{P}_{4m}$ estimators, as usual, are defined in Appendix B.1. For $\hat{U}_i$, we again use the definition (6) above, based on a data-dependent Rademacher complexity.

Algorithm 5 is largely based on the same principles as Algorithm 4, combined with Meta-Algorithm 3. As in Algorithm 4, the algorithm proceeds by repeatedly doubling the size of a labeled sample $\mathcal{L}_{i+1}$, while only requesting a subset of the labels in $\mathcal{L}_{i+1}$, inferring the others. As before, it updates the version space every time it doubles the size of the sample $\mathcal{L}_{i+1}$, and the update eliminates classifiers from the version space that make significantly more mistakes on $\mathcal{L}_{i+1}$ compared to others in the version space. In Algorithm 4, this is guaranteed to be effective, since the classifiers in the version space agree on all of the inferred labels, so that the differences of empirical error rates remain equal to the *true* differences of empirical error rates (i.e., under the true Y_m labels for all elements of $\mathcal{L}_{i+1}$); thus, the established results from the passive learning literature bounding the deviations of excess empirical error rates from excess true error rates can be applied, showing that this does not eliminate the best classifiers. In Algorithm 5, the situation is somewhat more subtle, but the principle remains the same. In this case, we *enforce* that the classifiers in the version space

agree on the inferred labels in $\mathcal{L}_{i+1}$ by explicitly removing the disagreeing classifiers in Step 8. Thus, as long as Step 8 does not eliminate all of the good classifiers, then neither will Step 10. To argue that Step 8 does not eliminate all good classifiers, we appeal to the same reasoning as for Meta-Algorithm 1 and Meta-Algorithm 3. That is, for $k \leq \tilde{d}_f$ and sufficiently large n, as long as there exist good classifiers in the version space, the labels $\hat{y}$ inferred in Step 7 will agree with some good classifiers, and thus Step 8 will not eliminate all good classifiers. However, for $k > \tilde{d}_f$, the labels $\hat{y}$ in Step 7 have no such guarantees, so that we are only guaranteed that *some* classifier in the version space is not eliminated. Thus, determining guarantees on the error rate of this algorithm hinges on bounding the worst excess error rate among all classifiers in the version space at the conclusion of the $k = \tilde{d}_f$ round. This is essentially determined by the size of $\mathcal{L}_{i_k}$ at the conclusion of that round, which itself is largely determined by how frequently the algorithm requests labels during this $k = \tilde{d}_f$ round. Thus, once again the analysis rests on bounding the rate at which the frequency of label requests shrinks in the $k = \tilde{d}_f$ round, which determines the rate of growth of $|\mathcal{L}_{i_k}|$, and thus the final guarantee on the excess error rate.

As before, for computational feasibility, we can maintain the sets V_i implicitly as a set of constraints imposed by the previous updates, so that we may perform the various calculations required for the estimators $\hat{P}$ as constrained optimizations. Also, the update to $\mathcal{L}_{i_k+1}$ in Step 8 is merely included to make the algorithm statement and the proofs somewhat more elegant; it can be omitted, as long as we compensate with an appropriate renormalization of the $\mathrm{er}_{\mathcal{L}_{i_k+1}}$ values in Step 10 (i.e., multiplying by $2^{-i_k}|\mathcal{L}_{i_k+1}|$). Additionally, the same potential improvements we proposed in Section 5.5 for Meta-Algorithm 3 can be made to Algorithm 5 as well, again with only minor modifications to the proofs. We should note that Algorithm 5 is certainly not the only reasonable way to extend Meta-Algorithm 3 to the agnostic setting. For instance, another natural extension of Meta-Algorithm 1 to the agnostic setting, based on a completely different idea, appears in the author's doctoral dissertation (Hanneke, 2009b); that method can be improved in a natural way to take advantage of the sequential aspect of active learning, yielding an agnostic extension of Meta-Algorithm 3 differing from Algorithm 5 in several interesting ways (see the discussion in Section 6.8 below).

In the next subsection, we will see that the label complexities achieved by Algorithm 5 are often significantly better than the known results for passive learning. In fact, they are often significantly better than the presently-known results for any *active* learning algorithms in the published literature.

6.7 Improved Label Complexity Bounds for Active Learning with Noise

Under Condition 1, we can extend Lemma 24 and Theorem 25 in an analogous way to how Theorem 16 extends Theorem 10. Specifically, we have the following result, the proof of which is included in Appendix E.2.

Lemma 26 *Let $\mathbb{C}$ be a VC class and suppose the joint distribution $\mathcal{P}_{XY}$ on $\mathcal{X} \times \{-1,+1\}$ satisfies Condition 1 for finite parameters μ and κ. There is a $(\mathbb{C}, \mathcal{P}_{XY})$-dependent constant $c \in (0,\infty)$ such that, for any $\varepsilon, \delta \in \left(0, e^{-3}\right)$, and any integer*

$$n \geq c \cdot \tilde{\theta}_f\left(\varepsilon^{\frac{1}{\kappa}}\right) \cdot \varepsilon^{\frac{2}{\kappa}-2} \cdot \log^2 \frac{1}{\varepsilon\delta},$$

if $\hat{h}_n$ is the output of Algorithm 5 when run with label budget n and confidence parameter δ, then on an event of probability at least $1-\delta$,

$$\mathrm{er}\left(\hat{h}_n\right) \leq \nu + \varepsilon.$$

This has the following implication for the label complexity of Algorithm 5.

Theorem 27 *Let $\mathbb{C}$ be a VC class and suppose the joint distribution $\mathcal{P}_{XY}$ on $\mathcal{X} \times \{-1,+1\}$ satisfies Condition 1 for finite parameters $\mu, \kappa \in (1,\infty)$. With an appropriate (n,κ)-dependent setting of δ, Algorithm 5 achieves a label complexity Λ_a with*

$$\Lambda_a(\nu+\varepsilon, \mathcal{P}_{XY}) = O\left(\tilde{\theta}_f\left(\varepsilon^{\frac{1}{\kappa}}\right) \cdot \varepsilon^{\frac{2}{\kappa}-2} \cdot \log^2 \frac{1}{\varepsilon}\right).$$

Proof Taking $\delta = n^{-\frac{\kappa}{2\kappa-2}}$, the result follows by simple algebra. ■

Theorem 27 represents an interesting generalization beyond the realizable case, and beyond the disagreement coefficient analysis. Note that if $\tilde{\theta}_f(\varepsilon) = o\left(\varepsilon^{-1}\log^{-2}(1/\varepsilon)\right)$, Theorem 27 represents an improvement over the known results for passive learning (Massart and Nédélec, 2006). As we always have $\tilde{\theta}_f(\varepsilon) = o(1/\varepsilon)$, we should typically expect such improvements for all but the most extreme learning problems. Recall that $\theta_f(\varepsilon)$ is often *not* $o(1/\varepsilon)$, so that Theorem 27 is often a much stronger statement than Theorem 25. In particular, this is a significant improvement over the known results for passive learning whenever $\tilde{\theta}_f < \infty$, and an equally significant improvement over Theorem 25 whenever $\tilde{\theta}_f < \infty$ but $\theta_f(\varepsilon) = \Omega(1/\varepsilon)$ (see above for examples of this). However, note that unlike Meta-Algorithm 3, Algorithm 5 is *not* an activizer. Indeed, it is not clear (to the author) how to modify the algorithm to make it a universal activizer for $\mathbb{C}$ (even for the realizable case), while maintaining the guarantees of Theorem 27.

As with Theorem 16 and Corollary 17, Algorithm 5 and Theorem 27 can potentially be improved in a variety of ways, as outlined in Section 5.5. In particular, Theorem 27 can be made slightly sharper in some cases by replacing $\tilde{\theta}_f\left(\varepsilon^{\frac{1}{\kappa}}\right)$ with the sometimes-smaller (though more complicated) quantity (4) (with $r_0 = \varepsilon^{\frac{1}{\kappa}}$).

6.8 Beyond Condition 1

While Theorem 27 represents an improvement over the known results for agnostic active learning, Condition 1 is not fully general, and disallows many important and interesting scenarios. In particular, one key property of Condition 1, heavily exploited in the label complexity proofs for both passive learning and disagreement-based active learning, is that it implies $\text{diam}(\mathbb{C}(\varepsilon)) \to 0$ as $\varepsilon \to 0$. In scenarios where this shrinking diameter condition is not satisfied, the existing proofs of (5) for passive learning break down, and furthermore, the disagreement-based algorithms themselves cease to give significant improvements over passive learning, for essentially the same reasons leading to the "only if" part of Theorem 5 (i.e., the sampling region never focuses beyond some nonzero-probability region). Even more alarming (at first glance) is the fact that this same problem can sometimes be observed for the $k = \tilde{d}_f$ round of Algorithm 5; that is, $\mathcal{P}\left(x : \mathcal{P}^{\tilde{d}_f-1}(S \in \mathcal{X}^{\tilde{d}_f-1} : V_{i_{\tilde{d}_f}+1} \text{ shatters } S \cup \{x\} | V_{i_{\tilde{d}_f}+1} \text{ shatters } S) \geq 1/2\right)$ is no longer guaranteed to approach 0 as the budget n increases (as it *does* when $\text{diam}(\mathbb{C}(\varepsilon)) \to 0$). Thus, if we wish to approach an understanding of improvements achievable by active learning in general, we must come to terms with scenarios where $\text{diam}(\mathbb{C}(\varepsilon))$ does not shrink to zero.

Interestingly, it seems that $\text{diam}(\mathbb{C}(\varepsilon)) \not\to 0$ might not be a problem for some algorithms based on shatterable sets, such as Algorithm 5. In particular, Algorithm 5 appears to continue exhibiting

reasonable behavior in such scenarios. That is, even if there is a nonshrinking probability that the query condition in Step 5 is satisfied for $k = \tilde{d}_f$, on any given sequence $\mathcal{Z}$ there must be *some* smallest value of k for which this probability *does* shrink as $n \to \infty$. For this value of k, we should expect to observe good behavior from the algorithm, in that (for sufficiently large n) the inferred labels in Step 7 will tend to agree with *some* classifier $f \in \text{cl}(\mathbb{C})$ with $\text{er}(f) = \nu(\mathbb{C}; \mathcal{P}_{XY})$. Thus, the algorithm addresses the problem of multiple optimal classifiers by effectively *selecting* one of the optimal classifiers.

To illustrate this phenomenon, consider learning with respect to the space of threshold classifiers (Example 1) with $\mathcal{P}$ uniform in $[0,1]$, and let $(X,Y) \sim \mathcal{P}_{XY}$ satisfy $\mathbb{P}(Y = +1|X) = 0$ for $X < 1/3$, $\mathbb{P}(Y = +1|X) = 1/2$ for $1/3 \leq X < 2/3$, and $\mathbb{P}(Y = +1|X) = 1$ for $2/3 \leq X$. As we know from above, $\tilde{d}_f = 1$ here. However, in this scenario we have $\text{DIS}(\mathbb{C}(\varepsilon)) \to [1/3, 2/3]$ as $\varepsilon \to 0$. Thus, Algorithm 4 never focuses its queries beyond a constant fraction of $\mathcal{X}$, and therefore cannot improve over certain passive learning algorithms in terms of the asymptotic dependence of its label complexity on ε (assuming a worst-case choice of $\hat{h}$ in Step 9). However, for $k = 2$ in Algorithm 5, every X_m will be assigned a label $\hat{y}$ in Step 7 (since no two points are shattered); furthermore, for sufficiently large n we have (with high probability) $\text{DIS}(V_{i_1})$ not too much larger than $[1/3, 2/3]$, so that most points in $\text{DIS}(V_{i_1})$ can be labeled either $+1$ or -1 by some optimal classifier. For us, this has two implications. First, one can show that with very high probability, the $S \in [1/3, 2/3]$[1] will dominate the votes for $\hat{y}$ in Step 7 (for all m processed while $k = 2$), so that the $\hat{y}$ inferred for any $X_m \notin [1/3, 2/3]$ will agree with all of the optimal classifiers. Second, the inferred labels $\hat{y}$ for $X_m \in [1/3, 2/3]$ will definitely agree with *some* optimal classifier. Since we also impose the $h(X_m) = \hat{y}$ constraint for V_{i_2+1} in Step 8, the inferred $\hat{y}$ labels must all be consistent with the *same* optimal classifier, so that V_{i_2+1} will quickly converge to within a small neighborhood around that classifier, without any further label requests. Note, however, that the particular optimal classifier the algorithm converges to will be a random variable, determined by the particular sequence of data points processed by the algorithm; thus, it cannot be determined a priori, which significantly complicates any general attempt to analyze the label complexity achieved by the algorithm for arbitrary $\mathbb{C}$ and $\mathcal{P}_{XY}$. In particular, for some $\mathbb{C}$ and $\mathcal{P}_{XY}$, even this minimal k for which convergence occurs may be a nondeterministic random variable. At this time, it is not entirely clear how general this phenomenon is (i.e., Algorithm 5 providing improvements over certain passive algorithms even for distributions with $\text{diam}(\mathbb{C}(\varepsilon)) \nrightarrow 0$), nor how to characterize the label complexity achieved by Algorithm 5 in general settings where $\text{diam}(\mathbb{C}(\varepsilon)) \nrightarrow 0$.

However, as mentioned earlier, there are other natural ways to generalize Meta-Algorithm 3 to handle noise, some of which have more predictable behavior. In particular, the original thesis work of Hanneke (2009b) explores a technique for active learning, which unlike Algorithm 5, only uses the *requested* labels, not the inferred labels, and as a consequence never eliminates any optimal classifier from V. Because of this fact, the sampling region for each k converges to a predictable limiting region, so that we have an accurate *a priori* characterization of the algorithm's behavior. However, it is not immediately clear (to the author) whether this alternative technique might lead to a method achieving results similar to Theorem 27.

To get a better understanding of the scenario where $\text{diam}(\mathbb{C}(\varepsilon)) \nrightarrow 0$, it will be helpful to partition the distributions into two distinct categories, which we will refer to as the *benign noise* case and the *misspecified model* case. The $\mathcal{P}_{XY}$ in the benign noise case are characterized by the property that $\nu(\mathbb{C}; \mathcal{P}_{XY}) = \nu^*(\mathcal{P}_{XY})$; this is in some ways similar to the realizable case, in that $\mathbb{C}$ can approximate an optimal classifier, except that the labels are stochastic. In the benign noise case, the only

reason $\mathrm{diam}(\mathbb{C}(\varepsilon))$ would not shrink to zero is if there is a nonzero probability set of points x with $\eta(x) = 1/2$; that is, there are at least two classifiers achieving the Bayes error rate, and they are at nonzero distance from each other, which must mean they disagree on some points that have equal probability of either label occurring. In contrast, the misspecified model case is characterized by $\nu(\mathbb{C}; P_{XY}) > \nu^*(P_{XY})$. In this case, if the diameter does not shrink, it is because of the existence of two classifiers $h_1, h_2 \in \mathrm{cl}(\mathbb{C})$ achieving error rate $\nu(\mathbb{C}; P_{XY})$, with $P(x : h_1(x) \neq h_2(x)) > 0$. However, unlike above, since they do not achieve the Bayes error rate, it is possible that a significant fraction of the set of points they disagree on may have $\eta(x) \neq 1/2$.

Intuitively, the benign noise case is relatively easier for active learning, since the noisy points that prevent $\mathrm{diam}(\mathbb{C}(\varepsilon))$ from shrinking can essentially be assigned arbitrary labels, as in the thresholds example above. For instance, as in Algorithm 5, we could assign a label to points in this region and discard any classifiers inconsistent with the label, confident that we have kept at least one optimal classifier. Another possibility is simply to ignore the points in this region, since in the end they are inconsequential for the excess error rate of the classifier we return; in some sense, this is the strategy taken by the method of Hanneke (2009b).

In contrast, the misspecified model case intuitively makes the active learning problem more difficult. For instance, if h_1 and h_2 in $\mathrm{cl}(\mathbb{C})$ both have error rate $\nu(\mathbb{C}; P_{XY})$, the original method of Hanneke (2009b) has the possibility of inferring the label $h_2(x)$ for some point x when in fact $h_1(x)$ is better for that particular x, and vice versa for the points x where $h_2(x)$ would be better, thus getting the worst of both and potentially doubling the error rate in the process. Algorithm 5 may fare better in this case, since imposing the inferred label $\hat{y}$ as a constraint in Step 8 effectively *selects* one of h_1 or h_2, and discards the other one. As before, whether Algorithm 5 selects h_1 or h_2 will generally depend on the particular data sequence $\mathcal{Z}$, which therefore makes any a priori analysis of the label complexity more challenging.

Interestingly, it turns out that, for the purpose of exploring Conjecture 23, we can circumvent all of these issues by noting that there is a trivial solution to the misspecified model case. Specifically, since in our present context we are only interested in the label complexity for achieving error rate better than $\nu + \varepsilon$, we can simply turn to any algorithm that asymptotically achieves an error rate strictly better than ν (e.g., Devroye et al., 1996), in which case the algorithm should require only a finite constant number of labels to achieve an expected error rate better than ν. To make the algorithm effective for the general case, we simply split our budget in three: one part for an active learning algorithm, such as Algorithm 5, for the benign noise case, one part for the method above handling the misspecified model case, and one part to select among their outputs. The full details of such a procedure are specified in Appendix E.3, along with a proof of its performance guarantees, which are summarized as follows.

Theorem 28 *Fix any concept space $\mathbb{C}$. Suppose there exists an active learning algorithm $\mathcal{A}_a$ achieving a label complexity Λ_a. Then there exists an active learning algorithm $\mathcal{A}'_a$ achieving a label complexity Λ'_a such that, for any distribution P_{XY} on $\mathcal{X} \times \{-1, +1\}$, there exists a function $\lambda(\varepsilon) \in \mathrm{Polylog}(1/\varepsilon)$ such that*

$$\Lambda'_a(\nu + \varepsilon, P_{XY}) \leq \begin{cases} \max\{2\Lambda_a(\nu + \varepsilon/2, P_{XY}), \lambda(\varepsilon)\}, & \text{in the benign noise case} \\ \lambda(\varepsilon), & \text{in the misspecified model case} \end{cases}.$$

The main point of Theorem 28 is that, for our purposes, we can safely ignore the misspecified model case (as its solution is a trivial extension), and focus entirely on the performance of algorithms for the benign noise case. In particular, for any label complexity Λ_p, every $\mathcal{P}_{XY} \in \text{Nontrivial}(\Lambda_p; \mathbb{C})$ in the misspecified model case has $\Lambda'_a(\nu + \varepsilon, \mathcal{P}_{XY}) = o(\Lambda_p(\nu + \varepsilon, \mathcal{P}_{XY}))$, for Λ'_a as in Theorem 28. Thus, if there exists an active meta-algorithm achieving the strong improvement guarantees of an activizer for some passive learning algorithm $\mathcal{A}_p$ (Definition 21) for all distributions $\mathcal{P}_{XY}$ in the benign noise case, then there exists an activizer for $\mathcal{A}_p$ with respect to $\mathbb{C}$ in the agnostic case.

7. Open Problems

In some sense, this work raises more questions than it answers. Here, we list several problems that remain open at this time. Resolving any of these problems would make a significant contribution to our understanding of the fundamental capabilities of active learning.

- We have established the existence of universal activizers for VC classes in the realizable case. However, we have not made any serious attempt to characterize the properties that such activizers can possess. In particular, as mentioned, it would be interesting to know whether activizers exist that *preserve* certain favorable properties of the given passive learning algorithm. For instance, we know that some passive learning algorithms (say, for linear separators) achieve a label complexity that is independent of the dimensionality of the space $\mathcal{X}$, under a large margin condition on f and $\mathcal{P}$ (Balcan, Blum, and Vempala, 2006b). Is there an activizer for such algorithms that preserves this large-margin-based dimension-independence in the label complexity? Similarly, there are passive algorithms whose label complexity has a weak dependence on dimensionality, due to sparsity considerations (Bunea, Tsybakov, and Wegkamp, 2009; Wang and Shen, 2007). Is there an activizer for these algorithms that preserves this sparsity-based weak dependence on dimension? Is there an activizer that preserves adaptiveness to the dimension of the manifold to which $\mathcal{P}$ is restricted? What about an activizer that is *sparsistent* (Rocha, Wang, and Yu, 2009), given any sparsistent passive learning algorithm as input? Is there an activizer that preserves admissibility, in that given any admissible passive learning algorithm, the activized algorithm is an admissible active learning algorithm? Is there an activizer that, given any minimax optimal passive learning algorithm as input, produces a minimax optimal active learning algorithm? What about preserving other notions of optimality, or other properties?

- There may be some waste in the above activizers, since the label requests used in their initial phase (reducing the version space) are not used by the passive algorithm to produce the final classifier. This guarantees the examples fed into the passive algorithm are conditionally independent given the number of examples. Intuitively, this seems necessary for the general results, since any dependence among the examples fed to the passive algorithm could influence its label complexity. However, it is not clear (to the author) how dramatic this effect can be, nor whether a simpler strategy (e.g., slightly randomizing the budget of label requests) might yield a similar effect while allowing a single-stage approach where all labels are used in the passive algorithm. It seems intuitively clear that some special types of passive algorithms should be able to use the full set of examples, from both phases, while still maintaining the strict improvements guaranteed in the main theorems above. What general properties must such passive algorithms possess?

- As previously mentioned, the vast majority of empirically-tested *heuristic* active learning algorithms in the published literature are designed in a reduction style, using a well-known passive learning algorithm as a subroutine, constructing sets of labeled examples and feeding them into the passive learning algorithm at various points in the execution of the active learning algorithm (e.g., Abe and Mamitsuka, 1998; McCallum and Nigam, 1998; Schohn and Cohn, 2000; Campbell, Cristianini, and Smola, 2000; Tong and Koller, 2001; Roy and McCallum, 2001; Muslea, Minton, and Knoblock, 2002; Lindenbaum, Markovitch, and Rusakov, 2004; Mitra, Murthy, and Pal, 2004; Roth and Small, 2006; Schein and Ungar, 2007; Har-Peled, Roth, and Zimak, 2007; Beygelzimer, Dasgupta, and Langford, 2009). However, rather than including some examples whose labels are requested and other examples whose labels are *inferred* in the sets of labeled examples given to the passive learning algorithm (as in our rigorous methods above), these heuristic methods typically only input to the passive algorithm the examples whose labels were *requested*. We should expect that meta-algorithms of this type could not be *universal* activizers for $\mathbb{C}$, but perhaps there do exist meta-algorithms of this type that are activizers for every passive learning algorithm of some special type. What are some general conditions on the passive learning algorithm so that some meta-algorithm of this type (i.e., feeding in only the *requested* labels) can activize every passive learning algorithm satisfying those conditions?

- As discussed earlier, the definition of "activizer" is based on a trade-off between the strength of claimed improvements for nontrivial scenarios, and ease of analysis within the framework. There are two natural questions regarding the possibility of stronger notions of "activizer." In Definition 3 we allow a constant factor c loss in the ε argument of the label complexity. In most scenarios, this loss is inconsequential (e.g., typically $\Lambda_p(\varepsilon/c, f, \mathcal{P}) = O(\Lambda_p(\varepsilon, f, \mathcal{P}))$), but one can construct scenarios where it does make a difference. In our proofs, we see that it is possible to achieve $c = 3$; in fact, a careful inspection of the proofs reveals we can even get $c = (1 + o(1))$, a function of ε, converging to 1. However, whether there exist universal activizers for every VC class that have $c = 1$ remains an open question.

  A second question regards our notion of "nontrivial problems." In Definition 3, we have chosen to think of any target and distribution with label complexity growing faster than $\text{Polylog}(1/\varepsilon)$ as *nontrivial*, and do not require the activized algorithm to improve over the underlying passive algorithm for scenarios that are trivial for the passive algorithm. As mentioned, Definition 3 does have implications for the label complexities of these problems, as the label complexity of the activized algorithm will improve over every nontrivial upper bound on the label complexity of the passive algorithm. However, in order to allow for various operations in the meta-algorithm that may introduce additive $\text{Polylog}(1/\varepsilon)$ terms due to exponentially small failure probabilities, such as the test that selects among hypotheses in ActiveSelect, we do not require the activized algorithm to achieve the same *order* of label complexity in trivial scenarios. For instance, there may be cases in which a passive algorithm achieves $O(1)$ label complexity for a particular $(f, \mathcal{P})$, but its activized counterpart has $\Theta(\log(1/\varepsilon))$ label complexity. The intention is to define a framework that focuses on nontrivial scenarios, where passive learning uses prohibitively many labels, rather than one that requires us to obsess over extra additive logarithmic terms. Nonetheless, there is a question of whether these losses in the label complexities of trivial problems are necessary to gain these improvements in the label complexities of nontrivial problems.

There is also the question of how much the definition of "nontrivial" can be relaxed. Specifically, we have the following question: to what extent can we relax the notion of "nontrivial" in Definition 3, while still maintaining the existence of universal activizers for VC classes? We see from our proofs that we can at least replace $\text{Polylog}(1/\varepsilon)$ with $O(\log(1/\varepsilon))$. However, it is not clear whether we can go further than this in the realizable case (e.g., to say "nontrivial" means $\omega(1)$). When there is noise, it is clear that we cannot relax the notion of "nontrivial" beyond replacing $\text{Polylog}(1/\varepsilon)$ with $O(\log(1/\varepsilon))$. Specifically, whenever $\text{DIS}(\mathbb{C}) \neq \emptyset$, for any label complexity Λ_a achieved by an active learning algorithm, there must be some $\mathcal{P}_{XY}$ with $\Lambda_a(\nu+\varepsilon, \mathcal{P}_{XY}) = \Omega(\log(1/\varepsilon))$, even with the support of $\mathcal{P}$ restricted to a *single point* $x \in \text{DIS}(\mathbb{C})$; the proof of this is via a reduction from sequential hypothesis testing for whether a coin has bias α or $1-\alpha$, for some $\alpha \in (0, 1/2)$. Since passive learning via empirical risk minimization can achieve label complexity $\Lambda_p(\nu+\varepsilon, \mathcal{P}_{XY}) = O(\log(1/\varepsilon))$ whenever the support of $\mathcal{P}$ is restricted to a single point, we cannot further relax the notion of "nontrivial," while preserving the possibility of a positive outcome for Conjecture 23. It is interesting to note that this entire issue vanishes if we are only interested in methods that achieve error at most ε with probability at least $1-\delta$, where $\delta \in (0,1)$ is some acceptable constant failure probability, as in the work of Balcan, Hanneke, and Vaughan (2010); in this case, we can simply take "nontrivial" to mean $\omega(1)$ label complexity, and both Meta-Algorithm 1 and Meta-Algorithm 3 remain universal activizers for $\mathbb{C}$ under this alternative definition, and achieve $O(1)$ label complexity in trivial scenarios.

- Another interesting question concerns efficiency. Suppose there exists an algorithm to find an element of $\mathbb{C}$ consistent with any labeled sequence $\mathcal{L}$ in time polynomial in $|\mathcal{L}|$ and d, and that $\mathcal{A}_p(\mathcal{L})$ has running time polynomial in $|\mathcal{L}|$ and d. Under these conditions, is there an activizer for $\mathcal{A}_p$ capable of achieving an error rate smaller than any ε in running time polynomial in $1/\varepsilon$ and d, given some appropriately large budget n? Recall that if we knew the value of $\tilde{d}_f$ and $\tilde{d}_f \leq c \log d$, then Meta-Algorithm 1 could be made efficient, as discussed above. Therefore, this question is largely focused on the issue of adapting to the value of $\tilde{d}_f$. Another related question is whether there is an efficient active learning algorithm achieving the label complexity bound of Corollary 7 or Corollary 17.

- One question that comes up in the results above is the minimum number of *batches* of label requests necessary for a universal activizer for $\mathbb{C}$. In Meta-Algorithm 0 and Theorem 5, we saw that sometimes two batches are sufficient: one to reduce the version space, and another to construct the labeled sample by requesting only those points in the region of disagreement. We certainly cannot use fewer than two batches in a universal activizer for any nontrivial concept space, so that this represents the minimum. However, to get a universal activizer for *every* concept space, we increased the number of batches to *three* in Meta-Algorithm 1. The question is whether this increase is really necessary. Is there always a universal activizer using only *two* batches of label requests, for every VC class $\mathbb{C}$?

- For some $\mathbb{C}$, the learning process in the above methods might be viewed in two components: one component that performs active learning as usual (say, disagreement-based) under the assumption that the target function is very simple, and another component that searches for signs that the target function is in fact more complex. Thus, for some natural classes such as linear separators, it would be interesting to find simpler, more specialized methods, which

explicitly execute these two components. For instance, for the first component, we might consider the usual margin-based active learning methods, which query near a current guess of the separator (Dasgupta, Kalai, and Monteleoni, 2005, 2009; Balcan, Broder, and Zhang, 2007), except that we bias toward simple hypotheses via a regularization penalty in the optimization that defines how we update the separator in response to a query. The second component might then be a simple random search for points whose correct classification requires larger values of the regularization term.

- Can we construct universal activizers for some concept spaces with infinite VC dimension? What about under some constraints on the distribution $\mathcal{P}$ or $\mathcal{P}_{XY}$ (e.g., the usual entropy conditions)? It seems we can still run Meta-Algorithm 1, Meta-Algorithm 3, and Algorithm 5 in this case, except we should increase the number of rounds (values of k) as a function of n; this may continue to have reasonable behavior even in some cases where $\tilde{d}_f = \infty$, especially when $\mathcal{P}^k(\partial^k f) \to 0$ as $k \to \infty$. However, it is not clear whether they will continue to guarantee the strict improvements over passive learning in the realizable case, nor what label complexity guarantees they will achieve. One specific question is whether there is a method always achieving label complexity $o\left(\varepsilon^{\frac{1-\rho}{\kappa}-2}\right)$, where ρ is from the entropy conditions (van der Vaart and Wellner, 1996) and κ is from Condition 1. This would be an improvement over the known results for passive learning (Mammen and Tsybakov, 1999; Tsybakov, 2004; Koltchinskii, 2006). Another related question is whether we can improve over the known results for active learning in these scenarios. Specifically, Hanneke (2011) proved a bound of $\tilde{O}\left(\theta_f\left(\varepsilon^{\frac{1}{\kappa}}\right)\varepsilon^{\frac{2-\rho}{\kappa}-2}\right)$ on the label complexity of a certain disagreement-based active learning method, under entropy conditions and Condition 1. Do there exist active learning methods achieving asymptotically smaller label complexities than this, in particular improving the $\theta_f\left(\varepsilon^{\frac{1}{\kappa}}\right)$ factor? The quantity $\tilde{\theta}_f\left(\varepsilon^{\frac{1}{\kappa}}\right)$ is no longer defined when $\tilde{d}_f = \infty$, so this might not be a direct extension of Theorem 27, but we could perhaps use the sequence of $\theta_f^{(k)}\left(\varepsilon^{\frac{1}{\kappa}}\right)$ values in some other way to replace $\theta_f\left(\varepsilon^{\frac{1}{\kappa}}\right)$ in this case.

- Generalizing the previous question, we might even be so bold as to ask whether there exists a universal activizer for the space of *all* classifiers. Let us refer to such a method as a *universal activizer* (in general). The present work shows that there is a universal activizer for every VC class. Furthermore, Lemma 34 implies that, for any sequence $\mathbb{C}_1, \mathbb{C}_2, \ldots$ of concept spaces for which there exist universal activizers, there also exists a universal activizer for $\bigcup_{i=1}^{\infty} \mathbb{C}_i$: namely, the method that runs each of the activizers for $\mathbb{C}_i$ with respective budgets $\lfloor 3n/(\pi i)^2 \rfloor$, for $i = 1, 2, \ldots, \lfloor \sqrt{3n}/\pi \rfloor$, producing hypotheses $h_1, \ldots, h_{\lfloor \sqrt{3n}/\pi \rfloor}$, then returns the value of ActiveSelect$(\{h_1, \ldots, h_{\lfloor \sqrt{3n}/\pi \rfloor}\}, \lceil n/2 \rceil, \{X_M, X_{M+1}, \ldots\})$, where M is larger than any index accessed by these $\lfloor \sqrt{3n}/\pi \rfloor$ activizers. In fact, the proof of Theorem 6 entails that the $o(\Lambda_p(\varepsilon, f, \mathcal{P}))$ guarantee holds for f in the *closure* $\text{cl}(\mathbb{C})$ of any VC class $\mathbb{C}$. Combined with the above trick, it follows that we can achieve the $o(\Lambda_p(\varepsilon, f, \mathcal{P}))$ strong improvement guarantee over passive learning for all f in $\bigcup_{i=1}^{\infty} \text{cl}(\mathbb{C}_i)$, where the $\mathbb{C}_i$ sets are VC classes. We can always construct a sequence of VC classes $\mathbb{C}_1, \mathbb{C}_2, \ldots$ such that $\text{cl}\left(\bigcup_{i=1}^{\infty} \mathbb{C}_i\right)$ is the set of all classifiers. However, $\bigcup_{i=1}^{\infty} \text{cl}(\mathbb{C}_i)$ is generally not the same as $\text{cl}\left(\bigcup_{i=1}^{\infty} \mathbb{C}_i\right)$, so that achieving $\Lambda_a(c\varepsilon, f, \mathcal{P}) = o(\Lambda_p(\varepsilon, f, \mathcal{P}))$ for all $f \in \bigcup_{i=1}^{\infty} \text{cl}(\mathbb{C}_i)$ does not necessarily guarantee the same for all $f \in \text{cl}\left(\bigcup_{i=1}^{\infty} \mathbb{C}_i\right)$. Thus, constructing a general universal activizer would be

a nontrivial extension of the present work, and the fundamental question of the existence (or nonexistence) of such meta-algorithms remains a fascinating open question.

- There is also a question about generalizing this approach to label spaces other than $\{-1,+1\}$, and possibly other loss functions. It should be straightforward to extend these results to the setting of multiclass classification. However, it is not clear what the implications would be for general structured prediction problems, where the label space may be quite large (even infinite), and the loss function involves a notion of *distance* between labels. From a practical perspective, this question is particularly interesting, since problems with more complicated label spaces are often the scenarios where active learning is most needed, as it takes substantial time or effort to label each example. At this time, there are no published theoretical results on the label complexity improvements achievable for general structured prediction problems.

- All of the claims in this work also hold when $\mathcal{A}_p$ is a *semi-supervised* passive learning algorithm, simply by withholding a set of unlabeled data points in a preprocessing step, and feeding them into the passive algorithm along with the labeled set generated by the activizer. However, it is not clear whether further claims are possible when activizing a semi-supervised algorithm, for instance by taking into account specific details of the learning bias used by the particular semi-supervised algorithm (e.g., a cluster assumption).

- The splitting index analysis of Dasgupta (2005) has the interesting feature of characterizing a *trade-off* between the number of label requests and the number of unlabeled examples used by the active learning algorithm. In the present work, we do not characterize any such trade-off. Indeed, the algorithms do not really have any parameter to adjust the number of unlabeled examples they use (aside from the precision of the $\hat{P}$ estimators), so that they simply use as many as they need and then halt. This is true in both the realizable case and in the agnostic case. It would be interesting to try to modify these algorithms and their analysis so that, when there are more unlabeled examples available than would be used by the above methods, the algorithms can take advantage of this in a way that can be reflected in improved label complexity bounds, and when there are fewer unlabeled examples available, the algorithms can alter their behavior to compensate for this, at the cost of an increased label complexity. This would be interesting both for the realizable and agnostic cases. In fact, in the agnostic case, there are no known methods that exhibit this type of trade-off.

- Finally, as mentioned in the previous section, there is a serious question concerning what types of algorithms can be activized in the agnostic case, and how large the improvements in label complexity will be. In particular, Conjecture 23 hypothesizes that for any VC class, we can activize some empirical risk minimization algorithm in the agnostic case. Resolving this conjecture (either positively or negatively) should significantly advance our understanding of the capabilities of active learning compared to passive learning.

Acknowledgments

I am grateful to Nina Balcan, Rui Castro, Sanjoy Dasgupta, Carlos Guestrin, Vladimir Koltchinskii, John Langford, Rob Nowak, Larry Wasserman, and Eric Xing for insightful discussions. I also thank the anonymous referees for several thoughtful comments and suggestions.

Appendix A. Proofs Related to Section 3: Disagreement-Based Learning

The following result follows from a theorem of Anthony and Bartlett (1999), based on the classic results of Vapnik (1982) (with slightly better constant factors); see also the work of Blumer, Ehrenfeucht, Haussler, and Warmuth (1989).

Lemma 29 *For any VC class $\mathbb{C}$, $m \in \mathbb{N}$, and classifier f such that $\forall r > 0, \mathrm{B}(f,r) \neq \emptyset$, let $V_m^\star = \{h \in \mathbb{C} : \forall i \leq m, h(X_i) = f(X_i)\}$; for any $\delta \in (0,1)$, there is an event $H_m(\delta)$ with $\mathbb{P}(H_m(\delta)) \geq 1-\delta$ such that, on $H_m(\delta)$, $V_m^\star \subseteq \mathrm{B}(f, \phi(m;\delta))$, where*

$$\phi(m;\delta) = 2\frac{d\ln\frac{2e\max\{m,d\}}{d} + \ln(2/\delta)}{m}.$$

A fact we will use repeatedly is that, for any $N(\varepsilon) = \omega(\log(1/\varepsilon))$, we have $\phi(N(\varepsilon);\varepsilon) = o(1)$.

Lemma 30 *For $\hat{P}_n(\mathrm{DIS}(V))$ from (1), on an event J_n with $\mathbb{P}(J_n) \geq 1 - 2\cdot\exp\{-n/4\}$,*

$$\max\{\mathcal{P}(\mathrm{DIS}(V)), 4/n\} \leq \hat{P}_n(\mathrm{DIS}(V)) \leq \max\{4\mathcal{P}(\mathrm{DIS}(V)), 8/n\}.$$

Proof Note that the sequence $\mathcal{U}_n$ from (1) is independent from both V and $\mathcal{L}$. By a Chernoff bound, on an event J_n with $\mathbb{P}(J_n) \geq 1 - 2\cdot\exp\{-n/4\}$,

$$\mathcal{P}(\mathrm{DIS}(V)) > 2/n \implies \frac{\mathcal{P}(\mathrm{DIS}(V))}{\frac{1}{n^2}\sum_{x\in\mathcal{U}_n}\mathbb{1}_{\mathrm{DIS}(V)}(x)} \in [1/2, 2],$$

$$\text{and } \mathcal{P}(\mathrm{DIS}(V)) \leq 2/n \implies \frac{1}{n^2}\sum_{x\in\mathcal{U}_n}\mathbb{1}_{\mathrm{DIS}(V)}(x) \leq 4/n.$$

This immediately implies the stated result. ■

Lemma 31 *Let $\lambda : (0,1) \to (0,\infty)$ and $L : \mathbb{N}\times(0,1) \to [0,\infty)$ be s.t. $\lambda(\varepsilon) = \omega(1)$, $L(1,\varepsilon) = 0$ and $L(n,\varepsilon) \to \infty$ as $n \to \infty$ for every $\varepsilon \in (0,1)$, and for any $\mathbb{N}$-valued $N(\varepsilon) = \omega(\lambda(\varepsilon))$, $L(N(\varepsilon),\varepsilon) = \omega(N(\varepsilon))$. Let $L^{-1}(m;\varepsilon) = \max\{n \in \mathbb{N} : L(n,\varepsilon) < m\}$ for every $m \in (0,\infty)$. Then for any $\Lambda : (0,1) \to (0,\infty)$ with $\Lambda(\varepsilon) = \omega(\lambda(\varepsilon))$, we have $L^{-1}(\Lambda(\varepsilon);\varepsilon) = o(\Lambda(\varepsilon))$.*

Proof First note that L^{-1} is well-defined and finite, due to the facts that $L(n,\varepsilon)$ can be 0 and is diverging in n. Let $\Lambda(\varepsilon) = \omega(\lambda(\varepsilon))$. It is fairly straightforward to show $L^{-1}(\Lambda(\varepsilon);\varepsilon) \neq \Omega(\Lambda(\varepsilon))$, but the stronger $o(\Lambda(\varepsilon))$ result takes slightly more work. Let $L(n,\varepsilon) = \min\left\{L(n,\varepsilon), n^2/\lambda(\varepsilon)\right\}$ for every $n \in \mathbb{N}$ and $\varepsilon \in (0,1)$, and let $L^{-1}(m;\varepsilon) = \max\{n \in \mathbb{N} : L(n,\varepsilon) < m\}$. We will first prove the result for L.

Note that by definition of L^{-1}, we know

$$\left(L^{-1}(\Lambda(\varepsilon);\varepsilon)+1\right)^2/\lambda(\varepsilon) \geq L\left(L^{-1}(\Lambda(\varepsilon);\varepsilon)+1,\varepsilon\right) \geq \Lambda(\varepsilon) = \omega(\lambda(\varepsilon)),$$

which implies $L^{-1}(\Lambda(\varepsilon);\varepsilon) = \omega(\lambda(\varepsilon))$. But, by definition of L^{-1} and the condition on L,

$$\Lambda(\varepsilon) > L\left(L^{-1}(\Lambda(\varepsilon);\varepsilon),\varepsilon\right) = \omega\left(L^{-1}(\Lambda(\varepsilon);\varepsilon)\right).$$

Since $L^{-1}(m;\varepsilon) \geq L^{-1}(m;\varepsilon)$ for all $m > 0$, this implies $\Lambda(\varepsilon) = \omega\left(L^{-1}(\Lambda(\varepsilon);\varepsilon)\right)$, or equivalently $L^{-1}(\Lambda(\varepsilon);\varepsilon) = o(\Lambda(\varepsilon))$. ■

Lemma 32 *For any VC class $\mathbb{C}$ and passive algorithm $\mathcal{A}_p$, if $\mathcal{A}_p$ achieves label complexity Λ_p, then Meta-Algorithm 0, with $\mathcal{A}_p$ as its argument, achieves a label complexity Λ_a such that, for every $f \in \mathbb{C}$ and distribution $\mathcal{P}$ over $\mathcal{X}$, if $\mathcal{P}(\partial_{\mathbb{C},\mathcal{P}} f) = 0$ and $\infty > \Lambda_p(\varepsilon, f, \mathcal{P}) = \omega(\log(1/\varepsilon))$, then $\Lambda_a(2\varepsilon, f, \mathcal{P}) = o(\Lambda_p(\varepsilon, f, \mathcal{P}))$.*

Proof This proof follows similar lines to a proof of a related result of Balcan, Hanneke, and Vaughan (2010). Suppose $\mathcal{A}_p$ achieves a label complexity Λ_p, and that $f \in \mathbb{C}$ and distribution $\mathcal{P}$ satisfy $\infty > \Lambda_p(\varepsilon, f, \mathcal{P}) = \omega(\log(1/\varepsilon))$ and $\mathcal{P}(\partial_{\mathbb{C},\mathcal{P}} f) = 0$. Let $\varepsilon \in (0,1)$. For $n \in \mathbb{N}$, let $\Delta_n(\varepsilon) = \mathcal{P}(\mathrm{DIS}(\mathrm{B}(f, \phi(\lfloor n/2 \rfloor; \varepsilon/2))))$, $L(n;\varepsilon) = \lfloor n / \max\{32/n, 16\Delta_n(\varepsilon)\} \rfloor$, and for $m \in (0,\infty)$ let $L^{-1}(m;\varepsilon) = \max\{n \in \mathbb{N} : L(n;\varepsilon) < m\}$. Suppose

$$n \geq \max\left\{12\ln(6/\varepsilon), 1 + L^{-1}(\Lambda_p(\varepsilon, f, \mathcal{P}); \varepsilon)\right\}.$$

Consider running Meta-Algorithm 0 with $\mathcal{A}_p$ and n as arguments, while f is the target function and $\mathcal{P}$ is the data distribution. Let V and $\mathcal{L}$ be as in Meta-Algorithm 0, and let $\hat{h}_n = \mathcal{A}_p(\mathcal{L})$ denote the classifier returned at the end.

By Lemma 29, on the event $H_{\lfloor n/2 \rfloor}(\varepsilon/2)$, $V \subseteq \mathrm{B}(f, \phi(\lfloor n/2 \rfloor; \varepsilon/2))$, so that $\mathcal{P}(\mathrm{DIS}(V)) \leq \Delta_n(\varepsilon)$. Letting $\mathcal{U} = \{X_{\lfloor n/2 \rfloor + 1}, \ldots, X_{\lfloor n/2 \rfloor + \lfloor n/(4\hat{\Delta}) \rfloor}\}$, by Lemma 30, on $H_{\lfloor n/2 \rfloor}(\varepsilon/2) \cap J_n$ we have

$$\lfloor n / \max\{32/n, 16\Delta_n(\varepsilon)\} \rfloor \leq |\mathcal{U}| \leq \lfloor n / \max\{4\mathcal{P}(\mathrm{DIS}(V)), 16/n\} \rfloor. \tag{7}$$

By a Chernoff bound, for an event K_n with $\mathbb{P}(K_n) \geq 1 - \exp\{-n/12\}$, on $H_{\lfloor n/2 \rfloor}(\varepsilon/2) \cap J_n \cap K_n$, $|\mathcal{U} \cap \mathrm{DIS}(V)| \leq 2\mathcal{P}(\mathrm{DIS}(V)) \cdot \lfloor n / \max\{4\mathcal{P}(\mathrm{DIS}(V)), 16/n\} \rfloor \leq \lceil n/2 \rceil$. Defining the event $G_n(\varepsilon) = H_{\lfloor n/2 \rfloor}(\varepsilon/2) \cap J_n \cap K_n$, we see that on $G_n(\varepsilon)$, every time $X_m \in \mathrm{DIS}(V)$ in Step 5 of Meta-Algorithm 0, we have $t < n$; therefore, since $f \in V$ implies that the inferred labels in Step 6 are correct as well, we have that on $G_n(\varepsilon)$,

$$\forall (x, \hat{y}) \in \mathcal{L}, \hat{y} = f(x). \tag{8}$$

Noting that

$$\mathbb{P}(G_n(\varepsilon)^c) \leq \mathbb{P}\left(H_{\lfloor n/2 \rfloor}(\varepsilon/2)^c\right) + \mathbb{P}(J_n^c) + \mathbb{P}(K_n^c) \leq \varepsilon/2 + 2 \cdot \exp\{-n/4\} + \exp\{-n/12\} \leq \varepsilon,$$

we have

$$\begin{aligned}
&\mathbb{E}\left[\mathrm{er}\left(\hat{h}_n\right)\right] \\
&\leq \mathbb{E}\left[\mathbb{1}_{G_n(\varepsilon)} \mathbb{1}\left[|\mathcal{L}| \geq \Lambda_p(\varepsilon, f, \mathcal{P})\right] \mathrm{er}\left(\hat{h}_n\right)\right] + \mathbb{P}(G_n(\varepsilon) \cap \{|\mathcal{L}| < \Lambda_p(\varepsilon, f, \mathcal{P})\}) + \mathbb{P}(G_n(\varepsilon)^c) \\
&\leq \mathbb{E}\left[\mathbb{1}_{G_n(\varepsilon)} \mathbb{1}\left[|\mathcal{L}| \geq \Lambda_p(\varepsilon, f, \mathcal{P})\right] \mathrm{er}\left(\mathcal{A}_p(\mathcal{L})\right)\right] + \mathbb{P}(G_n(\varepsilon) \cap \{|\mathcal{L}| < \Lambda_p(\varepsilon, f, \mathcal{P})\}) + \varepsilon.
\end{aligned} \tag{9}$$

On $G_n(\varepsilon)$, (7) implies $|\mathcal{L}| \geq L(n;\varepsilon)$, and we chose n large enough so that $L(n;\varepsilon) \geq \Lambda_p(\varepsilon, f, \mathcal{P})$. Thus, the second term in (9) is zero, and we have

$$\begin{aligned}
\mathbb{E}\left[\mathrm{er}\left(\hat{h}_n\right)\right] &\leq \mathbb{E}\left[\mathbb{1}_{G_n(\varepsilon)} \mathbb{1}\left[|\mathcal{L}| \geq \Lambda_p(\varepsilon, f, \mathcal{P})\right] \mathrm{er}\left(\mathcal{A}_p(\mathcal{L})\right)\right] + \varepsilon \\
&= \mathbb{E}\left[\mathbb{E}\left[\mathbb{1}_{G_n(\varepsilon)} \mathrm{er}\left(\mathcal{A}_p(\mathcal{L})\right) \middle| |\mathcal{L}|\right] \mathbb{1}\left[|\mathcal{L}| \geq \Lambda_p(\varepsilon, f, \mathcal{P})\right]\right] + \varepsilon.
\end{aligned} \tag{10}$$

For any $\ell \in \mathbb{N}$ with $\mathbb{P}(|\mathcal{L}| = \ell) > 0$, the conditional of $\mathcal{U} | \{|\mathcal{U}| = \ell\}$ is a product distribution $\mathcal{P}^\ell$; that is, the samples in $\mathcal{U}$ are conditionally independent and identically distributed with distribution $\mathcal{P}$, which

is the same as the distribution of $\{X_1, X_2, \ldots, X_\ell\}$. Therefore, for any such ℓ with $\ell \geq \Lambda_p(\varepsilon, f, \mathcal{P})$, by (8) we have

$$\mathbb{E}\left[\mathbb{1}_{G_n(\varepsilon)} \mathrm{er}\left(\mathcal{A}_p\left(\mathcal{L}\right)\right) \middle| \{|\mathcal{L}| = \ell\}\right] \leq \mathbb{E}\left[\mathrm{er}\left(\mathcal{A}_p\left(\mathcal{Z}_\ell\right)\right)\right] \leq \varepsilon.$$

In particular, this means (10) is at most 2ε. This implies Meta-Algorithm 0, with $\mathcal{A}_p$ as its argument, achieves a label complexity Λ_a such that

$$\Lambda_a(2\varepsilon, f, \mathcal{P}) \leq \max\left\{12\ln(6/\varepsilon), 1 + L^{-1}\left(\Lambda_p(\varepsilon, f, \mathcal{P}); \varepsilon\right)\right\}.$$

Since $\Lambda_p(\varepsilon, f, \mathcal{P}) = \omega(\log(1/\varepsilon)) \Rightarrow 12\ln(6/\varepsilon) = o(\Lambda_p(\varepsilon, f, \mathcal{P}))$, it remains only to show that $L^{-1}\left(\Lambda_p(\varepsilon, f, \mathcal{P}); \varepsilon\right) = o(\Lambda_p(\varepsilon, f, \mathcal{P}))$. Note that $\forall \varepsilon \in (0,1)$, $L(1;\varepsilon) = 0$ and $L(n;\varepsilon)$ is diverging in n. Furthermore, by the assumption $\mathcal{P}(\partial_{\mathbb{C},\mathcal{P}} f) = 0$, we know that for any $N(\varepsilon) = \omega(\log(1/\varepsilon))$, we have $\Delta_{N(\varepsilon)}(\varepsilon) = o(1)$ (by continuity of probability measures), which implies $L(N(\varepsilon);\varepsilon) = \omega(N(\varepsilon))$. Thus, since $\Lambda_p(\varepsilon, f, \mathcal{P}) = \omega(\log(1/\varepsilon))$, Lemma 31 implies $L^{-1}\left(\Lambda_p(\varepsilon, f, \mathcal{P}); \varepsilon\right) = o\left(\Lambda_p(\varepsilon, f, \mathcal{P})\right)$, as desired. ■

Lemma 33 *For any VC class $\mathbb{C}$, target function $f \in \mathbb{C}$, and distribution $\mathcal{P}$, if $\mathcal{P}(\partial_{\mathbb{C},\mathcal{P}} f) > 0$, then there exists a passive learning algorithm $\mathcal{A}_p$ achieving a label complexity Λ_p such that $(f, \mathcal{P}) \in$ Nontrivial(Λ_p), and for any label complexity Λ_a achieved by running Meta-Algorithm 0 with $\mathcal{A}_p$ as its argument, and any constant $c \in (0, \infty)$,*

$$\Lambda_a(c\varepsilon, f, \mathcal{P}) \neq o(\Lambda_p(\varepsilon, f, \mathcal{P})).$$

Proof The proof can be broken down into three essential claims. First, it follows from Lemma 35 below that, on an event H' of probability one, $\mathcal{P}(\partial_V f) \geq \mathcal{P}(\partial_{\mathbb{C}} f)$; since $\mathcal{P}(\mathrm{DIS}(V)) \geq \mathcal{P}(\partial_V f)$, we have $\mathcal{P}(\mathrm{DIS}(V)) \geq \mathcal{P}(\partial_{\mathbb{C}} f)$ on H'.

The second claim is that on $H' \cap J_n$, $|\mathcal{L}| = O(n)$. This follows from Lemma 30 and our first claim by noting that, on $H' \cap J_n$, $|\mathcal{L}| = \left\lfloor n/(4\hat{\Delta})\right\rfloor \leq n/(4\mathcal{P}(\mathrm{DIS}(V))) \leq n/(4\mathcal{P}(\partial_{\mathbb{C}} f))$.

Finally, we construct a passive algorithm $\mathcal{A}_p$ whose label complexity is not significantly improved when $|\mathcal{L}| = O(n)$. There is a fairly obvious randomized $\mathcal{A}_p$ with this property (simply returning $-f$ with probability $1/|\mathcal{L}|$, and otherwise f); however, we can even satisfy the property with a deterministic $\mathcal{A}_p$, as follows. Let $\mathcal{H}_f = \{h_i\}_{i=1}^{\infty}$ be any sequence of classifiers (not necessarily in $\mathbb{C}$) with $0 < \mathcal{P}(x : h_i(x) \neq f(x))$ strictly decreasing to 0, (say with $h_1 = -f$). We know such a sequence must exist since $\mathcal{P}(\partial_{\mathbb{C}} f) > 0$. Now define, for nonempty S,

$$\mathcal{A}_p(S) = \operatorname*{argmin}_{h_i \in \mathcal{H}_f} \mathcal{P}(x : h_i(x) \neq f(x)) + 2\mathbb{1}_{[0,1/|S|)}(\mathcal{P}(x : h_i(x) \neq f(x))).$$

$\mathcal{A}_p$ is constructed so that, in the special case that this particular f is the target function and this particular $\mathcal{P}$ is the data distribution, $\mathcal{A}_p(S)$ returns the $h_i \in \mathcal{H}_f$ with minimal $\mathrm{er}(h_i)$ such that $\mathrm{er}(h_i) \geq 1/|S|$. For completeness, let $\mathcal{A}_p(\emptyset) = h_1$. Define $\varepsilon_i = \mathrm{er}(h_i) = \mathcal{P}(x : h_i(x) \neq f(x))$.

Now let $\hat{h}_n$ be the returned classifier from running Meta-Algorithm 0 with $\mathcal{A}_p$ and n as inputs, let Λ_p be the (minimal) label complexity achieved by $\mathcal{A}_p$, and let Λ_a be the (minimal) label complexity achieved by Meta-Algorithm 0 with $\mathcal{A}_p$ as input. Take any $c \in (0, \infty)$, and i sufficiently large so that $\varepsilon_{i-1} < 1/2$. Then we know that for any $\varepsilon \in [\varepsilon_i, \varepsilon_{i-1})$, $\Lambda_p(\varepsilon, f, \mathcal{P}) = \lceil 1/\varepsilon_i \rceil$. In particular,

$\Lambda_p(\varepsilon, f, \mathcal{P}) \geq 1/\varepsilon$, so that $(f, \mathcal{P}) \in \text{Nontrivial}(\Lambda_p)$. Also, by Markov's inequality and the above results on $|\mathcal{L}|$,

$$\mathbb{E}[\text{er}(\hat{h}_n)] \geq \mathbb{E}\left[\frac{1}{|\mathcal{L}|}\right] \geq \frac{4\mathcal{P}(\partial_{\mathbb{C}} f)}{n}\mathbb{P}\left(\frac{1}{|\mathcal{L}|} > \frac{4\mathcal{P}(\partial_{\mathbb{C}} f)}{n}\right)$$
$$\geq \frac{4\mathcal{P}(\partial_{\mathbb{C}} f)}{n}\mathbb{P}(H' \cap J_n) \geq \frac{4\mathcal{P}(\partial_{\mathbb{C}} f)}{n}\left(1 - 2\cdot\exp\{-n/4\}\right).$$

This implies that for $4\ln(4) < n < \frac{2\mathcal{P}(\partial_{\mathbb{C}} f)}{c\varepsilon_i}$, we have $\mathbb{E}\left[\text{er}(\hat{h}_n)\right] > c\varepsilon_i$, so that for all sufficiently large i,

$$\Lambda_a(c\varepsilon_i, f, \mathcal{P}) \geq \frac{2\mathcal{P}(\partial_{\mathbb{C}} f)}{c\varepsilon_i} \geq \frac{\mathcal{P}(\partial_{\mathbb{C}} f)}{c}\left\lceil\frac{1}{\varepsilon_i}\right\rceil = \frac{\mathcal{P}(\partial_{\mathbb{C}} f)}{c}\Lambda_p(\varepsilon_i, f, \mathcal{P}).$$

Since this happens for all sufficiently large i, and thus for arbitrarily small ε_i values, we have

$$\Lambda_a(c\varepsilon, f, \mathcal{P}) \neq o\left(\Lambda_p(\varepsilon, f, \mathcal{P})\right).$$

■

Proof [Theorem 5] Theorem 5 now follows directly from Lemmas 32 and 33, corresponding to the "if" and "only if" parts of the claim, respectively. ■

Appendix B. Proofs Related to Section 4: Basic Activizer

In this section, we provide detailed definitions, lemmas and proofs related to Meta-Algorithm 1.

In fact, we will develop slightly more general results here. Specifically, we fix an arbitrary constant $\gamma \in (0,1)$, and will prove the result for a family of meta-algorithms parameterized by the value γ, used as the threshold in Steps 3 and 6 of Meta-Algorithm 1, which were set to $1/2$ above to simplify the algorithm. Thus, setting $\gamma = 1/2$ in the statements below will give the stated theorem.

Throughout this section, we will assume $\mathbb{C}$ is a VC class with VC dimension d, and let $\mathcal{P}$ denote the (arbitrary) marginal distribution of X_i ($\forall i$). We also fix an arbitrary classifier $f \in \text{cl}(\mathbb{C})$, where (as in Section 6) $\text{cl}(\mathbb{C}) = \{h : \forall r > 0, \text{B}(h,r) \neq \emptyset\}$ denotes the closure of $\mathbb{C}$. In the present context, f corresponds to the target function when running Meta-Algorithm 1. Thus, we will study the behavior of Meta-Algorithm 1 for this fixed f and $\mathcal{P}$; since they are chosen arbitrarily, to establish Theorem 6 it will suffice to prove that for any passive $\mathcal{A}_p$, Meta-Algorithm 1 with $\mathcal{A}_p$ as input achieves superior label complexity compared to $\mathcal{A}_p$ for this f and $\mathcal{P}$. In fact, because here we only assume $f \in \text{cl}(\mathbb{C})$ (rather than $f \in \mathbb{C}$), we actually end up proving a slightly more general version of Theorem 6. But more importantly, this relaxation to $\text{cl}(\mathbb{C})$ will also make the lemmas developed below more useful for subsequent proofs: namely, those in Appendix E.2. For this same reason, many of the lemmas of this section are substantially more general than is necessary for the proof of Theorem 6; the more general versions will be used in the proofs of results in later sections.

For any $m \in \mathbb{N}$, we define $V_m^\star = \{h \in \mathbb{C} : \forall i \leq m, h(X_i) = f(X_i)\}$. Additionally, for $\mathcal{H} \subseteq \mathbb{C}$, and an integer $k \geq 0$, we will adopt the notation

$$\mathcal{S}^k(\mathcal{H}) = \left\{S \in \mathcal{X}^k : \mathcal{H} \text{ shatters } S\right\},$$
$$\mathcal{S}^k(\mathcal{H}) = \mathcal{X}^k \setminus \mathcal{S}^k(\mathcal{H}),$$

and as in Section 5, we define the k-dimensional shatter core of f with respect to $\mathcal{H}$ (and $\mathcal{P}$) as

$$\partial^k_{\mathcal{H}} f = \lim_{r \to 0} \mathcal{S}^k \left(\mathrm{B}_{\mathcal{H}}(f, r) \right),$$

and further define

$$\partial^k_{\mathcal{H}} f = \mathcal{X}^k \setminus \partial^k_{\mathcal{H}} f.$$

Also as in Section 5, define

$$\tilde{d}_f = \min \left\{ k \in \mathbb{N} : \mathcal{P}^k \left(\partial^k_{\mathbb{C}} f \right) = 0 \right\}.$$

For convenience, we also define the abbreviation

$$\tilde{\delta}_f = \mathcal{P}^{\tilde{d}_f - 1} \left(\partial^{\tilde{d}_f - 1}_{\mathbb{C}} f \right).$$

Also, recall that we are using the convention that $\mathcal{X}^0 = \{\varnothing\}$, $\mathcal{P}^0(\mathcal{X}^0) = 1$, and we say a set of classifiers $\mathcal{H}$ shatters $\varnothing$ iff $\mathcal{H} \neq \{\}$. In particular, $\mathcal{S}^0(\mathcal{H}) \neq \{\}$ iff $\mathcal{H} \neq \{\}$, and $\partial^0_{\mathcal{H}} f \neq \{\}$ iff $\inf_{h \in \mathcal{H}} \mathcal{P}(x : h(x) \neq f(x)) = 0$. For any measurable sets $S_1, S_2 \subseteq \mathcal{X}^k$ with $\mathcal{P}^k(S_2) > 0$, as usual we define $\mathcal{P}^k(S_1 | S_2) = \mathcal{P}^k(S_1 \cap S_2) / \mathcal{P}^k(S_2)$; in the situation where $\mathcal{P}^k(S_2) = 0$, it will be convenient to define $\mathcal{P}^k(S_1 | S_2) = 0$. We use the definition of $\mathrm{er}(h)$ from above, and additionally define the *conditional* error rate $\mathrm{er}(h|S) = \mathcal{P}(\{x : h(x) \neq f(x)\} | S)$ for any measurable $S \subseteq \mathcal{X}$. We also adopt the usual short-hand for equalities and inequalities involving conditional expectations and probabilities given random variables, wherein for instance, we write $\mathbb{E}[X|Y] = Z$ to mean that there is a version of $\mathbb{E}[X|Y]$ that is everywhere equal to Z, so that in particular, any version of $\mathbb{E}[X|Y]$ equals Z almost everywhere (see, e.g., Ash and Doléans-Dade, 2000).

B.1 Definition of Estimators for Meta-Algorithm 1

While the estimated probabilities used in Meta-Algorithm 1 can be defined in a variety of ways to make it a universal activizer for $\mathbb{C}$, in the statement of Theorem 6 above and proof thereof below, we take the following specific definitions. After the definition, we discuss alternative possibilities.

Though it is a slight twist on the formal model, it will greatly simplify our discussion below to suppose we have access to two independent sequences of i.i.d. unlabeled examples $W_1 = \{w_1, w_2, \ldots\}$ and $W_2 = \{w'_1, w'_2, \ldots\}$, also independent from the main sequence $\{X_1, X_2, \ldots\}$, with $w_i, w'_i \sim \mathcal{P}$. Since the data sequence $\{X_1, X_2, \ldots\}$ is i.i.d., this is distributionally equivalent to supposing we partition the data sequence in a preprocessing step, into three subsequences, alternatingly assigning each data point to either $\mathcal{Z}'_X$, W_1, or W_2. Then, if we suppose $\mathcal{Z}'_X = \{X'_1, X'_2, \ldots\}$, and we replace all references to X_i with X'_i in the algorithms and results, we obtain the equivalent statements holding for the model as originally stated. Thus, supposing the existence of these W_i sequences simply serves to simplify notation, and does not represent a further assumption on top of the previously stated framework.

For each $k \geq 2$, we partition W_2 into subsets of size $k-1$, as follows. For $i \in \mathbb{N}$, let

$$S^{(k)}_i = \{w'_{1+(i-1)(k-1)}, \ldots, w'_{i(k-1)}\}.$$

We define the $\hat{P}_m$ estimators in terms of three types of functions, defined below. For any $\mathcal{H} \subseteq \mathbb{C}$, $x \in \mathcal{X}$, $y \in \{-1,+1\}$, $m \in \mathbb{N}$, we define

$$\hat{P}_m\left(S \in \mathcal{X}^{k-1} : \mathcal{H} \text{ shatters } S \cup \{x\} | \mathcal{H} \text{ shatters } S\right) = \hat{\Delta}_m^{(k)}(x, W_2, \mathcal{H}), \tag{11}$$

$$\hat{P}_m\left(S \in \mathcal{X}^{k-1} : \mathcal{H}[(x,-y)] \text{ does not shatter } S | \mathcal{H} \text{ shatters } S\right) = \hat{\Gamma}_m^{(k)}(x, y, W_2, \mathcal{H}), \tag{12}$$

$$\hat{P}_m\left(x : \hat{P}\left(S \in \mathcal{X}^{k-1} : \mathcal{H} \text{ shatters } S \cup \{x\} | \mathcal{H} \text{ shatters } S\right) \geq \gamma\right) = \hat{\Delta}_m^{(k)}(W_1, W_2, \mathcal{H}). \tag{13}$$

The quantities $\hat{\Delta}_m^{(k)}(x, W_2, \mathcal{H})$, $\hat{\Gamma}_m^{(k)}(x, y, W_2, \mathcal{H})$, and $\hat{\Delta}_m^{(k)}(W_1, W_2, \mathcal{H})$ are specified as follows.

For $k = 1$, $\hat{\Gamma}_m^{(1)}(x, y, W_2, \mathcal{H})$ is simply an indicator for whether every $h \in \mathcal{H}$ has $h(x) = y$, while $\hat{\Delta}_m^{(1)}(x, W_2, \mathcal{H})$ is an indicator for whether $x \in \mathrm{DIS}(\mathcal{H})$. Formally, they are defined as follows.

$$\hat{\Gamma}_m^{(1)}(x, y, W_2, \mathcal{H}) = \mathbb{1}_{\bigcap_{h \in \mathcal{H}} \{h(x)\}}(y).$$

$$\hat{\Delta}_m^{(1)}(x, W_2, \mathcal{H}) = \mathbb{1}_{\mathrm{DIS}(\mathcal{H})}(x).$$

For $k \geq 2$, we first define

$$M_m^{(k)}(\mathcal{H}) = \max\left\{1, \sum_{i=1}^{m^3} \mathbb{1}_{\mathcal{S}^{k-1}(\mathcal{H})}\left(S_i^{(k)}\right)\right\}.$$

Then we take the following definitions for $\hat{\Gamma}^{(k)}$ and $\hat{\Delta}^{(k)}$.

$$\hat{\Gamma}_m^{(k)}(x, y, W_2, \mathcal{H}) = \frac{1}{M_m^{(k)}(\mathcal{H})} \sum_{i=1}^{m^3} \mathbb{1}_{\mathcal{S}^{k-1}(\mathcal{H}[(x,-y)])}\left(S_i^{(k)}\right) \mathbb{1}_{\mathcal{S}^{k-1}(\mathcal{H})}\left(S_i^{(k)}\right). \tag{14}$$

$$\hat{\Delta}_m^{(k)}(x, W_2, \mathcal{H}) = \frac{1}{M_m^{(k)}(\mathcal{H})} \sum_{i=1}^{m^3} \mathbb{1}_{\mathcal{S}^{k}(\mathcal{H})}\left(S_i^{(k)} \cup \{x\}\right). \tag{15}$$

For the remaining estimator, for any k we generally define

$$\hat{\Delta}_m^{(k)}(W_1, W_2, \mathcal{H}) = \frac{2}{m} + \frac{1}{m^3} \sum_{i=1}^{m^3} \mathbb{1}_{[\gamma/4,\infty)}\left(\hat{\Delta}_m^{(k)}(w_i, W_2, \mathcal{H})\right).$$

The above definitions will be used in the proofs below. However, there are certainly viable alternative definitions one can consider, some of which may have interesting theoretical properties. In general, one has the same sorts of trade-offs present whenever estimating a conditional probability. For instance, we could replace "m^3" in (14) and (15) by $\min\left\{\ell \in \mathbb{N} : M_\ell^{(k)}(\mathcal{H}) = m^3\right\}$, and then normalize by m^3 instead of $M_m^{(k)}(\mathcal{H})$; this would give us m^3 samples from the conditional distribution with which to estimate the conditional probability. The advantages of this approach would be its simplicity or elegance, and possibly some improvement in the constant factors in the label complexity bounds below. On the other hand, the drawback of this alternative definition would be that we do not know a priori how many unlabeled samples we will need to process in order to calculate it;

indeed, for some values of k and $\mathcal{H}$, we expect $\mathcal{P}^{k-1}\left(\mathcal{S}^{k-1}(\mathcal{H})\right)=0$, so that $M_\ell^{(k)}(\mathcal{H})$ is bounded, and we might technically need to examine the entire sequence to distinguish this case from the case of very small $\mathcal{P}^{k-1}\left(\mathcal{S}^{k-1}(\mathcal{H})\right)$. Of course, these practical issues can be addressed with small modifications, but only at the expense of complicating the analysis, thus losing the elegance factor. For these reasons, we have opted for the slightly looser and less elegant, but more practical, definitions above in (14) and (15).

B.2 Proof of Theorem 6

At a high level, the structure of the proof is the following. The primary components of the proof are three lemmas: 34, 37, and 38. Setting aside, for a moment, the fact that we are using the $\hat{P}_m$ estimators rather than the actual probability values they estimate, Lemma 38 indicates that the number of data points in $\mathcal{L}_{\tilde{d}_f}$ grows superlinearly in n (the number of label requests), while Lemma 37 guarantees that the labels of these points are correct, and Lemma 34 tells us that the classifier returned in the end is never much worse than $\mathcal{A}_p(\mathcal{L}_{\tilde{d}_f})$. These three factors combine to prove the result. The rest of the proof is composed of supporting lemmas and details regarding the $\hat{P}_m$ estimators. Specifically, Lemmas 35 and 36 serve a supporting role, with the purpose of showing that the set of V-shatterable k-tuples converges to the k-dimensional shatter core (up to probability-zero differences). The other lemmas below (39–45) are needed primarily to extend the above basic idea to the actual scenario where the $\hat{P}_m$ estimators are used as surrogates for the probability values. Additionally, a sub-case of Lemma 45 is needed in order to guarantee the label request budget will not be reached prematurely. Again, in many cases we prove a more general lemma than is required for its use in the proof of Theorem 6; these more general results will be needed in subsequent proofs: namely, in the proofs of Theorem 16 and Lemma 26.

We begin with a lemma concerning the ActiveSelect subroutine.

Lemma 34 *For any $k^*, M, N \in \mathbb{N}$ with $k^* \leq N$, and N classifiers $\{h_1, h_2, \ldots, h_N\}$ (themselves possibly random variables independent from $\{X_M, X_{M+1}, \ldots\}$), a call to* ActiveSelect($\{h_1, h_2, \ldots, h_N\}$, m, $\{X_M, X_{M+1}, \ldots\}$) *makes at most m label requests, and if $h_{\hat{k}}$ is the classifier it returns, then with probability at least $1 - eN \cdot \exp\{-m/(72k^*N\ln(eN))\}$, we have $\mathrm{er}(h_{\hat{k}}) \leq 2\,\mathrm{er}(h_{k^*})$.*

Proof This proof is essentially identical to a similar result of Balcan, Hanneke, and Vaughan (2010), but is included here for completeness.

Let $M_k = \left\lfloor \frac{m}{k(N-k)\ln(eN)} \right\rfloor$. First note that the total number of label requests in ActiveSelect is at most m, since summing up the sizes of the batches of label requests made in all executions of Step 2 yields

$$\sum_{j=1}^{N-1}\sum_{k=j+1}^{N}\left\lfloor \frac{m}{j(N-j)\ln(eN)} \right\rfloor \leq \sum_{j=1}^{N-1}\frac{m}{j\ln(eN)} \leq m.$$

Let $k^{**} = \mathrm{argmin}_{k\in\{1,\ldots,k^*\}}\mathrm{er}(h_k)$. For any $j \in \{1,2,\ldots,k^{**}-1\}$ with $\mathcal{P}(x : h_j(x) \neq h_{k^{**}}(x)) > 0$, the law of large numbers implies that with probability one $|\{X_M, X_{M+1}, \ldots\} \cap \{x : h_j(x) \neq h_{k^{**}}(x)\}| \geq M_j$, and since $\mathrm{er}(h_{k^{**}} | \{x : h_j(x) \neq h_{k^{**}}(x)\}) \leq 1/2$, Hoeffding's inequality implies that

$$\mathbb{P}\left(m_{k^{**}j} > 7/12\right) \leq \exp\left\{-M_j/72\right\} \leq \exp\left\{1 - m/(72k^*N\ln(eN))\right\}.$$

A union bound implies

$$\mathbb{P}\left(\max_{j<k^{**}} m_{k^{**}j} > 7/12\right) \leq k^{**} \cdot \exp\left\{1 - m/\left(72k^*N\ln(eN)\right)\right\}.$$

In particular, note that when $\max_{j<k^{**}} m_{k^{**}j} \leq 7/12$, we must have $\hat{k} \geq k^{**}$.

Now suppose $j \in \{k^{**}+1,\ldots,N\}$ has $\mathrm{er}(h_j) > 2\mathrm{er}(h_{k^{**}})$. In particular, this implies $\mathrm{er}(h_j|\{x : h_{k^{**}}(x) \neq h_j(x)\}) > 2/3$ and $\mathcal{P}(x : h_j(x) \neq h_{k^{**}}(x)) > 0$, which again means (with probability one) $|\{X_M, X_{M+1}, \ldots\} \cap \{x : h_j(x) \neq h_{k^{**}}(x)\}| \geq M_{k^{**}}$. By Hoeffding's inequality, we have that

$$\mathbb{P}\left(m_{jk^{**}} \leq 7/12\right) \leq \exp\left\{-M_{k^{**}}/72\right\} \leq \exp\left\{1 - m/\left(72k^*N\ln(eN)\right)\right\}.$$

By a union bound, we have that

$$\mathbb{P}\left(\exists j > k^{**} : \mathrm{er}(h_j) > 2\mathrm{er}(h_{k^{**}}) \text{ and } m_{jk^{**}} \leq 7/12\right) \leq (N-k^{**}) \cdot \exp\left\{1 - m/\left(72k^*N\ln(eN)\right)\right\}.$$

In particular, when $\hat{k} \geq k^{**}$, and $m_{jk^{**}} > 7/12$ for all $j > k^{**}$ with $\mathrm{er}(h_j) > 2\mathrm{er}(h_{k^{**}})$, it must be true that $\mathrm{er}(h_{\hat{k}}) \leq 2\mathrm{er}(h_{k^{**}}) \leq 2\mathrm{er}(h_{k^*})$.

So, by a union bound, with probability $\geq 1 - eN \cdot \exp\left\{-m/\left(72k^*N\ln(eN)\right)\right\}$, the $\hat{k}$ chosen by ActiveSelect has $\mathrm{er}(h_{\hat{k}}) \leq 2\mathrm{er}(h_{k^*})$. ■

The next two lemmas describe the limiting behavior of $\mathcal{S}^k(V_m^\star)$. In particular, we see that its limiting value is precisely $\partial^k_{\mathbb{C}} f$ (up to zero-probability differences). Lemma 35 establishes that $\mathcal{S}^k(V_m^\star)$ does not decrease below $\partial^k_{\mathbb{C}} f$ (except for a zero-probability set), and Lemma 36 establishes that its limit is not larger than $\partial^k_{\mathbb{C}} f$ (again, except for a zero-probability set).

Lemma 35 *There is an event H' with $\mathbb{P}(H') = 1$ such that on H', $\forall m \in \mathbb{N}$, $\forall k \in \{0,\ldots,\tilde{d}_f - 1\}$, for any $\mathcal{H}$ with $V_m^\star \subseteq \mathcal{H} \subseteq \mathbb{C}$,*

$$\mathcal{P}^k\left(\mathcal{S}^k(\mathcal{H}) \middle| \partial^k_{\mathbb{C}} f\right) = \mathcal{P}^k\left(\partial^k_{\mathcal{H}} f \middle| \partial^k_{\mathbb{C}} f\right) = 1,$$

and

$$\forall i \in \mathbb{N}, \mathbb{1}_{\partial^k_{\mathcal{H}} f}\left(S_i^{(k+1)}\right) = \mathbb{1}_{\partial^k_{\mathbb{C}} f}\left(S_i^{(k+1)}\right).$$

Also, on H', every such $\mathcal{H}$ has $\mathcal{P}^k\left(\partial^k_{\mathcal{H}} f\right) = \mathcal{P}^k\left(\partial^k_{\mathbb{C}} f\right)$, and $M_\ell^{(k)}(\mathcal{H}) \to \infty$ as $\ell \to \infty$.

Proof We will show the first claim for the set $V_m^\star$, and the result will then hold for $\mathcal{H}$ by monotonicity. In particular, we will show this for any fixed $k \in \{0,\ldots,\tilde{d}_f - 1\}$ and $m \in \mathbb{N}$, and the existence of H' then holds by a union bound. Fix any set $S \in \partial^k_{\mathbb{C}} f$. Suppose $\mathrm{B}_{V_m^\star}(f,r)$ does not shatter S for some $r > 0$. There is an infinite sequence of sets $\{\{h_1^{(i)}, h_2^{(i)}, \ldots, h_{2^k}^{(i)}\}\}_i$ with $\forall j \leq 2^k$, $\mathcal{P}(x : h_j^{(i)}(x) \neq f(x)) \downarrow 0$, such that each $\{h_1^{(i)},\ldots,h_{2^k}^{(i)}\} \subseteq \mathrm{B}(f,r)$ and shatters S. Since $\mathrm{B}_{V_m^\star}(f,r)$ does not shatter S,

$$1 = \inf_i \mathbb{1}\left[\exists j : h_j^{(i)} \notin \mathrm{B}_{V_m^\star}(f,r)\right] = \inf_i \mathbb{1}\left[\exists j \leq 2^k, \ell \leq m : h_j^{(i)}(X_\ell) \neq f(X_\ell)\right].$$

But

$$\mathbb{P}\left(\inf_i \mathbb{1}\left[\exists j \le 2^k, \ell \le m : h_j^{(i)}(X_\ell) \ne f(X_\ell)\right] = 1\right) \le \inf_i \mathbb{P}\left(\exists j \le 2^k, \ell \le m : h_j^{(i)}(X_\ell) \ne f(X_\ell)\right)$$
$$\le \lim_{i\to\infty} \sum_{j\le 2^k} m\mathcal{P}\left(x : h_j^{(i)}(x) \ne f(x)\right) = \sum_{j\le 2^k} m \lim_{i\to\infty} \mathcal{P}\left(x : h_j^{(i)}(x) \ne f(x)\right) = 0,$$

where the second inequality follows by a union bound. Therefore, $\forall r > 0, \mathbb{P}\left(S \notin \mathcal{S}^k\left(\mathrm{B}_{V_m^\star}(f,r)\right)\right) = 0$. Furthermore, since $\mathcal{S}^k\left(\mathrm{B}_{V_m^\star}(f,r)\right)$ is monotonic in r, the dominated convergence theorem gives us that

$$\mathbb{P}\left(S \notin \partial^k_{V_m^\star} f\right) = \mathbb{E}\left[\lim_{r\to 0} \mathbb{1}_{\mathcal{S}^k(\mathrm{B}_{V_m^\star}(f,r))}(S)\right] = \lim_{r\to 0} \mathbb{P}\left(S \notin \mathcal{S}^k\left(\mathrm{B}_{V_m^\star}(f,r)\right)\right) = 0.$$

This implies that (letting $\mathbf{S} \sim \mathcal{P}^k$ be independent from $V_m^\star$)

$$\begin{aligned}
\mathbb{P}\left(\mathcal{P}^k\left(\partial^k_{V_m^\star} f \middle| \partial^k_{\mathbb{C}} f\right) > 0\right) &= \mathbb{P}\left(\mathcal{P}^k\left(\partial^k_{V_m^\star} f \cap \partial^k_{\mathbb{C}} f\right) > 0\right) \\
&= \lim_{\xi\to 0} \mathbb{P}\left(\mathcal{P}^k\left(\partial^k_{V_m^\star} f \cap \partial^k_{\mathbb{C}} f\right) > \xi\right) \\
&\le \lim_{\xi\to 0} \frac{1}{\xi}\mathbb{E}\left[\mathcal{P}^k\left(\partial^k_{V_m^\star} f \cap \partial^k_{\mathbb{C}} f\right)\right] \qquad \text{(Markov)} \\
&= \lim_{\xi\to 0} \frac{1}{\xi}\mathbb{E}\left[\mathbb{1}_{\partial^k_{\mathbb{C}} f}(\mathbf{S})\mathbb{P}\left(\mathbf{S} \notin \partial^k_{V_m^\star} f \middle| \mathbf{S}\right)\right] \qquad \text{(Fubini)} \\
&= \lim_{\xi\to 0} 0 = 0.
\end{aligned}$$

This establishes the first claim for $V_m^\star$, on an event of probability 1, and monotonicity extends the claim to any $\mathcal{H} \supseteq V_m^\star$. Also note that, on this event,

$$\mathcal{P}^k\left(\partial^k_{\mathcal{H}} f\right) \ge \mathcal{P}^k\left(\partial^k_{\mathcal{H}} f \cap \partial^k_{\mathbb{C}} f\right) = \mathcal{P}^k\left(\partial^k_{\mathcal{H}} f \middle| \partial^k_{\mathbb{C}} f\right)\mathcal{P}^k\left(\partial^k_{\mathbb{C}} f\right) = \mathcal{P}^k\left(\partial^k_{\mathbb{C}} f\right),$$

where the last equality follows from the first claim. Noting that for $\mathcal{H} \subseteq \mathbb{C}$, $\partial^k_{\mathcal{H}} f \subseteq \partial^k_{\mathbb{C}} f$, we must have

$$\mathcal{P}^k\left(\partial^k_{\mathcal{H}} f\right) = \mathcal{P}^k\left(\partial^k_{\mathbb{C}} f\right).$$

This establishes the third claim. From the first claim, for any given value of $i \in \mathbb{N}$ the second claim holds for $S_i^{(k+1)}$ (with $\mathcal{H} = V_m^\star$) on an additional event of probability 1; taking a union bound over all $i \in \mathbb{N}$ extends this claim to every $S_i^{(k)}$ on an event of probability 1. Monotonicity then implies

$$\mathbb{1}_{\partial^k_{\mathbb{C}} f}\left(S_i^{(k+1)}\right) = \mathbb{1}_{\partial^k_{V_m^\star} f}\left(S_i^{(k+1)}\right) \le \mathbb{1}_{\partial^k_{\mathcal{H}} f}\left(S_i^{(k+1)}\right) \le \mathbb{1}_{\partial^k_{\mathbb{C}} f}\left(S_i^{(k+1)}\right),$$

extending the result to general $\mathcal{H}$. Also, as $k < \tilde{d}_f$, we know $\mathcal{P}^k\left(\partial^k_{\mathbb{C}} f\right) > 0$, and since we also know $V_m^\star$ is independent from W_2, the strong law of large numbers implies the final claim (for $V_m^\star$) on an additional event of probability 1; again, monotonicity extends this claim to any $\mathcal{H} \supseteq V_m^\star$. Intersecting the above events over values $m \in \mathbb{N}$ and $k < \tilde{d}_f$ gives the event H', and as each of the above events has probability 1 and there are countably many such events, a union bound implies $\mathbb{P}(H') = 1$. ■

Note that one specific implication of Lemma 35, obtained by taking $k=0$, is that on H', $V_m^\star \neq \emptyset$ (even if $f \in \mathrm{cl}(\mathbb{C}) \setminus \mathbb{C}$). This is because, for $f \in \mathrm{cl}(\mathbb{C})$, we have $\partial_{\mathbb{C}}^0 f = \mathcal{X}^0$ so that $\mathcal{P}^0\left(\partial_{\mathbb{C}}^0 f\right) = 1$, which means $\mathcal{P}^0\left(\partial_{V_m^\star}^0 f\right) = 1$ (on H'), so that we must have $\partial_{V_m^\star}^0 f = \mathcal{X}^0$, which implies $V_m^\star \neq \emptyset$. In particular, this also means $f \in \mathrm{cl}\left(V_m^\star\right)$.

Lemma 36 *There is a monotonic function $q(r) = o(1)$ (as $r \to 0$) such that, on event H', for any $k \in \left\{0,\ldots,\tilde{d}_f - 1\right\}$, $m \in \mathbb{N}$, $r > 0$, and set $\mathcal{H}$ such that $V_m^\star \subseteq \mathcal{H} \subseteq \mathrm{B}(f,r)$,*

$$\mathcal{P}^k\left(\partial_{\mathbb{C}}^k f \middle| \mathcal{S}^k\left(\mathcal{H}\right)\right) \leq q(r).$$

In particular, for $\tau \in \mathbb{N}$ and $\delta > 0$, on $H_\tau(\delta) \cap H'$ (where $H_\tau(\delta)$ is from Lemma 29), every $m \geq \tau$ and $k \in \left\{0,\ldots,\tilde{d}_f - 1\right\}$ has $\mathcal{P}^k\left(\partial_{\mathbb{C}}^k f \middle| \mathcal{S}^k\left(V_m^\star\right)\right) \leq q(\phi(\tau;\delta))$.

Proof Fix any $k \in \left\{0,\ldots,\tilde{d}_f - 1\right\}$. By Lemma 35, we know that on event H',

$$\begin{aligned}\mathcal{P}^k\left(\partial_{\mathbb{C}}^k f \middle| \mathcal{S}^k\left(\mathcal{H}\right)\right) &= \frac{\mathcal{P}^k\left(\partial_{\mathbb{C}}^k f \cap \mathcal{S}^k\left(\mathcal{H}\right)\right)}{\mathcal{P}^k\left(\mathcal{S}^k\left(\mathcal{H}\right)\right)} \leq \frac{\mathcal{P}^k\left(\partial_{\mathbb{C}}^k f \cap \mathcal{S}^k\left(\mathcal{H}\right)\right)}{\mathcal{P}^k\left(\partial_{\mathcal{H}}^k f\right)} \\ &= \frac{\mathcal{P}^k\left(\partial_{\mathbb{C}}^k f \cap \mathcal{S}^k\left(\mathcal{H}\right)\right)}{\mathcal{P}^k\left(\partial_{\mathbb{C}}^k f\right)} \leq \frac{\mathcal{P}^k\left(\partial_{\mathbb{C}}^k f \cap \mathcal{S}^k\left(\mathrm{B}\left(f,r\right)\right)\right)}{\mathcal{P}^k\left(\partial_{\mathbb{C}}^k f\right)}.\end{aligned}$$

Define $q_k(r)$ as this latter quantity. Since $\mathcal{P}^k\left(\partial_{\mathbb{C}}^k f \cap \mathcal{S}^k\left(\mathrm{B}(f,r)\right)\right)$ is monotonic in r,

$$\lim_{r\to 0} \frac{\mathcal{P}^k\left(\partial_{\mathbb{C}}^k f \cap \mathcal{S}^k\left(\mathrm{B}(f,r)\right)\right)}{\mathcal{P}^k\left(\partial_{\mathbb{C}}^k f\right)} = \frac{\mathcal{P}^k\left(\partial_{\mathbb{C}}^k f \cap \lim_{r\to 0} \mathcal{S}^k\left(\mathrm{B}(f,r)\right)\right)}{\mathcal{P}^k\left(\partial_{\mathbb{C}}^k f\right)} = \frac{\mathcal{P}^k\left(\partial_{\mathbb{C}}^k f \cap \partial_{\mathbb{C}}^k f\right)}{\mathcal{P}^k\left(\partial_{\mathbb{C}}^k f\right)} = 0.$$

This proves $q_k(r) = o(1)$. Defining

$$q(r) = \max\left\{q_k(r) : k \in \left\{0,1,\ldots,\tilde{d}_f - 1\right\}\right\} = o(1)$$

completes the proof of the first claim.

For the final claim, simply recall that by Lemma 29, on $H_\tau(\delta)$, every $m \geq \tau$ has $V_m^\star \subseteq V_\tau^\star \subseteq \mathrm{B}(f,\phi(\tau;\delta))$. ■

Lemma 37 *For $\xi \in (0,1)$, define*

$$r_\xi = \sup\left\{r \in (0,1) : q(r) < \xi\right\}/2.$$

On H', $\forall k \in \left\{0,\ldots,\tilde{d}_f - 1\right\}$, $\forall \xi \in (0,1)$, $\forall m \in \mathbb{N}$, for any set $\mathcal{H}$ such that $V_m^\star \subseteq \mathcal{H} \subseteq \mathrm{B}(f,r_\xi)$,

$$\begin{aligned}&\mathcal{P}\left(x : \mathcal{P}^k\left(\mathcal{S}^k\left(\mathcal{H}[(x,f(x))]\right) \middle| \mathcal{S}^k\left(\mathcal{H}\right)\right) > \xi\right) \\ &\qquad = \mathcal{P}\left(x : \mathcal{P}^k\left(\mathcal{S}^k\left(\mathcal{H}[(x,f(x))]\right) \middle| \partial_{\mathcal{H}}^k f\right) > \xi\right) = 0. \quad (16)\end{aligned}$$

In particular, for $\delta \in (0,1)$, defining $\tau(\xi;\delta) = \min\left\{\tau \in \mathbb{N} : \sup_{m \geq \tau} \phi(m;\delta) \leq r_\xi\right\}$, $\forall \tau \geq \tau(\xi;\delta)$, and $\forall m \geq \tau$, on $H_\tau(\delta) \cap H'$, (16) holds for $\mathcal{H} = V_m^\star$.

Proof Fix $k, m, \mathcal{H}$ as described above, and suppose $q = \mathcal{P}^k\left(\partial^k_{\mathbb{C}} f | \mathcal{S}^k(\mathcal{H})\right) < \zeta$; by Lemma 36, this happens on H'. Since, $\partial^k_{\mathcal{H}} f \subseteq \mathcal{S}^k(\mathcal{H})$, we have that $\forall x \in \mathcal{X}$,

$$\mathcal{P}^k\left(\mathcal{S}^k\left(\mathcal{H}[(x,f(x))]\right)\middle|\mathcal{S}^k(\mathcal{H})\right) = \mathcal{P}^k\left(\mathcal{S}^k\left(\mathcal{H}[(x,f(x))]\right)\middle|\partial^k_{\mathcal{H}} f\right)\mathcal{P}^k\left(\partial^k_{\mathcal{H}} f\middle|\mathcal{S}^k(\mathcal{H})\right) + \mathcal{P}^k\left(\mathcal{S}^k\left(\mathcal{H}[(x,f(x))]\right)\middle|\mathcal{S}^k(\mathcal{H})\cap\partial^k_{\mathcal{H}} f\right)\mathcal{P}^k\left(\partial^k_{\mathcal{H}} f\middle|\mathcal{S}^k(\mathcal{H})\right).$$

Since all probability values are bounded by 1, we have

$$\mathcal{P}^k\left(\mathcal{S}^k\left(\mathcal{H}[(x,f(x))]\right)\middle|\mathcal{S}^k(\mathcal{H})\right) \leq \mathcal{P}^k\left(\mathcal{S}^k\left(\mathcal{H}[(x,f(x))]\right)\middle|\partial^k_{\mathcal{H}} f\right) + \mathcal{P}^k\left(\partial^k_{\mathcal{H}} f\middle|\mathcal{S}^k(\mathcal{H})\right). \tag{17}$$

Isolating the right-most term in (17), by basic properties of probabilities we have

$$\begin{aligned}
&\mathcal{P}^k\left(\partial^k_{\mathcal{H}} f\middle|\mathcal{S}^k(\mathcal{H})\right)\\
&= \mathcal{P}^k\left(\partial^k_{\mathcal{H}} f\middle|\mathcal{S}^k(\mathcal{H})\cap\partial^k_{\mathbb{C}} f\right)\mathcal{P}^k\left(\partial^k_{\mathbb{C}} f\middle|\mathcal{S}^k(\mathcal{H})\right) + \mathcal{P}^k\left(\partial^k_{\mathcal{H}} f\middle|\mathcal{S}^k(\mathcal{H})\cap\partial^k_{\mathbb{C}} f\right)\mathcal{P}^k\left(\partial^k_{\mathbb{C}} f\middle|\mathcal{S}^k(\mathcal{H})\right)\\
&\leq \mathcal{P}^k\left(\partial^k_{\mathbb{C}} f\middle|\mathcal{S}^k(\mathcal{H})\right) + \mathcal{P}^k\left(\partial^k_{\mathcal{H}} f\middle|\mathcal{S}^k(\mathcal{H})\cap\partial^k_{\mathbb{C}} f\right).
\end{aligned} \tag{18}$$

By assumption, the left term in (18) equals q. Examining the right term in (18), we see that

$$\begin{aligned}
\mathcal{P}^k\left(\partial^k_{\mathcal{H}} f\middle|\mathcal{S}^k(\mathcal{H})\cap\partial^k_{\mathbb{C}} f\right) &= \mathcal{P}^k\left(\mathcal{S}^k(\mathcal{H})\cap\partial^k_{\mathcal{H}} f\middle|\partial^k_{\mathbb{C}} f\right)/\mathcal{P}^k\left(\mathcal{S}^k(\mathcal{H})\middle|\partial^k_{\mathbb{C}} f\right)\\
&\leq \mathcal{P}^k\left(\partial^k_{\mathcal{H}} f\middle|\partial^k_{\mathbb{C}} f\right)/\mathcal{P}^k\left(\partial^k_{\mathcal{H}} f\middle|\partial^k_{\mathbb{C}} f\right).
\end{aligned} \tag{19}$$

By Lemma 35, on H' the denominator in (19) is 1 and the numerator is 0. Thus, combining this fact with (17) and (18), we have that on H',

$$\mathcal{P}\left(x : \mathcal{P}^k\left(\mathcal{S}^k(\mathcal{H}[(x,f(x))])\middle|\mathcal{S}^k(\mathcal{H})\right) > \zeta\right) \leq \mathcal{P}\left(x : \mathcal{P}^k\left(\mathcal{S}^k(\mathcal{H}[(x,f(x))])\middle|\partial^k_{\mathcal{H}} f\right) > \zeta - q\right). \tag{20}$$

Note that proving the right side of (20) equals zero will suffice to establish the result, since it upper bounds *both* the first expression of (16) (as just established) *and* the second expression of (16) (by monotonicity of measures). Letting $X \sim \mathcal{P}$ be independent from the other random variables $(\mathcal{Z}, W_1, W_2)$, by Markov's inequality, the right side of (20) is at most

$$\frac{1}{\zeta - q}\mathbb{E}\left[\mathcal{P}^k\left(\mathcal{S}^k\left(\mathcal{H}[(X,f(X))]\right)\middle|\partial^k_{\mathcal{H}} f\right)\middle|\mathcal{H}\right] = \frac{\mathbb{E}\left[\mathcal{P}^k\left(\mathcal{S}^k\left(\mathcal{H}[(X,f(X))]\right)\cap\partial^k_{\mathcal{H}} f\right)\middle|\mathcal{H}\right]}{(\zeta - q)\mathcal{P}^k\left(\partial^k_{\mathcal{H}} f\right)},$$

and by Fubini's theorem, this is (letting $\mathbf{S} \sim \mathcal{P}^k$ be independent from the other random variables)

$$\frac{\mathbb{E}\left[\mathbb{1}_{\partial^k_{\mathcal{H}} f}(\mathbf{S})\mathcal{P}\left(x : \mathbf{S} \notin \mathcal{S}^k\left(\mathcal{H}[(x,f(x))]\right)\right)\middle|\mathcal{H}\right]}{(\zeta - q)\mathcal{P}^k\left(\partial^k_{\mathcal{H}} f\right)}.$$

Lemma 35 implies this equals

$$\frac{\mathbb{E}\left[\mathbb{1}_{\partial^k_{\mathcal{H}} f}(\mathbf{S})\mathcal{P}\left(x : \mathbf{S} \notin \mathcal{S}^k\left(\mathcal{H}[(x,f(x))]\right)\right)\middle|\mathcal{H}\right]}{(\zeta - q)\mathcal{P}^k\left(\partial^k_{\mathbb{C}} f\right)}. \tag{21}$$

For any fixed $S \in \partial^k_{\mathcal{H}} f$, there is an infinite sequence of sets $\left\{\left\{h_1^{(i)}, h_2^{(i)}, \ldots, h_{2^k}^{(i)}\right\}\right\}_{i\in\mathbb{N}}$ with $\forall j \leq 2^k, \mathcal{P}\left(x : h_j^{(i)}(x) \neq f(x)\right) \downarrow 0$, such that each $\left\{h_1^{(i)}, \ldots, h_{2^k}^{(i)}\right\} \subseteq \mathcal{H}$ and shatters S. If $\mathcal{H}[(x, f(x))]$ does not shatter S, then

$$1 = \inf_i \mathbb{1}\left[\exists j : h_j^{(i)} \notin \mathcal{H}[(x, f(x))]\right] = \inf_i \mathbb{1}\left[\exists j : h_j^{(i)}(x) \neq f(x)\right].$$

In particular,

$$\begin{aligned}
&\mathcal{P}\left(x : S \notin \mathcal{S}^k\left(\mathcal{H}[(x, f(x))]\right)\right) \leq \mathcal{P}\left(x : \inf_i \mathbb{1}\left[\exists j : h_j^{(i)}(x) \neq f(x)\right] = 1\right) \\
&= \mathcal{P}\left(\bigcap_i \left\{x : \exists j : h_j^{(i)}(x) \neq f(x)\right\}\right) \leq \inf_i \mathcal{P}\left(x : \exists j \text{ s.t. } h_j^{(i)}(x) \neq f(x)\right) \\
&\leq \lim_{i\to\infty} \sum_{j\leq 2^k} \mathcal{P}\left(x : h_j^{(i)}(x) \neq f(x)\right) = \sum_{j\leq 2^k} \lim_{i\to\infty} \mathcal{P}\left(x : h_j^{(i)}(x) \neq f(x)\right) = 0.
\end{aligned}$$

Thus (21) is zero, which establishes the result.

The final claim is then implied by Lemma 29 and monotonicity of $V_m^\star$ in m: that is, on $H_\tau(\delta)$, $V_m^\star \subseteq V_\tau^\star \subseteq \mathrm{B}(f, \phi(\tau;\delta)) \subseteq \mathrm{B}(f, r_\xi)$. ■

Lemma 38 *For any $\xi \in (0,1)$, there are values $\left\{\Delta_n^{(\xi)}(\varepsilon) : n \in \mathbb{N}, \varepsilon \in (0,1)\right\}$ such that, for any $n \in \mathbb{N}$ and $\varepsilon > 0$, on event $H_{\lfloor n/3\rfloor}(\varepsilon/2) \cap H'$, letting $V = V^\star_{\lfloor n/3\rfloor}$,*

$$\mathcal{P}\left(x : \mathcal{P}^{\tilde{d}_f - 1}\left(S \in \mathcal{X}^{\tilde{d}_f-1} : S \cup \{x\} \in \mathcal{S}^{\tilde{d}_f}(V) \middle| \mathcal{S}^{\tilde{d}_f-1}(V)\right) \geq \xi\right) \leq \Delta_n^{(\xi)}(\varepsilon),$$

and for any $\mathbb{N}$-valued $N(\varepsilon) = \omega(\log(1/\varepsilon))$, $\Delta^{(\xi)}_{N(\varepsilon)}(\varepsilon) = o(1)$.

Proof Throughout, we suppose the event $H_{\lfloor n/3\rfloor}(\varepsilon/2) \cap H'$, and fix some $\xi \in (0,1)$. We have $\forall x$,

$$\begin{aligned}
&\mathcal{P}^{\tilde{d}_f-1}\left(S \in \mathcal{X}^{\tilde{d}_f-1} : S \cup \{x\} \in \mathcal{S}^{\tilde{d}_f}(V) \middle| \mathcal{S}^{\tilde{d}_f-1}(V)\right) \\
&= \mathcal{P}^{\tilde{d}_f-1}\left(S \in \mathcal{X}^{\tilde{d}_f-1} : S \cup \{x\} \in \mathcal{S}^{\tilde{d}_f}(V) \middle| \mathcal{S}^{\tilde{d}_f-1}(V) \cap \partial^{\tilde{d}_f-1}_{\mathbb{C}} f\right) \mathcal{P}^{\tilde{d}_f-1}\left(\partial^{\tilde{d}_f-1}_{\mathbb{C}} f \middle| \mathcal{S}^{\tilde{d}_f-1}(V)\right) \\
&+ \mathcal{P}^{\tilde{d}_f-1}\left(S \in \mathcal{X}^{\tilde{d}_f-1} : S \cup \{x\} \in \mathcal{S}^{\tilde{d}_f}(V) \middle| \mathcal{S}^{\tilde{d}_f-1}(V) \cap \partial^{\tilde{d}_f-1}_{\mathbb{C}} f\right) \mathcal{P}^{\tilde{d}_f-1}\left(\partial^{\tilde{d}_f-1}_{\mathbb{C}} f \middle| \mathcal{S}^{\tilde{d}_f-1}(V)\right) \\
&\leq \mathcal{P}^{\tilde{d}_f-1}\left(S \in \mathcal{X}^{\tilde{d}_f-1} : S \cup \{x\} \in \mathcal{S}^{\tilde{d}_f}(V) \middle| \mathcal{S}^{\tilde{d}_f-1}(V) \cap \partial^{\tilde{d}_f-1}_{\mathbb{C}} f\right) + \mathcal{P}^{\tilde{d}_f-1}\left(\partial^{\tilde{d}_f-1}_{\mathbb{C}} f \middle| \mathcal{S}^{\tilde{d}_f-1}(V)\right). \quad (22)
\end{aligned}$$

By Lemma 35, the left term in (22) equals

$$\begin{aligned}
&\mathcal{P}^{\tilde{d}_f-1}\left(S \in \mathcal{X}^{\tilde{d}_f-1} : S \cup \{x\} \in \mathcal{S}^{\tilde{d}_f}(V) \middle| \mathcal{S}^{\tilde{d}_f-1}(V) \cap \partial^{\tilde{d}_f-1}_{\mathbb{C}} f\right) \mathcal{P}^{\tilde{d}_f-1}\left(\mathcal{S}^{\tilde{d}_f-1}(V) \middle| \partial^{\tilde{d}_f-1}_{\mathbb{C}} f\right) \\
&= \mathcal{P}^{\tilde{d}_f-1}\left(S \in \mathcal{X}^{\tilde{d}_f-1} : S \cup \{x\} \in \mathcal{S}^{\tilde{d}_f}(V) \middle| \partial^{\tilde{d}_f-1}_{\mathbb{C}} f\right),
\end{aligned}$$

and by Lemma 36, the right term in (22) is at most $q(\phi(\lfloor n/3\rfloor;\varepsilon/2))$. Thus, we have

$$\begin{aligned}&\mathcal{P}\left(x:\mathcal{P}^{\tilde{d}_f-1}\left(S\in\mathcal{X}^{\tilde{d}_f-1}:S\cup\{x\}\in\mathcal{S}^{\tilde{d}_f}(V)\middle|\mathcal{S}^{\tilde{d}_f-1}(V)\right)\geq\zeta\right)\\&\leq\mathcal{P}\left(x:\mathcal{P}^{\tilde{d}_f-1}\left(S\in\mathcal{X}^{\tilde{d}_f-1}:S\cup\{x\}\in\mathcal{S}^{\tilde{d}_f}(V)\middle|\partial_{\mathbb{C}}^{\tilde{d}_f-1}f\right)\geq\zeta-q(\phi(\lfloor n/3\rfloor;\varepsilon/2))\right).\end{aligned}\tag{23}$$

For $n<3\tau(\zeta/2;\varepsilon/2)$ (for $\tau(\cdot;\cdot)$ defined in Lemma 37), we define $\Delta_n^{(\zeta)}(\varepsilon)=1$. Otherwise, suppose $n\geq 3\tau(\zeta/2;\varepsilon/2)$, so that $q(\phi(\lfloor n/3\rfloor;\varepsilon/2))<\zeta/2$, and thus (23) is at most

$$\mathcal{P}\left(x:\mathcal{P}^{\tilde{d}_f-1}\left(S\in\mathcal{X}^{\tilde{d}_f-1}:S\cup\{x\}\in\mathcal{S}^{\tilde{d}_f}(V)\middle|\partial_{\mathbb{C}}^{\tilde{d}_f-1}f\right)\geq\zeta/2\right).$$

By Lemma 29, this is at most

$$\mathcal{P}\left(x:\mathcal{P}^{\tilde{d}_f-1}\left(S\in\mathcal{X}^{\tilde{d}_f-1}:S\cup\{x\}\in\mathcal{S}^{\tilde{d}_f}\left(\mathrm{B}(f,\phi(\lfloor n/3\rfloor;\varepsilon/2))\right)\middle|\partial_{\mathbb{C}}^{\tilde{d}_f-1}f\right)\geq\zeta/2\right).$$

Letting $X\sim\mathcal{P}$, by Markov's inequality this is at most

$$\begin{aligned}&\frac{2}{\zeta}\mathbb{E}\left[\mathcal{P}^{\tilde{d}_f-1}\left(S\in\mathcal{X}^{\tilde{d}_f-1}:S\cup\{X\}\in\mathcal{S}^{\tilde{d}_f}\left(\mathrm{B}(f,\phi(\lfloor n/3\rfloor;\varepsilon/2))\right)\middle|\partial_{\mathbb{C}}^{\tilde{d}_f-1}f\right)\right]\\&=\frac{2}{\zeta\tilde{\delta}_f}\mathcal{P}^{\tilde{d}_f}\left(S\cup\{x\}\in\mathcal{X}^{\tilde{d}_f}:S\cup\{x\}\in\mathcal{S}^{\tilde{d}_f}\left(\mathrm{B}(f,\phi(\lfloor n/3\rfloor;\varepsilon/2))\right)\text{ and }S\in\partial_{\mathbb{C}}^{\tilde{d}_f-1}f\right)\\&\leq\frac{2}{\zeta\tilde{\delta}_f}\mathcal{P}^{\tilde{d}_f}\left(\mathcal{S}^{\tilde{d}_f}\left(\mathrm{B}(f,\phi(\lfloor n/3\rfloor;\varepsilon/2))\right)\right).\end{aligned}\tag{24}$$

Thus, defining $\Delta_n^{(\zeta)}(\varepsilon)$ as (24) for $n\geq 3\tau(\zeta/2;\varepsilon/2)$ establishes the first claim.

It remains only to prove the second claim. Let $N(\varepsilon)=\omega(\log(1/\varepsilon))$. Since $\tau(\zeta/2;\varepsilon/2)\leq\left\lceil\frac{4}{r_{\zeta/2}}\left(d\ln\left(\frac{4e}{r_{\zeta/2}}\right)+\ln\left(\frac{4}{\varepsilon}\right)\right)\right\rceil=O(\log(1/\varepsilon))$, we have that for all sufficiently small $\varepsilon>0$, $N(\varepsilon)\geq 3\tau(\zeta/2;\varepsilon/2)$, so that $\Delta_{N(\varepsilon)}^{(\zeta)}(\varepsilon)$ equals (24) (with $n=N(\varepsilon)$). Furthermore, since $\tilde{\delta}_f>0$, while $\mathcal{P}^{\tilde{d}_f}\left(\partial_{\mathbb{C}}^{\tilde{d}_f}f\right)=0$, and $\phi(\lfloor N(\varepsilon)/3\rfloor;\varepsilon/2)=o(1)$, by continuity of probability measures we know (24) is $o(1)$ when $n=N(\varepsilon)$, so that we generally have $\Delta_{N(\varepsilon)}^{(\zeta)}(\varepsilon)=o(1)$. ■

For any $m\in\mathbb{N}$, define

$$\tilde{M}(m)=m^3\tilde{\delta}_f/2.$$

Lemma 39 *There is a $(\mathbb{C},\mathcal{P},f)$-dependent constant $c^{(i)}\in(0,\infty)$ such that, for any $\tau\in\mathbb{N}$ there is an event $H_\tau^{(i)}\subseteq H'$ with*

$$\mathbb{P}\left(H_\tau^{(i)}\right)\geq 1-c^{(i)}\cdot\exp\left\{-\tilde{M}(\tau)/4\right\}$$

such that on $H_\tau^{(i)}$, if $\tilde{d}_f\geq 2$, then $\forall k\in\left\{2,\ldots,\tilde{d}_f\right\}$, $\forall m\geq\tau$, $\forall\ell\in\mathbb{N}$, for any set $\mathcal{H}$ such that $V_\ell^\star\subseteq\mathcal{H}\subseteq\mathbb{C}$,

$$M_m^{(k)}(\mathcal{H})\geq\tilde{M}(m).$$

Proof On H', Lemma 35 implies every $\mathbb{1}_{\mathcal{S}^{k-1}(\mathcal{H})}\left(S_i^{(k)}\right) \geq \mathbb{1}_{\partial_{\mathcal{H}}^{k-1}f}\left(S_i^{(k)}\right) = \mathbb{1}_{\partial_{\mathbb{C}}^{k-1}f}\left(S_i^{(k)}\right)$, so we focus on showing $\left|\left\{S_i^{(k)} : i \leq m^3\right\} \cap \partial_{\mathbb{C}}^{k-1}f\right| \geq \tilde{M}(m)$ on an appropriate event. We know

$$\begin{aligned}
&\mathbb{P}\left(\forall k \in \left\{2,\ldots,\tilde{d}_f\right\}, \forall m \geq \tau, \left|\left\{S_i^{(k)} : i \leq m^3\right\} \cap \partial_{\mathbb{C}}^{k-1}f\right| \geq \tilde{M}(m)\right) \\
&= 1 - \mathbb{P}\left(\exists k \in \left\{2,\ldots,\tilde{d}_f\right\}, m \geq \tau : \left|\left\{S_i^{(k)} : i \leq m^3\right\} \cap \partial_{\mathbb{C}}^{k-1}f\right| < \tilde{M}(m)\right) \\
&\geq 1 - \sum_{m\geq\tau}\sum_{k=2}^{\tilde{d}_f} \mathbb{P}\left(\left|\left\{S_i^{(k)} : i \leq m^3\right\} \cap \partial_{\mathbb{C}}^{k-1}f\right| < \tilde{M}(m)\right),
\end{aligned}$$

where the last line follows by a union bound. Thus, we will focus on bounding

$$\sum_{m\geq\tau}\sum_{k=2}^{\tilde{d}_f} \mathbb{P}\left(\left|\left\{S_i^{(k)} : i \leq m^3\right\} \cap \partial_{\mathbb{C}}^{k-1}f\right| < \tilde{M}(m)\right). \tag{25}$$

Fix any $k \in \left\{2,\ldots,\tilde{d}_f\right\}$, and integer $m \geq \tau$. Since

$$\mathbb{E}\left[\left|\left\{S_i^{(k)} : i \leq m^3\right\} \cap \partial_{\mathbb{C}}^{k-1}f\right|\right] = \mathcal{P}^{k-1}\left(\partial_{\mathbb{C}}^{k-1}f\right)m^3 \geq \tilde{\delta}_f m^3,$$

a Chernoff bound implies that

$$\begin{aligned}
\mathbb{P}\left(\left|\left\{S_i^{(k)} : i \leq m^3\right\} \cap \partial_{\mathbb{C}}^{k-1}f\right| < \tilde{M}(m)\right) &\leq \exp\left\{-m^3\mathcal{P}^{k-1}\left(\partial_{\mathbb{C}}^{k-1}f\right)/8\right\} \\
&\leq \exp\left\{-m^3\tilde{\delta}_f/8\right\}.
\end{aligned}$$

Thus, we have that (25) is at most

$$\begin{aligned}
\sum_{m\geq\tau}\sum_{k=2}^{\tilde{d}_f}\exp\left\{-m^3\tilde{\delta}_f/8\right\} &\leq \sum_{m\geq\tau}\tilde{d}_f\cdot\exp\left\{-m^3\tilde{\delta}_f/8\right\} \leq \sum_{m\geq\tau^3}\tilde{d}_f\cdot\exp\left\{-m\tilde{\delta}_f/8\right\} \\
&\leq \tilde{d}_f\cdot\exp\left\{-\tilde{M}(\tau)/4\right\} + \tilde{d}_f\cdot\int_{\tau^3}^{\infty}\exp\left\{-x\tilde{\delta}_f/8\right\}\mathrm{d}x \\
&= \tilde{d}_f\cdot\left(1+8/\tilde{\delta}_f\right)\cdot\exp\left\{-\tilde{M}(\tau)/4\right\} \\
&\leq \left(9\tilde{d}_f/\tilde{\delta}_f\right)\cdot\exp\left\{-\tilde{M}(\tau)/4\right\}.
\end{aligned}$$

Note that since $\mathbb{P}(H') = 1$, defining

$$H_\tau^{(i)} = \left\{\forall k \in \left\{2,\ldots,\tilde{d}_f\right\}, \forall m \geq \tau, \left|\left\{S_i^{(k)} : i \leq m^3\right\} \cap \partial_{\mathbb{C}}^{k-1}f\right| \geq \tilde{M}(m)\right\} \cap H'$$

has the required properties. ■

Lemma 40 *For any $\tau \in \mathbb{N}$, there is an event $G_\tau^{(i)}$ with*

$$\mathbb{P}\left(H_\tau^{(i)} \setminus G_\tau^{(i)}\right) \leq \left(121\tilde{d}_f/\tilde{\delta}_f\right) \cdot \exp\left\{-\tilde{M}(\tau)/60\right\}$$

such that, on $G_\tau^{(i)}$, if $\tilde{d}_f \geq 2$, then for every integer $s \geq \tau$ and $k \in \left\{2,\ldots,\tilde{d}_f\right\}$, $\forall r \in \left(0, r_{1/6}\right]$,

$$M_s^{(k)}\left(\mathrm{B}\left(f,r\right)\right) \leq (3/2)\left|\left\{S_i^{(k)} : i \leq s^3\right\} \cap \partial_{\mathbb{C}}^{k-1} f\right|.$$

Proof Fix integers $s \geq \tau$ and $k \in \left\{2,\ldots,\tilde{d}_f\right\}$, and let $r = r_{1/6}$. Define the set $\hat{\mathcal{S}}^{k-1} = \left\{S_i^{(k)} : i \leq s^3\right\} \cap \mathcal{S}^{k-1}\left(\mathrm{B}\left(f,r\right)\right)$. Note $\left|\hat{\mathcal{S}}^{k-1}\right| = M_s^{(k)}\left(\mathrm{B}\left(f,r\right)\right)$ and the elements of $\hat{\mathcal{S}}^{k-1}$ are conditionally i.i.d. given $M_s^{(k)}\left(\mathrm{B}\left(f,r\right)\right)$, each with conditional distribution equivalent to the conditional distribution of $S_1^{(k)}$ given $\left\{S_1^{(k)} \in \mathcal{S}^{k-1}\left(\mathrm{B}\left(f,r\right)\right)\right\}$. In particular,

$$\mathbb{E}\left[\left|\hat{\mathcal{S}}^{k-1} \cap \partial_{\mathbb{C}}^{k-1} f\right| \middle| M_s^{(k)}\left(\mathrm{B}\left(f,r\right)\right)\right] = \mathcal{P}^{k-1}\left(\partial_{\mathbb{C}}^{k-1} f \middle| \mathcal{S}^{k-1}\left(\mathrm{B}\left(f,r\right)\right)\right) M_s^{(k)}\left(\mathrm{B}\left(f,r\right)\right).$$

Define the event

$$G_\tau^{(i)}(k,s) = \left\{\left|\hat{\mathcal{S}}^{k-1}\right| \leq (3/2)\left|\hat{\mathcal{S}}^{k-1} \cap \partial_{\mathbb{C}}^{k-1} f\right|\right\}.$$

By Lemma 36 (indeed by definition of $q(r)$ and $r_{1/6}$) we have

$$\begin{aligned}
&1 - \mathbb{P}\left(G_\tau^{(i)}(k,s) \middle| M_s^{(k)}\left(\mathrm{B}\left(f,r\right)\right)\right) \\
&= \mathbb{P}\left(\left|\hat{\mathcal{S}}^{k-1} \cap \partial_{\mathbb{C}}^{k-1} f\right| < (2/3) M_s^{(k)}\left(\mathrm{B}\left(f,r\right)\right) \middle| M_s^{(k)}\left(\mathrm{B}\left(f,r\right)\right)\right) \\
&\leq \mathbb{P}\left(\left|\hat{\mathcal{S}}^{k-1} \cap \partial_{\mathbb{C}}^{k-1} f\right| < (4/5)\left(1 - q\left(r\right)\right) M_s^{(k)}\left(\mathrm{B}\left(f,r\right)\right) \middle| M_s^{(k)}\left(\mathrm{B}\left(f,r\right)\right)\right) \\
&\leq \mathbb{P}\left(\left|\hat{\mathcal{S}}^{k-1} \cap \partial_{\mathbb{C}}^{k-1} f\right| < (4/5)\mathcal{P}^{k-1}\left(\partial_{\mathbb{C}}^{k-1} f \middle| \mathcal{S}^{k-1}\left(\mathrm{B}\left(f,r\right)\right)\right) M_s^{(k)}\left(\mathrm{B}\left(f,r\right)\right) \middle| M_s^{(k)}\left(\mathrm{B}\left(f,r\right)\right)\right). \quad (26)
\end{aligned}$$

By a Chernoff bound, (26) is at most

$$\begin{aligned}
&\exp\left\{-M_s^{(k)}\left(\mathrm{B}\left(f,r\right)\right)\mathcal{P}^{k-1}\left(\partial_{\mathbb{C}}^{k-1} f \middle| \mathcal{S}^{k-1}\left(\mathrm{B}\left(f,r\right)\right)\right)/50\right\} \\
&\qquad \leq \exp\left\{-M_s^{(k)}\left(\mathrm{B}\left(f,r\right)\right)\left(1 - q\left(r\right)\right)/50\right\} \leq \exp\left\{-M_s^{(k)}\left(\mathrm{B}\left(f,r\right)\right)/60\right\}.
\end{aligned}$$

Thus, by Lemma 39,

$$\begin{aligned}
&\mathbb{P}\left(H_\tau^{(i)} \setminus G_\tau^{(i)}(k,s)\right) \leq \mathbb{P}\left(\left\{M_s^{(k)}\left(\mathrm{B}\left(f,r\right)\right) \geq \tilde{M}(s)\right\} \setminus G_\tau^{(i)}(k,s)\right) \\
&= \mathbb{E}\left[\left(1 - \mathbb{P}\left(G_\tau^{(i)}(k,s) \middle| M_s^{(k)}\left(\mathrm{B}\left(f,r\right)\right)\right)\right) \mathbb{1}_{[\tilde{M}(s),\infty)}\left(M_s^{(k)}\left(\mathrm{B}\left(f,r\right)\right)\right)\right] \\
&\leq \mathbb{E}\left[\exp\left\{-M_s^{(k)}\left(\mathrm{B}\left(f,r\right)\right)/60\right\} \mathbb{1}_{[\tilde{M}(s),\infty)}\left(M_s^{(k)}\left(\mathrm{B}\left(f,r\right)\right)\right)\right] \leq \exp\left\{-\tilde{M}(s)/60\right\}.
\end{aligned}$$

Now defining $G_\tau^{(i)} = \bigcap_{s\geq\tau}\bigcap_{k=2}^{\tilde{d}_f} G_\tau^{(i)}(k,s)$, a union bound implies

$$\begin{aligned}\mathbb{P}\left(H_\tau^{(i)} \setminus G_\tau^{(i)}\right) &\leq \sum_{s\geq\tau} \tilde{d}_f \cdot \exp\left\{-\tilde{M}(s)/60\right\} \\ &\leq \tilde{d}_f\left(\exp\left\{-\tilde{M}(\tau)/60\right\} + \int_{\tau^3}^{\infty} \exp\left\{-x\tilde{\delta}_f/120\right\} \mathrm{d}x\right) \\ &= \tilde{d}_f\left(1 + 120/\tilde{\delta}_f\right) \cdot \exp\left\{-\tilde{M}(\tau)/60\right\} \\ &\leq \left(121\tilde{d}_f/\tilde{\delta}_f\right) \cdot \exp\left\{-\tilde{M}(\tau)/60\right\}.\end{aligned}$$

This completes the proof for $r = r_{1/6}$. Monotonicity extends the result to any $r \in \left(0, r_{1/6}\right]$. ■

Lemma 41 *There exist $(\mathbb{C},\mathcal{P},f,\gamma)$-dependent constants $\tau^* \in \mathbb{N}$ and $c^{(ii)} \in (0,\infty)$ such that, for any integer $\tau \geq \tau^*$, there is an event $H_\tau^{(ii)} \subseteq G_\tau^{(i)}$ with*

$$\mathbb{P}\left(H_\tau^{(i)} \setminus H_\tau^{(ii)}\right) \leq c^{(ii)} \cdot \exp\left\{-\tilde{M}(\tau)^{1/3}/60\right\} \tag{27}$$

such that, on $H_\tau^{(i)} \cap H_\tau^{(ii)}$, $\forall s,m,\ell,k \in \mathbb{N}$ with $\ell < m$ and $k \leq \tilde{d}_f$, for any set of classifiers $\mathcal{H}$ with $V_\ell^\star \subseteq \mathcal{H}$, if either $k = 1$, or $s \geq \tau$ and $\mathcal{H} \subseteq \mathrm{B}(f, r_{(1-\gamma)/6})$, then

$$\hat{\Delta}_s^{(k)}(X_m, W_2, \mathcal{H}) < \gamma \implies \hat{\Gamma}_s^{(k)}(X_m, -f(X_m), W_2, \mathcal{H}) < \hat{\Gamma}_s^{(k)}(X_m, f(X_m), W_2, \mathcal{H}).$$

In particular, for $\delta \in (0,1)$ and $\tau \geq \max\{\tau((1-\gamma)/6;\delta), \tau^*\}$, on $H_\tau(\delta) \cap H_\tau^{(i)} \cap H_\tau^{(ii)}$, this is true for $\mathcal{H} = V_\ell^\star$ for every $k,\ell,m,s \in \mathbb{N}$ satisfying $\tau \leq \ell < m$, $\tau \leq s$, and $k \leq \tilde{d}_f$.

Proof Let $\tau^* = (6/(1-\gamma)) \cdot \left(2/\tilde{\delta}_f\right)^{1/3}$, and consider any $\tau,k,\ell,m,s,\mathcal{H}$ as described above. If $k = 1$, the result clearly holds. In particular, Lemma 35 implies that on $H_\tau^{(i)}$, $\mathcal{H}[(X_m, f(X_m))] \supseteq V_m^\star \neq \emptyset$, so that some $h \in \mathcal{H}$ has $h(X_m) = f(X_m)$, and therefore

$$\hat{\Gamma}_s^{(1)}(X_m, -f(X_m), W_2, \mathcal{H}) = \mathbb{1}_{\bigcap_{h\in\mathcal{H}}\{h(X_m)\}}(-f(X_m)) = 0,$$

and since $\hat{\Delta}_s^{(1)}(X_m, W_2, \mathcal{H}) = \mathbb{1}_{\mathrm{DIS}(\mathcal{H})}(X_m)$, if $\hat{\Delta}_s^{(1)}(X_m, W_2, \mathcal{H}) < \gamma$, then since $\gamma < 1$ we have $X_m \notin \mathrm{DIS}(\mathcal{H})$, so that

$$\hat{\Gamma}_s^{(1)}(X_m, f(X_m), W_2, \mathcal{H}) = \mathbb{1}_{\bigcap_{h\in\mathcal{H}}\{h(X_m)\}}(f(X_m)) = 1.$$

Otherwise, suppose $2 \leq k \leq \tilde{d}_f$. Note that on $H_\tau^{(i)} \cap G_\tau^{(i)}$, $\forall m \in \mathbb{N}$, and any $\mathcal{H}$ with $V_\ell^\star \subseteq \mathcal{H} \subseteq \mathrm{B}(f, r_{(1-\gamma)/6})$ for some $\ell \in \mathbb{N}$,

$$
\begin{aligned}
&\hat{\Gamma}_s^{(k)}(X_m, -f(X_m), W_2, \mathcal{H}) \\
&= \frac{1}{M_s^{(k)}(\mathcal{H})} \sum_{i=1}^{s^3} \mathbb{1}_{\mathcal{S}^{k-1}(\mathcal{H}[(X_m, f(X_m))])}\left(S_i^{(k)}\right) \mathbb{1}_{\mathcal{S}^{k-1}(\mathcal{H})}\left(S_i^{(k)}\right) \\
&\leq \frac{1}{\left|\left\{S_i^{(k)} : i \leq s^3\right\} \cap \partial_{\mathcal{H}}^{k-1} f\right|} \sum_{i=1}^{s^3} \mathbb{1}_{\mathcal{S}^{k-1}(V_m^\star)}\left(S_i^{(k)}\right) \mathbb{1}_{\mathcal{S}^{k-1}\left(\mathrm{B}(f, r_{(1-\gamma)/6})\right)}\left(S_i^{(k)}\right) \qquad \text{(monotonicity)} \\
&\leq \frac{1}{\left|\left\{S_i^{(k)} : i \leq s^3\right\} \cap \partial_{\mathcal{H}}^{k-1} f\right|} \sum_{i=1}^{s^3} \mathbb{1}_{\partial_{V_m^\star}^{k-1} f}\left(S_i^{(k)}\right) \mathbb{1}_{\mathcal{S}^{k-1}\left(\mathrm{B}(f, r_{(1-\gamma)/6})\right)}\left(S_i^{(k)}\right) \qquad \text{(monotonicity)} \\
&= \frac{1}{\left|\left\{S_i^{(k)} : i \leq s^3\right\} \cap \partial_{\mathbb{C}}^{k-1} f\right|} \sum_{i=1}^{s^3} \mathbb{1}_{\partial_{\mathbb{C}}^{k-1} f}\left(S_i^{(k)}\right) \mathbb{1}_{\mathcal{S}^{k-1}\left(\mathrm{B}(f, r_{(1-\gamma)/6})\right)}\left(S_i^{(k)}\right) \qquad \text{(Lemma 35)} \\
&\leq \frac{3}{2M_s^{(k)}(\mathrm{B}(f, r_{(1-\gamma)/6}))} \sum_{i=1}^{s^3} \mathbb{1}_{\partial_{\mathbb{C}}^{k-1} f}\left(S_i^{(k)}\right) \mathbb{1}_{\mathcal{S}^{k-1}\left(\mathrm{B}(f, r_{(1-\gamma)/6})\right)}\left(S_i^{(k)}\right). \qquad \text{(Lemma 40)}
\end{aligned}
$$

For brevity, let $\hat{\Gamma}$ denote this last quantity, and let $M_{ks} = M_s^{(k)}\left(\mathrm{B}\left(f, r_{(1-\gamma)/6}\right)\right)$. By Hoeffding's inequality, we have

$$
\mathbb{P}\left((2/3)\hat{\Gamma} > \mathcal{P}^{k-1}\left(\partial_{\mathbb{C}}^{k-1} f \middle| \mathcal{S}^{k-1}\left(\mathrm{B}\left(f, r_{(1-\gamma)/6}\right)\right)\right) + M_{ks}^{-1/3} \middle| M_{ks}\right) \leq \exp\left\{-2M_{ks}^{1/3}\right\}.
$$

Thus, by Lemmas 36, 39, and 40,

$$
\begin{aligned}
&\mathbb{P}\left(\left\{(2/3)\hat{\Gamma}_s^{(k)}(X_m, -f(X_m), W_2, \mathcal{H}) > q\left(r_{(1-\gamma)/6}\right) + \tilde{M}(s)^{-1/3}\right\} \cap H_\tau^{(i)} \cap G_\tau^{(i)}\right) \\
&\leq \mathbb{P}\left(\left\{(2/3)\hat{\Gamma} > \mathcal{P}^{k-1}\left(\partial_{\mathbb{C}}^{k-1} f \middle| \mathcal{S}^{k-1}\left(\mathrm{B}\left(f, r_{(1-\gamma)/6}\right)\right)\right) + \tilde{M}(s)^{-1/3}\right\} \cap H_\tau^{(i)}\right) \\
&\leq \mathbb{P}\left(\left\{(2/3)\hat{\Gamma} > \mathcal{P}^{k-1}\left(\partial_{\mathbb{C}}^{k-1} f \middle| \mathcal{S}^{k-1}\left(\mathrm{B}\left(f, r_{(1-\gamma)/6}\right)\right)\right) + M_{ks}^{-1/3}\right\} \cap \left\{M_{ks} \geq \tilde{M}(s)\right\}\right) \\
&= \mathbb{E}\left[\mathbb{P}\left((2/3)\hat{\Gamma} > \mathcal{P}^{k-1}\left(\partial_{\mathbb{C}}^{k-1} f \middle| \mathcal{S}^{k-1}\left(\mathrm{B}\left(f, r_{(1-\gamma)/6}\right)\right)\right) + M_{ks}^{-1/3} \middle| M_{ks}\right) \mathbb{1}_{[\tilde{M}(s), \infty)}(M_{ks})\right] \\
&\leq \mathbb{E}\left[\exp\left\{-2M_{ks}^{1/3}\right\} \mathbb{1}_{[\tilde{M}(s), \infty)}(M_{ks})\right] \leq \exp\left\{-2\tilde{M}(s)^{1/3}\right\}.
\end{aligned}
$$

Thus, there is an event $H_\tau^{(ii)}(k,s)$ with $\mathbb{P}\left(H_\tau^{(i)} \cap G_\tau^{(ii)} \setminus H_\tau^{(ii)}(k,s)\right) \leq \exp\left\{-2\tilde{M}(s)^{1/3}\right\}$ such that

$$
\hat{\Gamma}_s^{(k)}(X_m, -f(X_m), W_2, \mathcal{H}) \leq (3/2)\left(q\left(r_{(1-\gamma)/6}\right) + \tilde{M}(s)^{-1/3}\right)
$$

holds for these particular values of k and s.

To extend to the full range of values, we simply take $H_\tau^{(ii)} = G_\tau^{(i)} \cap \bigcap_{s\geq\tau}\bigcap_{k\leq\tilde{d}_f} H_\tau^{(ii)}(k,s)$. Since $\tau \geq (2/\tilde{\delta}_f)^{1/3}$, we have $\tilde{M}(\tau) \geq 1$, so a union bound implies

$$\begin{aligned}
\mathbb{P}\left(H_\tau^{(i)} \cap G_\tau^{(i)} \setminus H_\tau^{(ii)}\right) &\leq \sum_{s\geq\tau} \tilde{d}_f \cdot \exp\left\{-2\tilde{M}(s)^{1/3}\right\} \\
&\leq \tilde{d}_f \cdot \left(\exp\left\{-2\tilde{M}(\tau)^{1/3}\right\} + \int_\tau^\infty \exp\left\{-2\tilde{M}(x)^{1/3}\right\} \mathrm{d}x\right) \\
&= \tilde{d}_f \left(1 + 2^{-2/3}\tilde{\delta}_f^{-1/3}\right) \cdot \exp\left\{-2\tilde{M}(\tau)^{1/3}\right\} \leq 2\tilde{d}_f\tilde{\delta}_f^{-1/3} \cdot \exp\left\{-2\tilde{M}(\tau)^{1/3}\right\}.
\end{aligned}$$

Then Lemma 40 and a union bound imply

$$\begin{aligned}
\mathbb{P}\left(H_\tau^{(i)} \setminus H_\tau^{(ii)}\right) &\leq 2\tilde{d}_f\tilde{\delta}_f^{-1/3} \cdot \exp\left\{-2\tilde{M}(\tau)^{1/3}\right\} + 121\tilde{d}_f\tilde{\delta}_f^{-1} \cdot \exp\left\{-\tilde{M}(\tau)/60\right\} \\
&\leq 123\tilde{d}_f\tilde{\delta}_f^{-1} \cdot \exp\left\{-\tilde{M}(\tau)^{1/3}/60\right\}.
\end{aligned}$$

On $H_\tau^{(i)} \cap H_\tau^{(ii)}$, every such s, m, ℓ, k and $\mathcal{H}$ satisfy

$$\begin{aligned}
\hat{\Gamma}_s^{(k)}(X_m, -f(X_m), W_2, \mathcal{H}) &\leq (3/2)\left(q(r_{(1-\gamma)/6}) + \tilde{M}(s)^{-1/3}\right) \\
&< (3/2)\left((1-\gamma)/6 + (1-\gamma)/6\right) = (1-\gamma)/2, \qquad (28)
\end{aligned}$$

where the second inequality follows by definition of $r_{(1-\gamma)/6}$ and $s \geq \tau \geq \tau^*$.

If $\hat{\Delta}_s^{(k)}(X_m, W_2, \mathcal{H}) < \gamma$, then

$$1 - \gamma < 1 - \hat{\Delta}_s^{(k)}(X_m, W_2, \mathcal{H}) = \frac{1}{M_s^{(k)}(\mathcal{H})} \sum_{i=1}^{s^3} \mathbb{1}_{\mathcal{S}^{k-1}(\mathcal{H})}\left(S_i^{(k)}\right) \mathbb{1}_{\mathcal{S}^k(\mathcal{H})}\left(S_i^{(k)} \cup \{X_m\}\right). \qquad (29)$$

Finally, noting that we always have

$$\mathbb{1}_{\mathcal{S}^k(\mathcal{H})}\left(S_i^{(k)} \cup \{X_m\}\right) \leq \mathbb{1}_{\mathcal{S}^{k-1}(\mathcal{H}[(X_m, f(X_m))])}\left(S_i^{(k)}\right) + \mathbb{1}_{\mathcal{S}^{k-1}(\mathcal{H}[(X_m, -f(X_m))])}\left(S_i^{(k)}\right),$$

we have that, on the event $H_\tau^{(i)} \cap H_\tau^{(ii)}$, if $\hat{\Delta}_s^{(k)}(X_m, W_2, \mathcal{H}) < \gamma$, then

$$\begin{aligned}
&\hat{\Gamma}_s^{(k)}(X_m, -f(X_m), W_2, \mathcal{H}) \\
&< (1-\gamma)/2 = -(1-\gamma)/2 + (1-\gamma) && \text{by (28)} \\
&< -(1-\gamma)/2 + \frac{1}{M_s^{(k)}(\mathcal{H})} \sum_{i=1}^{s^3} \mathbb{1}_{\mathcal{S}^{k-1}(\mathcal{H})}\left(S_i^{(k)}\right) \mathbb{1}_{\mathcal{S}^k(\mathcal{H})}\left(S_i^{(k)} \cup \{X_m\}\right) && \text{by (29)} \\
&\leq -(1-\gamma)/2 + \frac{1}{M_s^{(k)}(\mathcal{H})} \sum_{i=1}^{s^3} \mathbb{1}_{\mathcal{S}^{k-1}(\mathcal{H})}\left(S_i^{(k)}\right) \mathbb{1}_{\mathcal{S}^{k-1}(\mathcal{H}[(X_m, f(X_m))])}\left(S_i^{(k)}\right) \\
&\qquad + \frac{1}{M_s^{(k)}(\mathcal{H})} \sum_{i=1}^{s^3} \mathbb{1}_{\mathcal{S}^{k-1}(\mathcal{H})}\left(S_i^{(k)}\right) \mathbb{1}_{\mathcal{S}^{k-1}(\mathcal{H}[(X_m, -f(X_m))])}\left(S_i^{(k)}\right) \\
&= -(1-\gamma)/2 + \hat{\Gamma}_s^{(k)}(X_m, -f(X_m), W_2, \mathcal{H}) + \hat{\Gamma}_s^{(k)}(X_m, f(X_m), W_2, \mathcal{H}) \\
&< \hat{\Gamma}_s^{(k)}(X_m, f(X_m), W_2, \mathcal{H}). && \text{by (28)}
\end{aligned}$$

The final claim in the lemma statement is then implied by Lemma 29, since we have $V_\ell^\star \subseteq V_\tau^\star \subseteq \mathrm{B}\left(f, \phi(\tau;\delta)\right) \subseteq \mathrm{B}\left(f, r_{(1-\gamma)/6}\right)$ on $H_\tau(\delta)$. ■

For any $k, \ell, m \in \mathbb{N}$, and any $x \in \mathcal{X}$, define

$$\hat{p}_x(k,\ell,m) = \hat{\Delta}_m^{(k)}\left(x, W_2, V_\ell^\star\right)$$
$$p_x(k,\ell) = \mathcal{P}^{k-1}\left(S \in \mathcal{X}^{k-1} : S \cup \{x\} \in \mathcal{S}^k\left(V_\ell^\star\right) \middle| \mathcal{S}^{k-1}\left(V_\ell^\star\right)\right).$$

Lemma 42 *For any $\xi \in (0,1)$, there is a $(\mathbb{C}, \mathcal{P}, f, \xi)$-dependent constant $c^{(iii)}(\xi) \in (0,\infty)$ such that, for any $\tau \in \mathbb{N}$, there is an event $H_\tau^{(iii)}(\xi)$ with*

$$\mathbb{P}\left(H_\tau^{(i)} \setminus H_\tau^{(iii)}(\xi)\right) \leq c^{(iii)}(\xi) \cdot \exp\left\{-\xi^2 \tilde{M}(\tau)\right\}$$

such that on $H_\tau^{(i)} \cap H_\tau^{(iii)}(\xi)$, $\forall k, \ell, m \in \mathbb{N}$ with $\tau \leq \ell \leq m$ and $k \leq \tilde{d}_f$, for any $x \in \mathcal{X}$,

$$\mathcal{P}\left(x : |p_x(k,\ell) - \hat{p}_x(k,\ell,m)| > \xi\right) \leq \exp\left\{-\xi^2 \tilde{M}(m)\right\}.$$

Proof Fix any $k, \ell, m \in \mathbb{N}$ with $\tau \leq \ell \leq m$ and $k \leq \tilde{d}_f$. Recall our convention that $\mathcal{X}^0 = \{\varnothing\}$ and $\mathcal{P}^0\left(\mathcal{X}^0\right) = 1$; thus, if $k = 1$, $\hat{p}_x(k,\ell,m) = \mathbb{1}_{\mathrm{DIS}\left(V_\ell^\star\right)}(x) = \mathbb{1}_{\mathcal{S}^1\left(V_\ell^\star\right)}(x) = p_x(k,\ell)$, so the result clearly holds for $k = 1$.

For the remaining case, suppose $2 \leq k \leq \tilde{d}_f$. To simplify notation, let $\tilde{m} = M_m^{(k)}\left(V_\ell^\star\right)$, $X = X_{\ell+1}$, $p_x = p_x(k,\ell)$ and $\hat{p}_x = \hat{p}_x(k,\ell,m)$. Consider the event

$$H^{(iii)}(k,\ell,m,\xi) = \left\{\mathcal{P}\left(x : |p_x - \hat{p}_x| > \xi\right) \leq \exp\left\{-\xi^2 \tilde{M}(m)\right\}\right\}.$$

We have

$$\mathbb{P}\left(H_\tau^{(i)} \setminus H^{(iii)}(k,\ell,m,\xi) \middle| V_\ell^\star\right) \tag{30}$$
$$\leq \mathbb{P}\left(\left\{\tilde{m} \geq \tilde{M}(m)\right\} \setminus H^{(iii)}(k,\ell,m,\xi) \middle| V_\ell^\star\right) \text{ (by Lemma 39)}$$
$$= \mathbb{P}\left(\left\{\tilde{m} \geq \tilde{M}(m)\right\} \cap \left\{\mathbb{P}\left(e^{s\tilde{m}|p_X - \hat{p}_X|} > e^{s\tilde{m}\xi} \middle| W_2, V_\ell^\star\right) > e^{-\xi^2 \tilde{M}(m)}\right\} \middle| V_\ell^\star\right), \tag{31}$$

for any value $s > 0$. Proceeding as in Chernoff's bounding technique, by Markov's inequality (31) is at most

$$\mathbb{P}\left(\left\{\tilde{m} \geq \tilde{M}(m)\right\} \cap \left\{e^{-s\tilde{m}\xi}\mathbb{E}\left[e^{s\tilde{m}|p_X - \hat{p}_X|} \middle| W_2, V_\ell^\star\right] > e^{-\xi^2 \tilde{M}(m)}\right\} \middle| V_\ell^\star\right)$$
$$\leq \mathbb{P}\left(\left\{\tilde{m} \geq \tilde{M}(m)\right\} \cap \left\{e^{-s\tilde{m}\xi}\mathbb{E}\left[e^{s\tilde{m}(p_X - \hat{p}_X)} + e^{s\tilde{m}(\hat{p}_X - p_X)} \middle| W_2, V_\ell^\star\right] > e^{-\xi^2 \tilde{M}(m)}\right\} \middle| V_\ell^\star\right)$$
$$= \mathbb{E}\left[\mathbb{1}_{[\tilde{M}(m),\infty)}(\tilde{m})\, \mathbb{P}\left(e^{-s\tilde{m}\xi}\mathbb{E}\left[e^{s\tilde{m}(p_X - \hat{p}_X)} + e^{s\tilde{m}(\hat{p}_X - p_X)} \middle| W_2, V_\ell^\star\right] > e^{-\xi^2 \tilde{M}(m)} \middle| \tilde{m}, V_\ell^\star\right) \middle| V_\ell^\star\right]$$

By Markov's inequality, this is at most

$$\begin{aligned}
&\mathbb{E}\left[\mathbb{1}_{[\tilde{M}(m),\infty)}(\tilde{m})\,e^{\xi^2\tilde{M}(m)}\mathbb{E}\left[e^{-s\tilde{m}\xi}\mathbb{E}\left[e^{s\tilde{m}(p_X-\hat{p}_X)}+e^{s\tilde{m}(\hat{p}_X-p_X)}\middle|W_2,V_\ell^\star\right]\middle|\tilde{m},V_\ell^\star\right]\middle|V_\ell^\star\right]\\
&=\mathbb{E}\left[\mathbb{1}_{[\tilde{M}(m),\infty)}(\tilde{m})\,e^{\xi^2\tilde{M}(m)}e^{-s\tilde{m}\xi}\mathbb{E}\left[e^{s\tilde{m}(p_X-\hat{p}_X)}+e^{s\tilde{m}(\hat{p}_X-p_X)}\middle|\tilde{m},V_\ell^\star\right]\middle|V_\ell^\star\right]\\
&=\mathbb{E}\left[\mathbb{1}_{[\tilde{M}(m),\infty)}(\tilde{m})\,e^{\xi^2\tilde{M}(m)}e^{-s\tilde{m}\xi}\mathbb{E}\left[\mathbb{E}\left[e^{s\tilde{m}(p_X-\hat{p}_X)}+e^{s\tilde{m}(\hat{p}_X-p_X)}\middle|X,\tilde{m},V_\ell^\star\right]\middle|\tilde{m},V_\ell^\star\right]\middle|V_\ell^\star\right]. \qquad (32)
\end{aligned}$$

The conditional distribution of $\tilde{m}\hat{p}_X$ given $(X,\tilde{m},V_\ell^\star)$ is Binomial$(\tilde{m},p_X)$, so letting $\left\{\mathbf{B}_j(p_X)\right\}_{j=1}^\infty$ denote a sequence of random variables, conditionally independent given $(X,\tilde{m},V_\ell^\star)$, with the conditional distribution of each $\mathbf{B}_j(p_X)$ being Bernoulli(p_X) given $(X,\tilde{m},V_\ell^\star)$, we have

$$\begin{aligned}
&\mathbb{E}\left[e^{s\tilde{m}(p_X-\hat{p}_X)}+e^{s\tilde{m}(\hat{p}_X-p_X)}\middle|X,\tilde{m},V_\ell^\star\right]\\
&=\mathbb{E}\left[e^{s\tilde{m}(p_X-\hat{p}_X)}\middle|X,\tilde{m},V_\ell^\star\right]+\mathbb{E}\left[e^{s\tilde{m}(\hat{p}_X-p_X)}\middle|X,\tilde{m},V_\ell^\star\right]\\
&=\mathbb{E}\left[\prod_{i=1}^{\tilde{m}}e^{s(p_X-\mathbf{B}_i(p_X))}\middle|X,\tilde{m},V_\ell^\star\right]+\mathbb{E}\left[\prod_{i=1}^{\tilde{m}}e^{s(\mathbf{B}_i(p_X)-p_X)}\middle|X,\tilde{m},V_\ell^\star\right]\\
&=\mathbb{E}\left[e^{s(p_X-\mathbf{B}_1(p_X))}\middle|X,\tilde{m},V_\ell^\star\right]^{\tilde{m}}+\mathbb{E}\left[e^{s(\mathbf{B}_1(p_X)-p_X)}\middle|X,\tilde{m},V_\ell^\star\right]^{\tilde{m}}. \qquad (33)
\end{aligned}$$

It is known that for $\mathbf{B}\sim$ Bernoulli(p), $\mathbb{E}\left[e^{s(\mathbf{B}-p)}\right]$ and $\mathbb{E}\left[e^{s(p-\mathbf{B})}\right]$ are at most $e^{s^2/8}$ (see, e.g., Lemma 8.1 of Devroye, Györfi, and Lugosi, 1996). Thus, taking $s=4\xi$, (33) is at most $2e^{2\tilde{m}\xi^2}$, and (32) is at most

$$\begin{aligned}
\mathbb{E}\left[\mathbb{1}_{[\tilde{M}(m),\infty)}(\tilde{m})\,2e^{\xi^2\tilde{M}(m)}e^{-4\tilde{m}\xi^2}e^{2\tilde{m}\xi^2}\middle|V_\ell^\star\right]&=\mathbb{E}\left[\mathbb{1}_{[\tilde{M}(m),\infty)}(\tilde{m})\,2e^{\xi^2\tilde{M}(m)}e^{-2\tilde{m}\xi^2}\middle|V_\ell^\star\right]\\
&\le 2\exp\left\{-\xi^2\tilde{M}(m)\right\}.
\end{aligned}$$

Since this bound holds for (30), the law of total probability implies

$$\mathbb{P}\left(H_\tau^{(i)}\setminus H^{(iii)}(k,\ell,m,\xi)\right)=\mathbb{E}\left[\mathbb{P}\left(H_\tau^{(i)}\setminus H^{(iii)}(k,\ell,m,\xi)\middle|V_\ell^\star\right)\right]\le 2\cdot\exp\left\{-\xi^2\tilde{M}(m)\right\}.$$

Defining $H_\tau^{(iii)}(\xi) = \bigcap_{\ell \geq \tau} \bigcap_{m \geq \ell} \bigcap_{k=2}^{\tilde{d}_f} H^{(iii)}(k,\ell,m,\xi)$, we have the required property for the claimed ranges of k, ℓ and m, and a union bound implies

$$\begin{aligned}
&\mathbb{P}\left(H_\tau^{(i)} \setminus H_\tau^{(iii)}(\xi)\right) \leq \sum_{\ell \geq \tau} \sum_{m \geq \ell} 2\tilde{d}_f \cdot \exp\left\{-\xi^2 \tilde{M}(m)\right\} \\
&\leq 2\tilde{d}_f \cdot \sum_{\ell \geq \tau} \left(\exp\left\{-\xi^2 \tilde{M}(\ell)\right\} + \int_{\ell^3}^{\infty} \exp\left\{-x\xi^2 \tilde{\delta}_f/2\right\} \mathrm{d}x\right) \\
&= 2\tilde{d}_f \cdot \sum_{\ell \geq \tau} \left(1 + 2\xi^{-2}\tilde{\delta}_f^{-1}\right) \cdot \exp\left\{-\xi^2 \tilde{M}(\ell)\right\} \\
&\leq 2\tilde{d}_f \cdot \left(1 + 2\xi^{-2}\tilde{\delta}_f^{-1}\right) \cdot \left(\exp\left\{-\xi^2 \tilde{M}(\tau)\right\} + \int_{\tau^3}^{\infty} \exp\left\{-x\xi^2 \tilde{\delta}_f/2\right\} \mathrm{d}x\right) \\
&= 2\tilde{d}_f \cdot \left(1 + 2\xi^{-2}\tilde{\delta}_f^{-1}\right)^2 \cdot \exp\left\{-\xi^2 \tilde{M}(\tau)\right\} \\
&\leq 18\tilde{d}_f \xi^{-4} \tilde{\delta}_f^{-2} \cdot \exp\left\{-\xi^2 \tilde{M}(\tau)\right\}.
\end{aligned}$$

■

For $k, \ell, m \in \mathbb{N}$ and $\xi \in (0,1)$, define

$$p_\xi(k,\ell,m) = \mathcal{P}\left(x : \hat{p}_x(k,\ell,m) \geq \xi\right). \tag{34}$$

Lemma 43 *For any $\alpha, \xi, \delta \in (0,1)$, $\beta \in \left(0, 1-\sqrt{\alpha}\right]$, and integer $\tau \geq \tau(\beta;\delta)$, on $H_\tau(\delta) \cap H_\tau^{(i)} \cap H_\tau^{(iii)}(\beta\xi)$, for any $k, \ell, \ell', m \in \mathbb{N}$ with $\tau \leq \ell \leq \ell' \leq m$ and $k \leq \tilde{d}_f$,*

$$p_\xi(k,\ell',m) \leq \mathcal{P}\left(x : p_x(k,\ell) \geq \alpha\xi\right) + \exp\left\{-\beta^2\xi^2\tilde{M}(m)\right\}. \tag{35}$$

Proof Fix any $\alpha, \xi, \delta \in (0,1)$, $\beta \in \left(0, 1-\sqrt{\alpha}\right]$, $\tau, k, \ell, \ell', m \in \mathbb{N}$ with $\tau(\beta;\delta) \leq \tau \leq \ell \leq \ell' \leq m$ and $k \leq \tilde{d}_f$.

If $k = 1$, the result clearly holds. In particular, we have

$$p_\xi(1,\ell',m) = \mathcal{P}\left(\mathrm{DIS}\left(V_{\ell'}^\star\right)\right) \leq \mathcal{P}\left(\mathrm{DIS}\left(V_\ell^\star\right)\right) = \mathcal{P}\left(x : p_x(1,\ell) \geq \alpha\xi\right).$$

Otherwise, suppose $2 \leq k \leq \tilde{d}_f$. By a union bound,

$$\begin{aligned}
p_\xi(k,\ell',m) &= \mathcal{P}\left(x : \hat{p}_x(k,\ell',m) \geq \xi\right) \\
&\leq \mathcal{P}\left(x : p_x(k,\ell') \geq \sqrt{\alpha}\xi\right) + \mathcal{P}\left(x : \left|p_x(k,\ell') - \hat{p}_x(k,\ell',m)\right| > (1-\sqrt{\alpha})\xi\right).
\end{aligned} \tag{36}$$

Since

$$\mathcal{P}\left(x : \left|p_x(k,\ell') - \hat{p}_x(k,\ell',m)\right| > (1-\sqrt{\alpha})\xi\right) \leq \mathcal{P}\left(x : \left|p_x(k,\ell') - \hat{p}_x(k,\ell',m)\right| > \beta\xi\right),$$

Lemma 42 implies that, on $H_\tau^{(i)} \cap H_\tau^{(iii)}(\beta\xi)$,

$$\mathcal{P}\left(x : \left|p_x(k,\ell') - \hat{p}_x(k,\ell',m)\right| > (1-\sqrt{\alpha})\xi\right) \leq \exp\left\{-\beta^2\xi^2\tilde{M}(m)\right\}. \tag{37}$$

It remains only to examine the first term on the right side of (36). For this, if $\mathcal{P}^{k-1}\left(\mathcal{S}^{k-1}\left(V_{\ell'}^\star\right)\right)=0$, then the first term is 0 by our aforementioned convention, and thus (35) holds; otherwise, since

$$\forall x\in\mathcal{X},\left\{S\in\mathcal{X}^{k-1}:S\cup\{x\}\in\mathcal{S}^k\left(V_{\ell'}^\star\right)\right\}\subseteq\mathcal{S}^{k-1}\left(V_{\ell'}^\star\right),$$

we have

$$\begin{aligned}&\mathcal{P}\left(x:p_x(k,\ell')\geq\sqrt{\alpha}\xi\right)=\mathcal{P}\left(x:\mathcal{P}^{k-1}\left(S\in\mathcal{X}^{k-1}:S\cup\{x\}\in\mathcal{S}^k\left(V_{\ell'}^\star\right)\middle|\mathcal{S}^{k-1}\left(V_{\ell'}^\star\right)\right)\geq\sqrt{\alpha}\xi\right)\\&=\mathcal{P}\left(x:\mathcal{P}^{k-1}\left(S\in\mathcal{X}^{k-1}:S\cup\{x\}\in\mathcal{S}^k\left(V_{\ell'}^\star\right)\right)\geq\sqrt{\alpha}\xi\mathcal{P}^{k-1}\left(\mathcal{S}^{k-1}\left(V_{\ell'}^\star\right)\right)\right).\end{aligned}\tag{38}$$

By Lemma 35 and monotonicity, on $H_\tau^{(i)}\subseteq H'$, (38) is at most

$$\mathcal{P}\left(x:\mathcal{P}^{k-1}\left(S\in\mathcal{X}^{k-1}:S\cup\{x\}\in\mathcal{S}^k\left(V_{\ell'}^\star\right)\right)\geq\sqrt{\alpha}\xi\mathcal{P}^{k-1}\left(\partial_{\mathbb{C}}^{k-1}f\right)\right),$$

and monotonicity implies this is at most

$$\mathcal{P}\left(x:\mathcal{P}^{k-1}\left(S\in\mathcal{X}^{k-1}:S\cup\{x\}\in\mathcal{S}^k\left(V_\ell^\star\right)\right)\geq\sqrt{\alpha}\xi\mathcal{P}^{k-1}\left(\partial_{\mathbb{C}}^{k-1}f\right)\right).\tag{39}$$

By Lemma 36, for $\tau\geq\tau(\beta;\delta)$, on $H_\tau(\delta)\cap H_\tau^{(i)}$,

$$\mathcal{P}^{k-1}\left(\partial_{\mathbb{C}}^{k-1}f\middle|\mathcal{S}^{k-1}\left(V_\ell^\star\right)\right)\leq q(\phi(\tau;\delta))<\beta\leq1-\sqrt{\alpha},$$

which implies

$$\begin{aligned}\mathcal{P}^{k-1}\left(\partial_{\mathbb{C}}^{k-1}f\right)&\geq\mathcal{P}^{k-1}\left(\partial_{\mathbb{C}}^{k-1}f\cap\mathcal{S}^{k-1}\left(V_\ell^\star\right)\right)\\&=\left(1-\mathcal{P}^{k-1}\left(\partial_{\mathbb{C}}^{k-1}f\middle|\mathcal{S}^{k-1}\left(V_\ell^\star\right)\right)\right)\mathcal{P}^{k-1}\left(\mathcal{S}^{k-1}\left(V_\ell^\star\right)\right)\geq\sqrt{\alpha}\mathcal{P}^{k-1}\left(\mathcal{S}^{k-1}\left(V_\ell^\star\right)\right).\end{aligned}$$

Altogether, for $\tau\geq\tau(\beta;\delta)$, on $H_\tau(\delta)\cap H_\tau^{(i)}$, (39) is at most

$$\mathcal{P}\left(x:\mathcal{P}^{k-1}\left(S\in\mathcal{X}^{k-1}:S\cup\{x\}\in\mathcal{S}^k\left(V_\ell^\star\right)\right)\geq\alpha\xi\mathcal{P}^{k-1}\left(\mathcal{S}^{k-1}\left(V_\ell^\star\right)\right)\right)=\mathcal{P}\left(x:p_x(k,\ell)\geq\alpha\xi\right),$$

which, combined with (36) and (37), establishes (35). ■

Lemma 44 *There are events* $\left\{H_\tau^{(iv)}:\tau\in\mathbb{N}\right\}$ *with*

$$\mathbb{P}\left(H_\tau^{(iv)}\right)\geq1-3\tilde{d}_f\cdot\exp\{-2\tau\}$$

s.t. for any $\xi\in(0,\gamma/16]$, $\delta\in(0,1)$, *letting* $\tau^{(iv)}(\xi;\delta)=\max\left\{\tau(4\xi/\gamma;\delta),\left(\frac{4}{\tilde{\delta}_f\xi^2}\ln\left(\frac{4}{\tilde{\delta}_f\xi^2}\right)\right)^{1/3}\right\}$, *for any integer* $\tau\geq\tau^{(iv)}(\xi;\delta)$, *on* $H_\tau(\delta)\cap H_\tau^{(i)}\cap H_\tau^{(iii)}(\xi)\cap H_\tau^{(iv)}$, $\forall k\in\left\{1,\ldots,\tilde{d}_f\right\}$, $\forall\ell\in\mathbb{N}$ *with* $\ell\geq\tau$,

$$\mathcal{P}\left(x:p_x(k,\ell)\geq\gamma/2\right)+\exp\left\{-\gamma^2\tilde{M}(\ell)/256\right\}\leq\hat{\Delta}_\ell^{(k)}\left(W_1,W_2,V_\ell^\star\right)\tag{40}$$

$$\leq\mathcal{P}\left(x:p_x(k,\ell)\geq\gamma/8\right)+4\ell^{-1}.\tag{41}$$

Proof For any $k,\ell \in \mathbb{N}$, by Hoeffding's inequality and the law of total probability, on an event $G^{(iv)}(k,\ell)$ with $\mathbb{P}\left(G^{(iv)}(k,\ell)\right) \geq 1-2\exp\{-2\ell\}$, we have

$$\left| p_{\gamma/4}(k,\ell,\ell) - \ell^{-3}\sum_{i=1}^{\ell^3} \mathbb{1}_{[\gamma/4,\infty)}\left(\hat{\Delta}_\ell^{(k)}(w_i, W_2, V_\ell^\star)\right)\right| \leq \ell^{-1}. \tag{42}$$

Define the event $H_\tau^{(iv)} = \bigcap_{\ell\geq\tau}\bigcap_{k=1}^{\tilde{d}_f} G^{(iv)}(k,\ell)$. By a union bound, we have

$$\begin{aligned}1-\mathbb{P}\left(H_\tau^{(iv)}\right) &\leq 2\tilde{d}_f \cdot \sum_{\ell\geq\tau}\exp\{-2\ell\} \\ &\leq 2\tilde{d}_f\cdot\left(\exp\{-2\tau\} + \int_\tau^\infty \exp\{-2x\}\,\mathrm{d}x\right) = 3\tilde{d}_f\cdot\exp\{-2\tau\}.\end{aligned}$$

Now fix any $\ell\geq\tau$ and $k\in\left\{1,\ldots,\tilde{d}_f\right\}$. By a union bound,

$$\mathcal{P}\left(x : p_x(k,\ell)\geq\gamma/2\right) \leq \mathcal{P}\left(x:\hat{p}_x(k,\ell,\ell)\geq\gamma/4\right) + \mathcal{P}\left(x:|p_x(k,\ell)-\hat{p}_x(k,\ell,\ell)|>\gamma/4\right). \tag{43}$$

By Lemma 42, on $H_\tau^{(i)}\cap H_\tau^{(iii)}(\xi)$,

$$\mathcal{P}\left(x:|p_x(k,\ell)-\hat{p}_x(k,\ell,\ell)|>\gamma/4\right) \leq \mathcal{P}\left(x:|p_x(k,\ell)-\hat{p}_x(k,\ell,\ell)|>\xi\right) \leq \exp\left\{-\xi^2\tilde{M}(\ell)\right\}. \tag{44}$$

Also, on $H_\tau^{(iv)}$, (42) implies

$$\begin{aligned}\mathcal{P}\left(x:\hat{p}_x(k,\ell,\ell)\geq\gamma/4\right) &= p_{\gamma/4}(k,\ell,\ell) \\ &\leq \ell^{-1} + \ell^{-3}\sum_{i=1}^{\ell^3}\mathbb{1}_{[\gamma/4,\infty)}\left(\hat{\Delta}_\ell^{(k)}(w_i,W_2,V_\ell^\star)\right) \\ &= \hat{\Delta}_\ell^{(k)}(W_1,W_2,V_\ell^\star) - \ell^{-1}.\end{aligned} \tag{45}$$

Combining (43) with (44) and (45) yields

$$\mathcal{P}\left(x:p_x(k,\ell)\geq\gamma/2\right) \leq \hat{\Delta}_\ell^{(k)}(W_1,W_2,V_\ell^\star) - \ell^{-1} + \exp\left\{-\xi^2\tilde{M}(\ell)\right\}. \tag{46}$$

For $\tau\geq\tau^{(iv)}(\xi;\delta)$, $\exp\left\{-\xi^2\tilde{M}(\ell)\right\} - \ell^{-1} \leq -\exp\left\{-\gamma^2\tilde{M}(\ell)/256\right\}$, so that (46) implies the first inequality of the lemma: namely (40).

For the second inequality (i.e., (41)), on $H_\tau^{(iv)}$, (42) implies we have

$$\hat{\Delta}_\ell^{(k)}(W_1,W_2,V_\ell^\star) \leq p_{\gamma/4}(k,\ell,\ell) + 3\ell^{-1}. \tag{47}$$

Also, by Lemma 43 (with $\alpha=1/2$, $\zeta=\gamma/4$, $\beta=\xi/\zeta<1-\sqrt{\alpha}$), for $\tau\geq\tau^{(iv)}(\xi;\delta)$, on $H_\tau(\delta)\cap H_\tau^{(i)}\cap H_\tau^{(iii)}(\xi)$,

$$p_{\gamma/4}(k,\ell,\ell) \leq \mathcal{P}\left(x:p_x(k,\ell)\geq\gamma/8\right) + \exp\left\{-\xi^2\tilde{M}(\ell)\right\}. \tag{48}$$

Thus, combining (47) with (48) yields

$$\hat{\Delta}_\ell^{(k)}(W_1,W_2,V_\ell^\star) \leq \mathcal{P}\left(x:p_x(k,\ell)\geq\gamma/8\right) + 3\ell^{-1} + \exp\left\{-\xi^2\tilde{M}(\ell)\right\}.$$

For $\tau \geq \tau^{(iv)}(\xi;\delta)$, we have $\exp\left\{-\xi^2\tilde{M}(\ell)\right\} \leq \ell^{-1}$, which establishes (41). ■

For $n \in \mathbb{N}$ and $k \in \{1,\ldots,d+1\}$, define the set

$$\mathcal{U}_n^{(k)} = \left\{m_n+1,\ldots,m_n+\left\lfloor n/\left(6\cdot 2^k \hat{\Delta}_{m_n}^{(k)}(W_1,W_2,V)\right)\right\rfloor\right\},$$

where $m_n = \lfloor n/3 \rfloor$; $\mathcal{U}_n^{(k)}$ represents the set of indices processed in the inner loop of Meta-Algorithm 1 for the specified value of k.

Lemma 45 *There are $(f,\mathbb{C},\mathcal{P},\gamma)$-dependent constants $\hat{c}_1,\hat{c}_2 \in (0,\infty)$ such that, for any $\varepsilon \in (0,1)$ and integer $n \geq \hat{c}_1 \ln(\hat{c}_2/\varepsilon)$, on an event $\hat{H}_n(\varepsilon)$ with*

$$\mathbb{P}(\hat{H}_n(\varepsilon)) \geq 1-(3/4)\varepsilon, \tag{49}$$

we have, for $V = V_{m_n}^\star$,

$$\forall k \in \left\{1,\ldots,\tilde{d}_f\right\}, \left|\left\{m \in \mathcal{U}_n^{(k)} : \hat{\Delta}_m^{(k)}(X_m,W_2,V) \geq \gamma\right\}\right| \leq \left\lfloor n/\left(3\cdot 2^k\right)\right\rfloor, \tag{50}$$

$$\hat{\Delta}_{m_n}^{(\tilde{d}_f)}(W_1,W_2,V) \leq \Delta_n^{(\gamma/8)}(\varepsilon) + 4m_n^{-1}, \tag{51}$$

and $\forall m \in \mathcal{U}_n^{(\tilde{d}_f)}$,

$$\hat{\Delta}_m^{(\tilde{d}_f)}(X_m,W_2,V) < \gamma \Rightarrow \hat{\Gamma}_m^{(\tilde{d}_f)}(X_m,-f(X_m),W_2,V) < \hat{\Gamma}_m^{(\tilde{d}_f)}(X_m,f(X_m),W_2,V). \tag{52}$$

Proof Suppose $n \geq \hat{c}_1 \ln(\hat{c}_2/\varepsilon)$, where

$$\hat{c}_1 = \max\left\{\frac{2^{\tilde{d}_f+12}}{\tilde{\delta}_f\gamma^2}, \frac{24}{r_{(1/16)}}, \frac{24}{r_{(1-\gamma)/6}}, 3\tau^*\right\}$$
$$\text{and } \hat{c}_2 = \max\left\{4\left(c^{(i)}+c^{(ii)}+c^{(iii)}(\gamma/16)+6\tilde{d}_f\right), 4\left(\frac{4e}{r_{(1/16)}}\right)^d, 4\left(\frac{4e}{r_{(1-\gamma)/6}}\right)^d\right\}.$$

In particular, we have chosen $\hat{c}_1$ and $\hat{c}_2$ large enough so that

$$m_n \geq \max\left\{\tau(1/16;\varepsilon/2), \tau^{(iv)}(\gamma/16;\varepsilon/2), \tau((1-\gamma)/6;\varepsilon/2), \tau^*\right\}.$$

We begin with (50). By Lemmas 43 and 44, on the event

$$\hat{H}_n^{(1)}(\varepsilon) = H_{m_n}(\varepsilon/2) \cap H_{m_n}^{(i)} \cap H_{m_n}^{(iii)}(\gamma/16) \cap H_{m_n}^{(iv)},$$

$\forall m \in \mathcal{U}_n^{(k)}, \forall k \in \left\{1,\ldots,\tilde{d}_f\right\}$,

$$\begin{aligned} p_\gamma(k,m_n,m) &\leq \mathcal{P}(x : p_x(k,m_n) \geq \gamma/2) + \exp\left\{-\gamma^2\tilde{M}(m)/256\right\} \\ &\leq \mathcal{P}(x : p_x(k,m_n) \geq \gamma/2) + \exp\left\{-\gamma^2\tilde{M}(m_n)/256\right\} \leq \hat{\Delta}_{m_n}^{(k)}(W_1,W_2,V). \end{aligned} \tag{53}$$

Recall that $\left\{X_m : m \in \mathcal{U}_n^{(k)}\right\}$ is a sample of size $\left\lfloor n/(6\cdot 2^k \hat{\Delta}_{m_n}^{(k)}(W_1,W_2,V))\right\rfloor$, conditionally i.i.d. (given (W_1,W_2,V)) with conditional distributions $\mathcal{P}$. Thus, $\forall k \in \left\{1,\ldots,\tilde{d}_f\right\}$, on $\hat{H}_n^{(1)}(\varepsilon)$,

$$\begin{aligned}
&\mathbb{P}\left(\left|\left\{m \in \mathcal{U}_n^{(k)} : \hat{\Delta}_m^{(k)}(X_m,W_2,V) \geq \gamma\right\}\right| > n/\left(3\cdot 2^k\right) \middle| W_1,W_2,V\right) \\
&\leq \mathbb{P}\left(\left|\left\{m \in \mathcal{U}_n^{(k)} : \hat{\Delta}_m^{(k)}(X_m,W_2,V) \geq \gamma\right\}\right| > 2\left|\mathcal{U}_n^{(k)}\right| \hat{\Delta}_{m_n}^{(k)}(W_1,W_2,V) \middle| W_1,W_2,V\right) \\
&\leq \mathbb{P}\left(\mathbf{B}\left(|\mathcal{U}_n^{(k)}|, \hat{\Delta}_{m_n}^{(k)}(W_1,W_2,V)\right) > 2\left|\mathcal{U}_n^{(k)}\right| \hat{\Delta}_{m_n}^{(k)}(W_1,W_2,V) \middle| W_1,W_2,V\right), \qquad (54)
\end{aligned}$$

where this last inequality follows from (53), and $\mathbf{B}(u,p) \sim \text{Binomial}(u,p)$ is independent from W_1,W_2,V (for any fixed u and p). By a Chernoff bound, (54) is at most

$$\exp\left\{-\left\lfloor n/\left(6\cdot 2^k \hat{\Delta}_{m_n}^{(k)}(W_1,W_2,V)\right)\right\rfloor \hat{\Delta}_{m_n}^{(k)}(W_1,W_2,V)/3\right\} \leq \exp\left\{1 - n/\left(18\cdot 2^k\right)\right\}.$$

By the law of total probability and a union bound, there exists an event $\hat{H}_n^{(2)}$ with

$$\mathbb{P}\left(\hat{H}_n^{(1)}(\varepsilon) \setminus \hat{H}_n^{(2)}\right) \leq \tilde{d}_f \cdot \exp\left\{1 - n/\left(18\cdot 2^{\tilde{d}_f}\right)\right\}$$

such that, on $\hat{H}_n^{(1)}(\varepsilon) \cap \hat{H}_n^{(2)}$, (50) holds.

Next, by Lemma 44, on $\hat{H}_n^{(1)}(\varepsilon)$,

$$\hat{\Delta}_{m_n}^{(\tilde{d}_f)}(W_1,W_2,V) \leq \mathcal{P}\left(x : p_x\left(\tilde{d}_f, m_n\right) \geq \gamma/8\right) + 4m_n^{-1},$$

and by Lemma 38, on $\hat{H}_n^{(1)}(\varepsilon)$, this is at most $\Delta_n^{(\gamma/8)}(\varepsilon) + 4m_n^{-1}$, which establishes (51).

Finally, Lemma 41 implies that on $\hat{H}_n^{(1)}(\varepsilon) \cap H_{m_n}^{(ii)}$, $\forall m \in \mathcal{U}_n^{(\tilde{d}_f)}$, (52) holds.

Thus, defining

$$\hat{H}_n(\varepsilon) = \hat{H}_n^{(1)}(\varepsilon) \cap \hat{H}_n^{(2)} \cap H_{m_n}^{(ii)},$$

it remains only to establish (49). By a union bound, we have

$$\begin{aligned}
1 - \mathbb{P}\left(\hat{H}_n\right) &\leq \left(1 - \mathbb{P}\left(H_{m_n}(\varepsilon/2)\right)\right) + \left(1 - \mathbb{P}\left(H_{m_n}^{(i)}\right)\right) + \mathbb{P}\left(H_{m_n}^{(i)} \setminus H_{m_n}^{(ii)}\right) \\
&\quad + \mathbb{P}\left(H_{m_n}^{(i)} \setminus H_{m_n}^{(iii)}(\gamma/16)\right) + \left(1 - \mathbb{P}\left(H_{m_n}^{(iv)}\right)\right) + \mathbb{P}\left(\hat{H}_n^{(1)}(\varepsilon) \setminus \hat{H}_n^{(2)}\right). \\
&\leq \varepsilon/2 + c^{(i)} \cdot \exp\left\{-\tilde{M}(m_n)/4\right\} + c^{(ii)} \cdot \exp\left\{-\tilde{M}(m_n)^{1/3}/60\right\} \\
&\quad + c^{(iii)}(\gamma/16) \cdot \exp\left\{-\tilde{M}(m_n)\gamma^2/256\right\} + 3\tilde{d}_f \cdot \exp\{-2m_n\} \\
&\quad + \tilde{d}_f \cdot \exp\left\{1 - n/\left(18\cdot 2^{\tilde{d}_f}\right)\right\} \\
&\leq \varepsilon/2 + \left(c^{(i)} + c^{(ii)} + c^{(iii)}(\gamma/16) + 6\tilde{d}_f\right) \cdot \exp\left\{-n\tilde{\delta}_f \gamma^2 2^{-\tilde{d}_f - 12}\right\}. \qquad (55)
\end{aligned}$$

We have chosen n large enough so that (55) is at most $(3/4)\varepsilon$, which establishes (49). ■

The following result is a slightly stronger version of Theorem 6.

Lemma 46 *For any passive learning algorithm $\mathcal{A}_p$, if $\mathcal{A}_p$ achieves a label complexity Λ_p with $\infty > \Lambda_p(\varepsilon, f, \mathcal{P}) = \omega(\log(1/\varepsilon))$, then Meta-Algorithm 1, with $\mathcal{A}_p$ as its argument, achieves a label complexity Λ_a such that $\Lambda_a(3\varepsilon, f, \mathcal{P}) = o(\Lambda_p(\varepsilon, f, \mathcal{P}))$.*

Proof Suppose $\mathcal{A}_p$ achieves label complexity Λ_p with $\infty > \Lambda_p(\varepsilon, f, \mathcal{P}) = \omega(\log(1/\varepsilon))$. Let $\varepsilon \in (0,1)$, define $L(n;\varepsilon) = \left\lfloor n / \left(6 \cdot 2^{\tilde{d}_f} \left(\Delta_n^{(\gamma/8)}(\varepsilon) + 4m_n^{-1}\right)\right)\right\rfloor$ (for any $n \in \mathbb{N}$), and let $L^{-1}(m;\varepsilon) = \max\{n \in \mathbb{N} : L(n;\varepsilon) < m\}$ (for any $m \in (0,\infty)$). Define

$$c_1 = \max\left\{\hat{c}_1, 2 \cdot 6^3 (d+1)\tilde{d}_f \ln(e(d+1))\right\} \quad \text{and} \quad c_2 = \max\left\{\hat{c}_2, 4e(d+1)\right\},$$

and suppose

$$n \geq \max\left\{c_1 \ln(c_2/\varepsilon), 1 + L^{-1}\left(\Lambda_p(\varepsilon, f, \mathcal{P}); \varepsilon\right)\right\}.$$

Consider running Meta-Algorithm 1 with $\mathcal{A}_p$ and n as inputs, while f is the target function and $\mathcal{P}$ is the data distribution.

Letting $\hat{h}_n$ denote the classifier returned from Meta-Algorithm 1, Lemma 34 implies that on an event $\hat{E}_n$ with $\mathbb{P}(\hat{E}_n) \geq 1 - e(d+1) \cdot \exp\left\{-\lfloor n/3 \rfloor / (72\tilde{d}_f(d+1)\ln(e(d+1)))\right\} \geq 1 - \varepsilon/4$, we have

$$\mathrm{er}(\hat{h}_n) \leq 2\,\mathrm{er}\left(\mathcal{A}_p\left(\mathcal{L}_{\tilde{d}_f}\right)\right).$$

By a union bound, the event $\hat{G}_n(\varepsilon) = \hat{E}_n \cap \hat{H}_n(\varepsilon)$ has $\mathbb{P}\left(\hat{G}_n(\varepsilon)\right) \geq 1 - \varepsilon$. Thus,

$$\begin{aligned}
\mathbb{E}\left[\mathrm{er}\left(\hat{h}_n\right)\right] &\leq \mathbb{E}\left[\mathbb{1}_{\hat{G}_n(\varepsilon)} \mathbb{1}\left[|\mathcal{L}_{\tilde{d}_f}| \geq \Lambda_p(\varepsilon, f, \mathcal{P})\right] \mathrm{er}\left(\hat{h}_n\right)\right] \\
&\qquad + \mathbb{P}\left(\hat{G}_n(\varepsilon) \cap \left\{|\mathcal{L}_{\tilde{d}_f}| < \Lambda_p(\varepsilon, f, \mathcal{P})\right\}\right) + \mathbb{P}\left(\hat{G}_n(\varepsilon)^c\right) \\
&\leq \mathbb{E}\left[\mathbb{1}_{\hat{G}_n(\varepsilon)} \mathbb{1}\left[|\mathcal{L}_{\tilde{d}_f}| \geq \Lambda_p(\varepsilon, f, \mathcal{P})\right] 2\,\mathrm{er}\left(\mathcal{A}_p\left(\mathcal{L}_{\tilde{d}_f}\right)\right)\right] \\
&\qquad + \mathbb{P}\left(\hat{G}_n(\varepsilon) \cap \left\{|\mathcal{L}_{\tilde{d}_f}| < \Lambda_p(\varepsilon, f, \mathcal{P})\right\}\right) + \varepsilon. \qquad (56)
\end{aligned}$$

On $\hat{G}_n(\varepsilon)$, (51) of Lemma 45 implies $|\mathcal{L}_{\tilde{d}_f}| \geq L(n;\varepsilon)$, and we chose n large enough so that $L(n;\varepsilon) \geq \Lambda_p(\varepsilon, f, \mathcal{P})$. Thus, the second term in (56) is zero, and we have

$$\begin{aligned}
\mathbb{E}\left[\mathrm{er}\left(\hat{h}_n\right)\right] &\leq 2 \cdot \mathbb{E}\left[\mathbb{1}_{\hat{G}_n(\varepsilon)} \mathbb{1}\left[|\mathcal{L}_{\tilde{d}_f}| \geq \Lambda_p(\varepsilon, f, \mathcal{P})\right] \mathrm{er}\left(\mathcal{A}_p\left(\mathcal{L}_{\tilde{d}_f}\right)\right)\right] + \varepsilon \\
&= 2 \cdot \mathbb{E}\left[\mathbb{E}\left[\mathbb{1}_{\hat{G}_n(\varepsilon)} \mathrm{er}\left(\mathcal{A}_p\left(\mathcal{L}_{\tilde{d}_f}\right)\right) \middle| |\mathcal{L}_{\tilde{d}_f}|\right] \mathbb{1}\left[|\mathcal{L}_{\tilde{d}_f}| \geq \Lambda_p(\varepsilon, f, \mathcal{P})\right]\right] + \varepsilon. \qquad (57)
\end{aligned}$$

Note that for any ℓ with $\mathbb{P}(|\mathcal{L}_{\tilde{d}_f}| = \ell) > 0$, the conditional distribution of $\left\{X_m : m \in \mathcal{U}_n^{(\tilde{d}_f)}\right\}$ given $\left\{|\mathcal{L}_{\tilde{d}_f}| = \ell\right\}$ is simply the product $\mathcal{P}^\ell$ (i.e., conditionally i.i.d.), which is the same as the distribution of $\{X_1, X_2, \ldots, X_\ell\}$. Furthermore, on $\hat{G}_n(\varepsilon)$, (50) implies that the $t < \lfloor 2n/3 \rfloor$ condition is always satisfied in Step 6 of Meta-Algorithm 1 while $k \leq \tilde{d}_f$, and (52) implies that the inferred labels from Step 8 for $k = \tilde{d}_f$ are all correct. Therefore, for any such ℓ with $\ell \geq \Lambda_p(\varepsilon, f, \mathcal{P})$, we have

$$\mathbb{E}\left[\mathbb{1}_{\hat{G}_n(\varepsilon)} \mathrm{er}\left(\mathcal{A}_p\left(\mathcal{L}_{\tilde{d}_f}\right)\right) \middle| \left\{|\mathcal{L}_{\tilde{d}_f}| = \ell\right\}\right] \leq \mathbb{E}\left[\mathrm{er}\left(\mathcal{A}_p\left(\mathcal{Z}_\ell\right)\right)\right] \leq \varepsilon.$$

In particular, this means (57) is at most 3ε. This implies that Meta-Algorithm 1, with $\mathcal{A}_p$ as its argument, achieves a label complexity Λ_a such that

$$\Lambda_a(3\varepsilon, f, \mathcal{P}) \leq \max\left\{c_1 \ln(c_2/\varepsilon), 1 + L^{-1}\left(\Lambda_p(\varepsilon, f, \mathcal{P}); \varepsilon\right)\right\}.$$

Since $\Lambda_p(\varepsilon, f, \mathcal{P}) = \omega(\log(1/\varepsilon)) \Rightarrow c_1 \ln(c_2/\varepsilon) = o\left(\Lambda_p(\varepsilon, f, \mathcal{P})\right)$, it remains only to show that $L^{-1}\left(\Lambda_p(\varepsilon, f, \mathcal{P}); \varepsilon\right) = o\left(\Lambda_p(\varepsilon, f, \mathcal{P})\right)$. Note that $\forall \varepsilon \in (0,1)$, $L(1;\varepsilon) = 0$ and $L(n;\varepsilon)$ is diverging in n. Furthermore, by Lemma 38, we know that for any $\mathbb{N}$-valued $N(\varepsilon) = \omega(\log(1/\varepsilon))$, we have $\Delta_{N(\varepsilon)}^{(\gamma/8)}(\varepsilon) = o(1)$, which implies $L(N(\varepsilon);\varepsilon) = \omega(N(\varepsilon))$. Thus, since $\Lambda_p(\varepsilon, f, \mathcal{P}) = \omega(\log(1/\varepsilon))$, Lemma 31 implies $L^{-1}\left(\Lambda_p(\varepsilon, f, \mathcal{P}); \varepsilon\right) = o\left(\Lambda_p(\varepsilon, f, \mathcal{P})\right)$, as desired.

This establishes the result for an arbitrary $\gamma \in (0,1)$. To specialize to the specific procedure stated as Meta-Algorithm 1, we simply take $\gamma = 1/2$. ■

Proof [Theorem 6] Theorem 6 now follows immediately from Lemma 46. Specifically, we have proven Lemma 46 for an arbitrary distribution $\mathcal{P}$ on $\mathcal{X}$, an arbitrary $f \in \mathrm{cl}(\mathbb{C})$, and an arbitrary passive algorithm $\mathcal{A}_p$. Therefore, it will certainly hold for every $\mathcal{P}$ and $f \in \mathbb{C}$, and since every $(f, \mathcal{P}) \in \mathrm{Nontrivial}(\Lambda_p)$ has $\infty > \Lambda_p(\varepsilon, f, \mathcal{P}) = \omega(\log(1/\varepsilon))$, the implication that Meta-Algorithm 1 activizes every passive algorithm $\mathcal{A}_p$ for $\mathbb{C}$ follows. ■

Careful examination of the proofs above reveals that the "3" in Lemma 46 can be set to any arbitrary constant strictly larger than 1, by an appropriate modification of the "7/12" threshold in ActiveSelect. In fact, if we were to replace Step 4 of ActiveSelect by instead selecting $\hat{k} = \mathrm{argmin}_k \max_{j \neq k} m_{kj}$ (where $m_{kj} = \mathrm{er}_{Q_{kj}}(h_k)$ when $k < j$), then we could even make this a certain $(1 + o(1))$ function of ε, at the expense of larger constant factors in Λ_a.

Appendix C. The Label Complexity of Meta-Algorithm 2

As mentioned, Theorem 10 is essentially implied by the details of the proof of Theorem 16 in Appendix D below. Here we present a proof of Theorem 13, along with two useful related lemmas. The first, Lemma 47, lower bounds the expected number of label requests Meta-Algorithm 2 would make while processing a given number of random unlabeled examples. The second, Lemma 48, bounds the amount by which each label request is expected to reduce the probability mass in the region of disagreement. Although we will only use Lemma 48 in our proof of Theorem 13, Lemma 47 may be of independent interest, as it provides additional insights into the behavior of disagreement based methods, as related to the disagreement coefficient, and is included for this reason.

Throughout, we fix an arbitrary class $\mathbb{C}$, a target function $f \in \mathbb{C}$, and a distribution $\mathcal{P}$, and we continue using the notational conventions of the proofs above, such as $V_m^\star = \{h \in \mathbb{C} : \forall i \leq m, h(X_i) = f(X_i)\}$ (with $V_0^\star = \mathbb{C}$). Additionally, for $t \in \mathbb{N}$, define the random variable

$$M(t) = \min\left\{m \in \mathbb{N} : \sum_{\ell=1}^{m} \mathbb{1}_{\mathrm{DIS}\left(V_{\ell-1}^\star\right)}(X_\ell) = t\right\},$$

which represents the index of the t^{th} unlabeled example Meta-Algorithm 2 would request the label of (assuming it has not yet halted).

The two aforementioned lemmas are formally stated as follows.

Lemma 47 *For any $r \in (0,1)$ and $\ell \in \mathbb{N}$,*

$$\mathbb{E}\left[\mathcal{P}\left(\mathrm{DIS}\left(V_\ell^\star \cap \mathrm{B}\left(f,r\right)\right)\right)\right] \geq (1-r)^\ell \mathcal{P}\left(\mathrm{DIS}\left(\mathrm{B}\left(f,r\right)\right)\right),$$

and

$$\mathbb{E}\left[\sum_{m=1}^{\lceil 1/r \rceil} \mathbb{1}_{\mathrm{DIS}\left(V_{m-1}^\star \cap \mathrm{B}(f,r)\right)}\left(X_m\right)\right] \geq \frac{\mathcal{P}\left(\mathrm{DIS}\left(\mathrm{B}(f,r)\right)\right)}{2r}.$$

Lemma 48 *For any $r \in (0,1)$ and $n \in \mathbb{N}$,*

$$\mathbb{E}\left[\mathcal{P}\left(\mathrm{DIS}\left(V_{M(n)}^\star \cap \mathrm{B}\left(f,r\right)\right)\right)\right] \geq \mathcal{P}\left(\mathrm{DIS}\left(\mathrm{B}(f,r)\right)\right) - nr.$$

Note these results immediately imply that

$$\mathbb{E}\left[\sum_{m=1}^{\lceil 1/r \rceil} \mathbb{1}_{\mathrm{DIS}\left(V_{m-1}^\star\right)}\left(X_m\right)\right] \geq \frac{\mathcal{P}\left(\mathrm{DIS}\left(\mathrm{B}(f,r)\right)\right)}{2r}$$

and

$$\mathbb{E}\left[\mathcal{P}\left(\mathrm{DIS}\left(V_{M(n)}^\star\right)\right)\right] \geq \mathcal{P}\left(\mathrm{DIS}\left(\mathrm{B}(f,r)\right)\right) - nr,$$

which are then directly relevant to the expected number of label requests made by Meta-Algorithm 2 among the first m data points, and the probability Meta-Algorithm 2 requests the label of the next point, after already making n label requests, respectively.

Before proving these lemmas, let us first mention their relevance to the disagreement coefficient analysis. Specifically, for any $\varepsilon \in (0,r]$, we have

$$\mathbb{E}\left[\sum_{m=1}^{\lceil 1/\varepsilon \rceil} \mathbb{1}_{\mathrm{DIS}\left(V_{m-1}^\star\right)}\left(X_m\right)\right] \geq \mathbb{E}\left[\sum_{m=1}^{\lceil 1/r \rceil} \mathbb{1}_{\mathrm{DIS}\left(V_{m-1}^\star\right)}\left(X_m\right)\right] \geq \frac{\mathcal{P}\left(\mathrm{DIS}\left(\mathrm{B}(f,r)\right)\right)}{2r}.$$

In particular, maximizing over $r > \varepsilon$, we have

$$\mathbb{E}\left[\sum_{m=1}^{\lceil 1/\varepsilon \rceil} \mathbb{1}_{\mathrm{DIS}\left(V_{m-1}^\star\right)}\left(X_m\right)\right] \geq \theta_f(\varepsilon)/2.$$

Thus, the expected number of label requests among the first $\lceil 1/\varepsilon \rceil$ unlabeled examples processed by Meta-Algorithm 2 is at least $\theta_f(\varepsilon)/2$ (assuming it does not halt first). Similarly, for any $\varepsilon \in (0,r]$, for any $n \leq \mathcal{P}(\mathrm{DIS}(\mathrm{B}(f,r)))/(2r)$, Lemma 48 implies

$$\mathbb{E}\left[\mathcal{P}\left(\mathrm{DIS}\left(V_{M(n)}^\star\right)\right)\right] \geq \mathcal{P}\left(\mathrm{DIS}\left(\mathrm{B}(f,r)\right)\right)/2 \geq \mathcal{P}\left(\mathrm{DIS}\left(\mathrm{B}(f,\varepsilon)\right)\right)/2.$$

Maximizing over $r > \varepsilon$, we see that

$$n \leq \theta_f(\varepsilon)/2 \implies \mathbb{E}\left[\mathcal{P}\left(\mathrm{DIS}\left(V_{M(n)}^\star\right)\right)\right] \geq \mathcal{P}\left(\mathrm{DIS}\left(\mathrm{B}(f,\varepsilon)\right)\right)/2.$$

In other words, for Meta-Algorithm 2 to arrive at a region of disagreement with expected probability mass less than $\mathcal{P}(\mathrm{DIS}(\mathrm{B}(f,\varepsilon)))/2$ requires a budget n of at least $\theta_f(\varepsilon)/2$.

We now present proofs of Lemmas 47 and 48.

Proof [Lemma 47] Let $D_m = \mathrm{DIS}\left(V_m^\star \cap \mathrm{B}(f,r)\right)$. Since

$$\mathbb{E}\left[\sum_{m=1}^{\lceil 1/r\rceil} \mathbb{1}_{D_{m-1}}(X_m)\right] = \sum_{m=1}^{\lceil 1/r\rceil} \mathbb{E}\left[\mathbb{P}\left(X_m \in D_{m-1} \middle| V_{m-1}^\star\right)\right] = \sum_{m=1}^{\lceil 1/r\rceil} \mathbb{E}\left[\mathcal{P}\left(D_{m-1}\right)\right], \tag{58}$$

we focus on lower bounding $\mathbb{E}\left[\mathcal{P}\left(D_m\right)\right]$ for $m \in \mathbb{N}\cup\{0\}$. Note that for any $x \in \mathrm{DIS}(\mathrm{B}(f,r))$, there exists some $h_x \in \mathrm{B}(f,r)$ with $h_x(x) \neq f(x)$, and if this $h_x \in V_m^\star$, then $x \in D_m$ as well. This means $\forall x, \mathbb{1}_{D_m}(x) \geq \mathbb{1}_{\mathrm{DIS}(\mathrm{B}(f,r))}(x)\cdot \mathbb{1}_{V_m^\star}(h_x) = \mathbb{1}_{\mathrm{DIS}(\mathrm{B}(f,r))}(x)\cdot \prod_{\ell=1}^m \mathbb{1}_{\mathrm{DIS}(\{h_x,f\})^c}(X_\ell)$. Therefore,

$$\begin{aligned}
\mathbb{E}\left[\mathcal{P}\left(D_m\right)\right] &= \mathbb{P}\left(X_{m+1}\in D_m\right) = \mathbb{E}\left[\mathbb{E}\left[\mathbb{1}_{D_m}(X_{m+1})\middle| X_{m+1}\right]\right] \\
&\geq \mathbb{E}\left[\mathbb{E}\left[\mathbb{1}_{\mathrm{DIS}(\mathrm{B}(f,r))}(X_{m+1})\cdot\prod_{\ell=1}^m \mathbb{1}_{\mathrm{DIS}(\{h_{X_{m+1}},f\})^c}(X_\ell)\middle| X_{m+1}\right]\right] \\
&= \mathbb{E}\left[\prod_{\ell=1}^m \mathbb{P}\left(h_{X_{m+1}}(X_\ell) = f(X_\ell)\middle| X_{m+1}\right)\mathbb{1}_{\mathrm{DIS}(\mathrm{B}(f,r))}(X_{m+1})\right] \qquad (59)\\
&\geq \mathbb{E}\left[(1-r)^m \mathbb{1}_{\mathrm{DIS}(\mathrm{B}(f,r))}(X_{m+1})\right] = (1-r)^m \mathcal{P}(\mathrm{DIS}(\mathrm{B}(f,r))), \qquad (60)
\end{aligned}$$

where the equality in (59) is by conditional independence of the $\mathbb{1}_{\mathrm{DIS}(\{h_{X_{m+1}},f\})^c}(X_\ell)$ indicators, given X_{m+1}, and the inequality in (60) is due to $h_{X_{m+1}} \in \mathrm{B}(f,r)$. This indicates (58) is at least

$$\begin{aligned}
\sum_{m=1}^{\lceil 1/r\rceil}(1-r)^{m-1}\mathcal{P}\left(\mathrm{DIS}\left(\mathrm{B}(f,r)\right)\right) &= \left(1-(1-r)^{\lceil 1/r\rceil}\right)\frac{\mathcal{P}\left(\mathrm{DIS}\left(\mathrm{B}(f,r)\right)\right)}{r} \\
&\geq \left(1-\frac{1}{e}\right)\frac{\mathcal{P}\left(\mathrm{DIS}\left(\mathrm{B}(f,r)\right)\right)}{r} \geq \frac{\mathcal{P}\left(\mathrm{DIS}\left(\mathrm{B}(f,r)\right)\right)}{2r}.
\end{aligned}$$

■

Proof [Lemma 48] For each $m \in \mathbb{N}\cup\{0\}$, let $D_m = \mathrm{DIS}\left(\mathrm{B}(f,r)\cap V_m^\star\right)$. For convenience, let $M(0) = 0$. We prove the result by induction. We clearly have $\mathbb{E}\left[\mathcal{P}\left(D_{M(0)}\right)\right] = \mathbb{E}\left[\mathcal{P}\left(D_0\right)\right] = \mathcal{P}(\mathrm{DIS}(\mathrm{B}(f,r)))$, which serves as our base case. Now fix any $n \in \mathbb{N}$ and take as the inductive hypothesis that

$$\mathbb{E}\left[\mathcal{P}\left(D_{M(n-1)}\right)\right] \geq \mathcal{P}(\mathrm{DIS}(\mathrm{B}(f,r))) - (n-1)r.$$

As in the proof of Lemma 47, for any $x \in D_{M(n-1)}$, there exists $h_x \in \mathrm{B}(f,r)\cap V_{M(n-1)}^\star$ with $h_x(x) \neq f(x)$; unlike the proof of Lemma 47, here h_x is a random variable, determined by $V_{M(n-1)}^\star$. If h_x is also in $V_{M(n)}^\star$, then $x \in D_{M(n)}$ as well. Thus, $\forall x, \mathbb{1}_{D_{M(n)}}(x) \geq \mathbb{1}_{D_{M(n-1)}}(x)\cdot \mathbb{1}_{V_{M(n)}^\star}(h_x) = \mathbb{1}_{D_{M(n-1)}}(x)\cdot \mathbb{1}_{\mathrm{DIS}(\{h_x,f\})^c}(X_{M(n)})$, where this last equality is due to the fact that every $m \in \{M(n-1)+1,\ldots, M(n)-1\}$ has $X_m \notin \mathrm{DIS}\left(V_{m-1}^\star\right)$, so that in particular $h_x(X_m) = f(X_m)$. Therefore, letting $X \sim \mathcal{P}$ be

independent of the data $\mathcal{Z}$,

$$\mathbb{E}\left[\mathcal{P}\left(D_{M(n)}\right)\right] = \mathbb{E}\left[\mathbb{1}_{D_{M(n)}}(X)\right] \geq \mathbb{E}\left[\mathbb{1}_{D_{M(n-1)}}(X)\cdot\mathbb{1}_{\mathrm{DIS}(\{h_X,f\})^c}(X_{M(n)})\right]$$
$$= \mathbb{E}\left[\mathbb{1}_{D_{M(n-1)}}(X)\cdot\mathbb{P}\left(h_X(X_{M(n)}) = f(X_{M(n)})\middle| X, V^\star_{M(n-1)}\right)\right]. \tag{61}$$

The conditional distribution of $X_{M(n)}$ given $V^\star_{M(n-1)}$ is merely $\mathcal{P}$ but with support restricted to $\mathrm{DIS}\left(V^\star_{M(n-1)}\right)$ and renormalized to a probability measure: that is $\mathcal{P}\left(\cdot\middle|\mathrm{DIS}\left(V^\star_{M(n-1)}\right)\right)$. Thus, since any $x \in D_{M(n-1)}$ has $\mathrm{DIS}(\{h_x,f\}) \subseteq \mathrm{DIS}\left(V^\star_{M(n-1)}\right)$, we have

$$\mathbb{P}\left(h_x(X_{M(n)}) \neq f(X_{M(n)})\middle| V^\star_{M(n-1)}\right) = \frac{\mathcal{P}\left(\mathrm{DIS}(\{h_x,f\})\right)}{\mathcal{P}\left(\mathrm{DIS}\left(V^\star_{M(n-1)}\right)\right)} \leq \frac{r}{\mathcal{P}\left(D_{M(n-1)}\right)},$$

where the inequality follows from $h_x \in \mathrm{B}(f,r)$ and $D_{M(n-1)} \subseteq \mathrm{DIS}\left(V^\star_{M(n-1)}\right)$. Therefore, (61) is at least

$$\mathbb{E}\left[\mathbb{1}_{D_{M(n-1)}}(X)\cdot\left(1-\frac{r}{\mathcal{P}(D_{M(n-1)})}\right)\right]$$
$$= \mathbb{E}\left[\mathbb{P}\left(X\in D_{M(n-1)}\middle| D_{M(n-1)}\right)\cdot\left(1-\frac{r}{\mathcal{P}(D_{M(n-1)})}\right)\right]$$
$$= \mathbb{E}\left[\mathcal{P}\left(D_{M(n-1)}\right)\cdot\left(1-\frac{r}{\mathcal{P}(D_{M(n-1)})}\right)\right] = \mathbb{E}\left[\mathcal{P}\left(D_{M(n-1)}\right)\right] - r.$$

By the inductive hypothesis, this is at least $\mathcal{P}(\mathrm{DIS}(\mathrm{B}(f,r))) - nr$. ■

With Lemma 48 in hand, we are ready for the proof of Theorem 13.

Proof [Theorem 13] Let $\mathbb{C}$, f, $\mathcal{P}$, and λ be as in the theorem statement. For $m \in \mathbb{N}$, let $\lambda^{-1}(m) = \inf\{\varepsilon > 0 : \lambda(\varepsilon) \leq m\}$, or 1 if this is not defined. We define $\mathcal{A}_p$ as a randomized algorithm such that, for $m \in \mathbb{N}$ and $\mathcal{L} \in (\mathcal{X}\times\{-1,+1\})^m$, $\mathcal{A}_p(\mathcal{L})$ returns f with probability $1-\lambda^{-1}(|\mathcal{L}|)$ and returns $-f$ with probability $\lambda^{-1}(|\mathcal{L}|)$ (independent of the contents of $\mathcal{L}$). Note that, for any integer $m \geq \lambda(\varepsilon)$, $\mathbb{E}\left[\mathrm{er}\left(\mathcal{A}_p\left(\mathcal{Z}_m\right)\right)\right] = \lambda^{-1}(m) \leq \lambda^{-1}(\lambda(\varepsilon)) \leq \varepsilon$. Therefore, $\mathcal{A}_p$ achieves some label complexity Λ_p with $\Lambda_p(\varepsilon,f,\mathcal{P}) = \lambda(\varepsilon)$ for all $\varepsilon > 0$.

If $\theta_f\left(\lambda(\varepsilon)^{-1}\right) \neq \omega(1)$, then monotonicity implies $\theta_f\left(\lambda(\varepsilon)^{-1}\right) = O(1)$, and since every label complexity Λ_a is $\Omega(1)$, the result clearly holds. Otherwise, suppose $\theta_f\left(\lambda(\varepsilon)^{-1}\right) = \omega(1)$; in particular, this means $\exists\varepsilon_0 \in (0,1/2)$ such that $\theta_f\left(\lambda(2\varepsilon_0)^{-1}\right) \geq 12$. Fix any $\varepsilon \in (0,\varepsilon_0)$, let $r > \lambda(2\varepsilon)^{-1}$ be such that $\frac{\mathcal{P}(\mathrm{DIS}(\mathrm{B}(f,r)))}{r} \geq \theta_f\left(\lambda(2\varepsilon)^{-1}\right)/2$, and let $n \in \mathbb{N}$ satisfy $n \leq \theta_f\left(\lambda(2\varepsilon)^{-1}\right)/4$.

Consider running Meta-Algorithm 2 with arguments $\mathcal{A}_p$ and n, and let $\hat{\mathcal{L}}$ denote the final value of the set $\mathcal{L}$, and let $\breve{m}$ denote the value of m upon reaching Step 6. Note that any $m < \lambda(2\varepsilon)$ and $L \in (\mathcal{X}\times\{-1,+1\})^m$ has $\mathrm{er}\left(\mathcal{A}_p(L)\right) = \lambda^{-1}(m) \geq \inf\{\varepsilon' > 0 : \lambda(\varepsilon') < \lambda(2\varepsilon)\} \geq 2\varepsilon$. Therefore, we have

$$\mathbb{E}\left[\mathrm{er}\left(\mathcal{A}_p\left(\hat{\mathcal{L}}\right)\right)\right] \geq 2\varepsilon\mathbb{P}\left(|\hat{\mathcal{L}}| < \lambda(2\varepsilon)\right) = 2\varepsilon\mathbb{P}\left(\left\lfloor n/\left(6\hat{\Delta}\right)\right\rfloor < \lambda(2\varepsilon)\right)$$
$$= 2\varepsilon\mathbb{P}\left(\hat{\Delta} > \frac{n}{6\lambda(2\varepsilon)}\right) = 2\varepsilon\left(1-\mathbb{P}\left(\hat{\Delta} \leq \frac{n}{6\lambda(2\varepsilon)}\right)\right). \tag{62}$$

Since $n \leq \theta_f\left(\lambda(2\varepsilon)^{-1}\right)/4 \leq \mathcal{P}(\mathrm{DIS}(\mathrm{B}(f,r)))/(2r) < \lambda(2\varepsilon)\mathcal{P}(\mathrm{DIS}(\mathrm{B}(f,r)))/2$, we have

$$\mathbb{P}\left(\hat{\Delta} \leq \frac{n}{6\lambda(2\varepsilon)}\right) \leq \mathbb{P}\left(\hat{\Delta} < \mathcal{P}(\mathrm{DIS}(\mathrm{B}(f,r)))/12\right)$$
$$\leq \mathbb{P}\left(\left\{\mathcal{P}\left(\mathrm{DIS}\left(V^\star_{\check{m}}\right)\right) < \mathcal{P}(\mathrm{DIS}(\mathrm{B}(f,r)))/12\right\} \cup \left\{\hat{\Delta} < \mathcal{P}\left(\mathrm{DIS}\left(V^\star_{\check{m}}\right)\right)\right\}\right). \quad (63)$$

Since $\check{m} \leq M(\lceil n/2 \rceil)$, monotonicity and a union bound imply this is at most

$$\mathbb{P}\left(\mathcal{P}\left(\mathrm{DIS}\left(V^\star_{M(\lceil n/2\rceil)}\right)\right) < \mathcal{P}(\mathrm{DIS}(\mathrm{B}(f,r)))/12\right) + \mathbb{P}\left(\hat{\Delta} < \mathcal{P}\left(\mathrm{DIS}\left(V^\star_{\check{m}}\right)\right)\right). \quad (64)$$

Markov's inequality implies

$$\begin{aligned}
&\mathbb{P}\left(\mathcal{P}\left(\mathrm{DIS}\left(V^\star_{M(\lceil n/2\rceil)}\right)\right) < \mathcal{P}(\mathrm{DIS}(\mathrm{B}(f,r)))/12\right) \\
&= \mathbb{P}\left(\mathcal{P}(\mathrm{DIS}(\mathrm{B}(f,r))) - \mathcal{P}\left(\mathrm{DIS}\left(V^\star_{M(\lceil n/2\rceil)}\right)\right) > \frac{11}{12}\mathcal{P}(\mathrm{DIS}(\mathrm{B}(f,r)))\right) \\
&\leq \mathbb{P}\left(\mathcal{P}(\mathrm{DIS}(\mathrm{B}(f,r))) - \mathcal{P}\left(\mathrm{DIS}\left(V^\star_{M(\lceil n/2\rceil)} \cap \mathrm{B}(f,r)\right)\right) > \frac{11}{12}\mathcal{P}(\mathrm{DIS}(\mathrm{B}(f,r)))\right) \\
&\leq \frac{\mathbb{E}\left[\mathcal{P}(\mathrm{DIS}(\mathrm{B}(f,r))) - \mathcal{P}\left(\mathrm{DIS}\left(V^\star_{M(\lceil n/2\rceil)} \cap \mathrm{B}(f,r)\right)\right)\right]}{\frac{11}{12}\mathcal{P}(\mathrm{DIS}(\mathrm{B}(f,r)))} \\
&= \frac{12}{11}\left(1 - \frac{\mathbb{E}\left[\mathcal{P}\left(\mathrm{DIS}\left(V^\star_{M(\lceil n/2\rceil)} \cap \mathrm{B}(f,r)\right)\right)\right]}{\mathcal{P}(\mathrm{DIS}(\mathrm{B}(f,r)))}\right).
\end{aligned}$$

Lemma 48 implies this is at most $\frac{12}{11}\frac{\lceil n/2\rceil r}{\mathcal{P}(\mathrm{DIS}(\mathrm{B}(f,r)))} \leq \frac{12}{11}\left\lceil \frac{\mathcal{P}(\mathrm{DIS}(\mathrm{B}(f,r)))}{4r}\right\rceil \frac{r}{\mathcal{P}(\mathrm{DIS}(\mathrm{B}(f,r)))}$. Since any $a \geq 3/2$ has $\lceil a\rceil \leq (3/2)a$, and $\theta_f\left(\lambda(2\varepsilon)^{-1}\right) \geq 12$ implies $\frac{\mathcal{P}(\mathrm{DIS}(\mathrm{B}(f,r)))}{4r} \geq 3/2$, we have $\left\lceil \frac{\mathcal{P}(\mathrm{DIS}(\mathrm{B}(f,r)))}{4r}\right\rceil \leq \frac{3}{8}\frac{\mathcal{P}(\mathrm{DIS}(\mathrm{B}(f,r)))}{r}$, so that $\frac{12}{11}\left\lceil \frac{\mathcal{P}(\mathrm{DIS}(\mathrm{B}(f,r)))}{4r}\right\rceil \frac{r}{\mathcal{P}(\mathrm{DIS}(\mathrm{B}(f,r)))} \leq \frac{9}{22}$. Combining the above, we have

$$\mathbb{P}\left(\mathcal{P}\left(\mathrm{DIS}\left(V^\star_{M(\lceil n/2\rceil)}\right)\right) < \mathcal{P}(\mathrm{DIS}(\mathrm{B}(f,r)))/12\right) \leq \frac{9}{22}. \quad (65)$$

Examining the second term in (64), Hoeffding's inequality and the definition of $\hat{\Delta}$ from (13) imply

$$\mathbb{P}\left(\hat{\Delta} < \mathcal{P}\left(\mathrm{DIS}\left(V^\star_{\check{m}}\right)\right)\right) = \mathbb{E}\left[\mathbb{P}\left(\hat{\Delta} < \mathcal{P}\left(\mathrm{DIS}\left(V^\star_{\check{m}}\right)\right) \middle| V^\star_{\check{m}}, \check{m}\right)\right] \leq \mathbb{E}\left[e^{-8\check{m}}\right] \leq e^{-8} < 1/11. \quad (66)$$

Combining (62), (63), (64), (65), and (66) implies

$$\mathbb{E}\left[\mathrm{er}\left(\mathcal{A}_p\left(\hat{\mathcal{L}}\right)\right)\right] > 2\varepsilon\left(1 - \frac{9}{22} - \frac{1}{11}\right) = \varepsilon.$$

Thus, for any label complexity Λ_a achieved by running Meta-Algorithm 2 with $\mathcal{A}_p$ as its argument, we must have $\Lambda_a(\varepsilon, f, \mathcal{P}) > \theta_f\left(\lambda(2\varepsilon)^{-1}\right)/4$. Since this is true for all $\varepsilon \in (0, \varepsilon_0)$, this establishes the result. ■

Appendix D. The Label Complexity of Meta-Algorithm 3

As in Appendix B, we will assume $\mathbb{C}$ is a fixed VC class, $\mathcal{P}$ is some arbitrary distribution, and $f \in \text{cl}(\mathbb{C})$ is an arbitrary fixed function. We continue using the notation introduced above: in particular, $\mathcal{S}^k(\mathcal{H}) = \left\{S \in \mathcal{X}^k : \mathcal{H} \text{ shatters } S\right\}$, $\mathcal{S}^k(\mathcal{H}) = \mathcal{X}^k \setminus \mathcal{S}^k(\mathcal{H})$, $\partial^k_{\mathcal{H}} f = \mathcal{X}^k \setminus \partial^k_{\mathcal{H}} f$, and $\tilde{\delta}_f = \mathcal{P}^{\tilde{d}_f - 1}\left(\partial^{\tilde{d}_f - 1}_{\mathbb{C}} f\right)$. Also, as above, we will prove a more general result replacing the "$1/2$" in Steps 5, 9, and 12 of Meta-Algorithm 3 with an arbitrary value $\gamma \in (0,1)$; thus, the specific result for the stated algorithm will be obtained by taking $\gamma = 1/2$.

For the estimators $\hat{P}_m$ in Meta-Algorithm 3, we take precisely the same definitions as given in Appendix B.1 for the estimators in Meta-Algorithm 1. In particular, the quantities $\hat{\Delta}^{(k)}_m(x, W_2, \mathcal{H})$, $\hat{\Delta}^{(k)}_m(W_1, W_2, \mathcal{H})$, $\hat{\Gamma}^{(k)}_m(x, y, W_2, \mathcal{H})$, and $M^{(k)}_m(\mathcal{H})$ are all defined as in Appendix B.1, and the $\hat{P}_m$ estimators are again defined as in (11), (12) and (13).

Also, we sometimes refer to quantities defined above, such as $p_\xi(k, \ell, m)$ (defined in (34)), as well as the various events from the lemmas of the previous appendix, such as $H_\tau(\delta), H', H^{(i)}_\tau, H^{(ii)}_\tau$, $H^{(iii)}_\tau(\xi), H^{(iv)}_\tau$, and $G^{(i)}_\tau$.

D.1 Proof of Theorem 16

Throughout the proof, we will make reference to the sets V_m defined in Meta-Algorithm 3. Also let $V^{(k)}$ denote the final value of V obtained for the specified value of k in Meta-Algorithm 3. Both V_m and $V^{(k)}$ are implicitly functions of the budget, n, given to Meta-Algorithm 3. As above, we continue to denote by $V^\star_m = \{h \in \mathbb{C} : \forall i \le m, h(X_m) = f(X_m)\}$. One important fact we will use repeatedly below is that if $V_m = V^\star_m$ for some m, then since Lemma 35 implies that $V^\star_m \neq \emptyset$ on H', we must have that all of the previous $\hat{y}$ values were consistent with f, which means that $\forall \ell \le m$, $V_\ell = V^\star_\ell$. In particular, if $V^{(k')} = V^\star_m$ for the largest m value obtained while $k = k'$ in Meta-Algorithm 3, then $V_\ell = V^\star_\ell$ for all ℓ obtained while $k \le k'$ in Meta-Algorithm 3.

Additionally, define $\tilde{m}_n = \lfloor n/24 \rfloor$, and note that the value $m = \lceil n/6 \rceil$ is obtained while $k = 1$ in Meta-Algorithm 3. We also define the following quantities, which we will show are typically equal to related quantities in Meta-Algorithm 3. Define $\hat{m}_0 = 0$, $T^\star_0 = \lceil 2n/3 \rceil$, and $\hat{t}_0 = 0$, and for each $k \in \{1, \ldots, d+1\}$, inductively define

$$
\begin{aligned}
T_k^\star &= T_{k-1}^\star - \hat{t}_{k-1}, \\
I_{mk}^\star &= \mathbb{1}_{[\gamma,\infty)}\left(\hat{\Delta}_m^{(k)}\left(X_m, W_2, V_{m-1}^\star\right)\right), \forall m \in \mathbb{N}, \\
\check{m}_k &= \min\left\{ m \geq \hat{m}_{k-1} : \sum_{\ell=\hat{m}_{k-1}+1}^{m} I_{\ell k}^\star = \lceil T_k^\star/4 \rceil \right\} \cup \left\{ \max\left\{ k \cdot 2^n + 1, \hat{m}_{k-1} \right\} \right\}, \\
\hat{m}_k &= \check{m}_k + \left\lfloor T_k^\star / \left(3\hat{\Delta}_{\check{m}_k}^{(k)}\left(W_1, W_2, V_{\check{m}_k}^\star\right)\right)\right\rfloor, \\
\check{\mathcal{U}}_k &= (\hat{m}_{k-1}, \check{m}_k] \cap \mathbb{N}, \\
\hat{\mathcal{U}}_k &= (\check{m}_k, \hat{m}_k] \cap \mathbb{N}, \\
C_{mk}^\star &= \mathbb{1}_{\left[0, \left\lfloor 3T_k^\star/4 \right\rfloor\right)}\left(\sum_{\ell=\hat{m}_{k-1}+1}^{m-1} I_{\ell k}^\star \right) \\
Q_k^\star &= \sum_{m \in \hat{\mathcal{U}}_k} I_{mk}^\star \cdot C_{mk}^\star, \\
\text{and } \hat{t}_k &= Q_k^\star + \sum_{m \in \check{\mathcal{U}}_k} I_{mk}^\star.
\end{aligned}
$$

The meaning of these values can be understood in the context of Meta-Algorithm 3, under the condition that $V_m = V_m^\star$ for values of m obtained for the respective value of k. Specifically, under this condition, $T_k^\star$ corresponds to T_k, $\hat{t}_k$ represents the final value t for round k, $\check{m}_k$ represents the value of m upon reaching Step 9 in round k, while $\hat{m}_k$ represents the value of m at the end of round k, $\check{\mathcal{U}}_k$ corresponds to the set of indices arrived at in Step 4 during round k, while $\hat{\mathcal{U}}_k$ corresponds to the set of indices arrived at in Step 11 during round k, for $m \in \check{\mathcal{U}}_k$, $I_{mk}^\star$ indicates whether the label of X_m is requested, while for $m \in \hat{\mathcal{U}}_k$, $I_{mk}^\star \cdot C_{mk}^\star$ indicates whether the label of X_m is requested. Finally $Q_k^\star$ corresponds to the number of label requests in Step 13 during round k. In particular, note $\check{m}_1 \geq \tilde{m}_n$.

Lemma 49 *For any $\tau \in \mathbb{N}$, on the event $H' \cap G_\tau^{(i)}$, $\forall k, \ell, m \in \mathbb{N}$ with $k \leq \tilde{d}_f$, $\forall x \in \mathcal{X}$, for any sets $\mathcal{H}$ and $\mathcal{H}'$ with $V_\ell^\star \subseteq \mathcal{H} \subseteq \mathcal{H}' \subseteq \mathrm{B}(f, r_{1/6})$, if either $k = 1$ or $m \geq \tau$, then*

$$
\hat{\Delta}_m^{(k)}\left(x, W_2, \mathcal{H}\right) \leq (3/2)\hat{\Delta}_m^{(k)}\left(x, W_2, \mathcal{H}'\right).
$$

In particular, for any $\delta \in (0,1)$ and $\tau \geq \tau(1/6;\delta)$, on $H' \cap H_\tau(\delta) \cap G_\tau^{(i)}$, $\forall k, \ell, \ell', m \in \mathbb{N}$ with $m \geq \tau$, $\ell \geq \ell' \geq \tau$, and $k \leq \tilde{d}_f$, $\forall x \in \mathcal{X}$, $\hat{\Delta}_m^{(k)}\left(x, W_2, V_\ell^\star\right) \leq (3/2)\hat{\Delta}_m^{(k)}\left(x, W_2, V_{\ell'}^\star\right)$.

Proof First note that $\forall m \in \mathbb{N}$, $\forall x \in \mathcal{X}$,

$$
\hat{\Delta}_m^{(1)}\left(x, W_2, \mathcal{H}\right) = \mathbb{1}_{\mathrm{DIS}(\mathcal{H})}(x) \leq \mathbb{1}_{\mathrm{DIS}(\mathcal{H}')}(x) = \hat{\Delta}_m^{(1)}\left(x, W_2, \mathcal{H}'\right),
$$

so the result holds for $k = 1$. Lemma 35, Lemma 40, and monotonicity of $M_m^{(k)}(\cdot)$ imply that on $H' \cap G_\tau^{(i)}$, for any $m \geq \tau$ and $k \in \left\{2, \ldots, \tilde{d}_f\right\}$,

$$
M_m^{(k)}\left(\mathcal{H}\right) \geq \sum_{i=1}^{m^3} \mathbb{1}_{\partial_{\mathbb{C}}^{k-1} f}\left(S_i^{(k)}\right) \geq (2/3) M_m^{(k)}\left(\mathrm{B}(f, r_{1/6})\right) \geq (2/3) M_m^{(k)}\left(\mathcal{H}'\right),
$$

so that $\forall x \in \mathcal{X}$,

$$\begin{aligned}\hat{\Delta}_m^{(k)}(x, W_2, \mathcal{H}) &= M_m^{(k)}(\mathcal{H})^{-1} \sum_{i=1}^{m^3} \mathbb{1}_{\mathcal{S}^k(\mathcal{H})}\left(S_i^{(k)} \cup \{x\}\right) \\ &\leq M_m^{(k)}(\mathcal{H})^{-1} \sum_{i=1}^{m^3} \mathbb{1}_{\mathcal{S}^k(\mathcal{H}')}\left(S_i^{(k)} \cup \{x\}\right) \\ &\leq (3/2) M_m^{(k)}(\mathcal{H}')^{-1} \sum_{i=1}^{m^3} \mathbb{1}_{\mathcal{S}^k(\mathcal{H}')}\left(S_i^{(k)} \cup \{x\}\right) = (3/2)\hat{\Delta}_m^{(k)}(x, W_2, \mathcal{H}').\end{aligned}$$

The final claim follows from Lemma 29. ■

Lemma 50 *For any $k \in \{1, \ldots, d+1\}$, if $n \geq 3 \cdot 4^{k-1}$, then $T_k^\star \geq 4^{1-k}(2n/3)$ and $\hat{t}_k \leq \lfloor 3T_k^\star/4 \rfloor$.*

Proof Recall $T_1^\star = \lceil 2n/3 \rceil \geq 2n/3$. If $n \geq 2$, we also have $\lfloor 3T_1^\star/4 \rfloor \geq \lceil T_1^\star/4 \rceil$, so that (due to the $C_{m1}^\star$ factors) $\hat{t}_1 \leq \lfloor 3T_1^\star/4 \rfloor$. For the purpose of induction, suppose some $k \in \{2, \ldots, d+1\}$ has $n \geq 3 \cdot 4^{k-1}$, $T_{k-1}^\star \geq 4^{2-k}(2n/3)$, and $\hat{t}_{k-1} \leq \lfloor 3T_{k-1}^\star/4 \rfloor$. Then $T_k^\star = T_{k-1}^\star - \hat{t}_{k-1} \geq T_{k-1}^\star/4 \geq 4^{1-k}(2n/3)$, and since $n \geq 3 \cdot 4^{k-1}$, we also have $\lfloor 3T_k^\star/4 \rfloor \geq \lceil T_k^\star/4 \rceil$, so that $\hat{t}_k \leq \lfloor 3T_k^\star/4 \rfloor$ (again, due to the $C_{mk}^\star$ factors). Thus, by induction, this holds for all $k \in \{1, \ldots, d+1\}$ with $n \geq 3 \cdot 4^{k-1}$. ■

The next lemma indicates that the "$t < \lfloor 3T_k/4 \rfloor$" constraint in Step 12 is redundant for $k \leq \tilde{d}_f$. It is similar to (50) in Lemma 45, but is made only slightly more complicated by the fact that the $\hat{\Delta}^{(k)}$ estimate is calculated in Step 9 based on a set V_m different from the ones used to decide whether or not to request a label in Step 12.

Lemma 51 *There exist $(\mathbb{C}, \mathcal{P}, f, \gamma)$-dependent constants $\tilde{c}_1^{(i)}, \tilde{c}_2^{(i)} \in [1, \infty)$ such that, for any $\delta \in (0,1)$, and any integer $n \geq \tilde{c}_1^{(i)} \ln\left(\tilde{c}_2^{(i)}/\delta\right)$, on an event*

$$\tilde{H}_n^{(i)}(\delta) \subseteq G_{\tilde{m}_n}^{(i)} \cap H_{\tilde{m}_n}(\delta) \cap H_{\tilde{m}_n}^{(i)} \cap H_{\tilde{m}_n}^{(iii)}(\gamma/16) \cap H_{\tilde{m}_n}^{(iv)}$$

with $\mathbb{P}\left(\tilde{H}_n^{(i)}(\delta)\right) \geq 1 - 2\delta$, $\forall k \in \left\{1, \ldots, \tilde{d}_f\right\}$, $\hat{t}_k = \sum_{m=\hat{m}_{k-1}+1}^{\hat{m}_k} I_{mk}^\star \leq 3T_k^\star/4$.

Proof Define the constants

$$\tilde{c}_1^{(i)} = \max\left\{\tfrac{192d}{r_{(3/32)}}, \tfrac{3 \cdot 4^{\tilde{d}_f+6}}{\tilde{\delta}_f \gamma^2}\right\}, \quad \tilde{c}_2^{(i)} = \max\left\{\frac{8e}{r_{(3/32)}}, \left(c^{(i)} + c^{(iii)}(\gamma/16) + 125\tilde{d}_f \tilde{\delta}_f^{-1}\right)\right\},$$

and let $n^{(i)}(\delta) = \tilde{c}_1^{(i)} \ln\left(\tilde{c}_2^{(i)}/\delta\right)$. Fix any integer $n \geq n^{(i)}(\delta)$ and consider the event

$$\tilde{H}_n^{(1)}(\delta) = G_{\tilde{m}_n}^{(i)} \cap H_{\tilde{m}_n}(\delta) \cap H_{\tilde{m}_n}^{(i)} \cap H_{\tilde{m}_n}^{(iii)}(\gamma/16) \cap H_{\tilde{m}_n}^{(iv)}.$$

By Lemma 49 and the fact that $\check{m}_k \geq \tilde{m}_n$ for all $k \geq 1$, since $n \geq n^{(i)}(\delta) \geq 24\tau(1/6;\delta)$, on $\tilde{H}_n^{(1)}(\delta)$, $\forall k \in \{1,\ldots,\tilde{d}_f\}$, $\forall m \in \hat{\mathcal{U}}_k$,

$$\hat{\Delta}_m^{(k)}\left(X_m, W_2, V_{m-1}^\star\right) \leq (3/2)\hat{\Delta}_m^{(k)}\left(X_m, W_2, V_{\check{m}_k}^\star\right). \tag{67}$$

Now fix any $k \in \{1,\ldots,\tilde{d}_f\}$. Since $n \geq n^{(i)}(\delta) \geq 27 \cdot 4^{k-1}$, Lemma 50 implies $T_k^\star \geq 18$, which means $3T_k^\star/4 - \lceil T_k^\star/4 \rceil \geq 4T_k^\star/9$. Also note $\sum_{m \in \check{\mathcal{U}}_k} I_{mk}^\star \leq \lceil T_k^\star/4 \rceil$. Let $N_k = (4/3)\hat{\Delta}_{\check{m}_k}^{(k)}\left(W_1, W_2, V_{\check{m}_k}^\star\right)\left|\hat{\mathcal{U}}_k\right|$, and note that $\left|\hat{\mathcal{U}}_k\right| = \left\lfloor T_k^\star / \left(3\hat{\Delta}_{\check{m}_k}^{(k)}\left(W_1, W_2, V_{\check{m}_k}^\star\right)\right)\right\rfloor$, so that $N_k \leq (4/9)T_k^\star$. Thus, we have

$$\begin{aligned}
&\mathbb{P}\left(\tilde{H}_n^{(1)}(\delta) \cap \left\{\sum_{m=\hat{m}_{k-1}+1}^{\hat{m}_k} I_{mk}^\star > 3T_k^\star/4\right\}\right) \\
&\leq \mathbb{P}\left(\tilde{H}_n^{(1)}(\delta) \cap \left\{\sum_{m \in \check{\mathcal{U}}_k} I_{mk}^\star > 4T_k^\star/9\right\}\right) \leq \mathbb{P}\left(\tilde{H}_n^{(1)}(\delta) \cap \left\{\sum_{m \in \hat{\mathcal{U}}_k} I_{mk}^\star > N_k\right\}\right) \\
&\leq \mathbb{P}\left(\tilde{H}_n^{(1)}(\delta) \cap \left\{\sum_{m \in \hat{\mathcal{U}}_k} \mathbb{1}_{[2\gamma/3,\infty)}\left(\hat{\Delta}_m^{(k)}\left(X_m, W_2, V_{\check{m}_k}^\star\right)\right) > N_k\right\}\right),
\end{aligned} \tag{68}$$

where this last inequality is by (67). To simplify notation, define $\tilde{Z}_k = \left(T_k^\star, \check{m}_k, W_1, W_2, V_{\check{m}_k}^\star\right)$. By Lemmas 43 and 44 (with $\beta = 3/32$, $\xi = 2\gamma/3$, $\alpha = 3/4$, and $\xi = \gamma/16$), since $n \geq n^{(i)}(\delta) \geq 24 \cdot \max\left\{\tau^{(iv)}(\gamma/16;\delta), \tau(3/32;\delta)\right\}$, on $\tilde{H}_n^{(1)}(\delta)$, $\forall m \in \hat{\mathcal{U}}_k$,

$$\begin{aligned}
p_{2\gamma/3}(k, \check{m}_k, m) &\leq \mathcal{P}\left(x : p_x(k, \check{m}_k) \geq \gamma/2\right) + \exp\left\{-\gamma^2 \tilde{M}(m)/256\right\} \\
&\leq \mathcal{P}\left(x : p_x(k, \check{m}_k) \geq \gamma/2\right) + \exp\left\{-\gamma^2 \tilde{M}(\check{m}_k)/256\right\} \\
&\leq \hat{\Delta}_{\check{m}_k}^{(k)}\left(W_1, W_2, V_{\check{m}_k}^\star\right).
\end{aligned}$$

Letting $\tilde{G}_n'(k)$ denote the event $p_{2\gamma/3}(k, \check{m}_k, m) \leq \hat{\Delta}_{\check{m}_k}^{(k)}\left(W_1, W_2, V_{\check{m}_k}^\star\right)$, we see that $\tilde{G}_n'(k) \supseteq \tilde{H}_n^{(1)}(\delta)$. Thus, since the $\mathbb{1}_{[2\gamma/3,\infty)}\left(\hat{\Delta}_m^{(k)}\left(X_m, W_2, V_{\check{m}_k}^\star\right)\right)$ variables are conditionally independent given $\tilde{Z}_k$ for $m \in \hat{\mathcal{U}}_k$, each with respective conditional distribution Bernoulli $\left(p_{2\gamma/3}(k, \check{m}_k, m)\right)$, the law of total probability and a Chernoff bound imply that (68) is at most

$$\begin{aligned}
&\mathbb{P}\left(\tilde{G}_n'(k) \cap \left\{\sum_{m \in \hat{\mathcal{U}}_k} \mathbb{1}_{[2\gamma/3,\infty)}\left(\hat{\Delta}_m^{(k)}\left(X_m, W_2, V_{\check{m}_k}^\star\right)\right) > N_k\right\}\right) \\
&= \mathbb{E}\left[\mathbb{P}\left(\sum_{m \in \hat{\mathcal{U}}_k} \mathbb{1}_{[2\gamma/3,\infty)}\left(\hat{\Delta}_m^{(k)}\left(X_m, W_2, V_{\check{m}_k}^\star\right)\right) > N_k \middle| \tilde{Z}_k\right) \cdot \mathbb{1}_{\tilde{G}_n'(k)}\right] \\
&\leq \mathbb{E}\left[\exp\left\{-\hat{\Delta}_{\check{m}_k}^{(k)}\left(W_1, W_2, V_{\check{m}_k}^\star\right)\left|\hat{\mathcal{U}}_k\right|/27\right\}\right] \leq \mathbb{E}\left[\exp\{-T_k^\star/162\}\right] \leq \exp\left\{-n/\left(243 \cdot 4^{k-1}\right)\right\},
\end{aligned}$$

where the last inequality is by Lemma 50. Thus, there exists $\tilde{G}_n(k)$ with $\mathbb{P}\left(\tilde{H}_n^{(1)}(\delta) \setminus \tilde{G}_n(k)\right) \leq \exp\left\{-n/\left(243 \cdot 4^{k-1}\right)\right\}$ such that, on $\tilde{H}_n^{(1)}(\delta) \cap \tilde{G}_n(k)$, we have $\sum_{m=\hat{m}_{k-1}+1}^{\hat{m}_k} I_{mk}^\star \leq 3T_k^\star/4$. Defining $\tilde{H}_n^{(i)}(\delta) = \tilde{H}_n^{(1)}(\delta) \cap \bigcap_{k=1}^{\tilde{d}_f} \tilde{G}_n(k)$, a union bound implies

$$\mathbb{P}\left(\tilde{H}_n^{(1)}(\delta) \setminus \tilde{H}_n^{(i)}(\delta)\right) \leq \tilde{d}_f \cdot \exp\left\{-n/\left(243 \cdot 4^{\tilde{d}_f-1}\right)\right\}, \tag{69}$$

and on $\tilde{H}_n^{(i)}(\delta)$, every $k \in \left\{1, \ldots, \tilde{d}_f\right\}$ has $\sum_{m=\hat{m}_{k-1}+1}^{\hat{m}_k} I_{mk}^\star \leq 3T_k^\star/4$. In particular, this means the $C_{mk}^\star$ factors are redundant in $Q_k^\star$, so that $\hat{t}_k = \sum_{m=\hat{m}_{k-1}+1}^{\hat{m}_k} I_{mk}^\star$.

To get the stated probability bound, a union bound implies that

$$\begin{aligned}
1 - \mathbb{P}\left(\tilde{H}_n^{(1)}(\delta)\right) &\leq \left(1 - \mathbb{P}\left(H_{\tilde{m}_n}(\delta)\right)\right) + \left(1 - \mathbb{P}\left(H_{\tilde{m}_n}^{(i)}\right)\right) + \mathbb{P}\left(H_{\tilde{m}_n}^{(i)} \setminus H_{\tilde{m}_n}^{(iii)}(\gamma/16)\right) \\
&\quad + \left(1 - \mathbb{P}\left(H_{\tilde{m}_n}^{(iv)}\right)\right) + \mathbb{P}\left(H_{\tilde{m}_n}^{(i)} \setminus G_{\tilde{m}_n}^{(i)}\right) \\
&\leq \delta + c^{(i)} \cdot \exp\left\{-\tilde{M}\left(\tilde{m}_n\right)/4\right\} \\
&\quad + c^{(iii)}(\gamma/16) \cdot \exp\left\{-\tilde{M}\left(\tilde{m}_n\right)\gamma^2/256\right\} + 3\tilde{d}_f \cdot \exp\left\{-2\tilde{m}_n\right\} \\
&\quad + 121\tilde{d}_f\tilde{\delta}_f^{-1} \cdot \exp\left\{-\tilde{M}\left(\tilde{m}_n\right)/60\right\} \\
&\leq \delta + \left(c^{(i)} + c^{(iii)}(\gamma/16) + 124\tilde{d}_f\tilde{\delta}_f^{-1}\right) \cdot \exp\left\{-\tilde{m}_n\tilde{\delta}_f\gamma^2/512\right\}.
\end{aligned} \tag{70}$$

Since $n \geq n^{(i)}(\delta) \geq 24$, we have $\tilde{m}_n \geq n/48$, so that summing (69) and (70) gives us

$$1 - \mathbb{P}\left(\tilde{H}_n^{(i)}(\delta)\right) \leq \delta + \left(c^{(i)} + c^{(iii)}(\gamma/16) + 125\tilde{d}_f\tilde{\delta}_f^{-1}\right) \cdot \exp\left\{-n\tilde{\delta}_f\gamma^2/\left(512 \cdot 48 \cdot 4^{\tilde{d}_f-1}\right)\right\}. \tag{71}$$

Finally, note that we have chosen $n^{(i)}(\delta)$ sufficiently large so that (71) is at most 2δ. ■

The next lemma indicates that the redundancy of the "$t < \lfloor 3T_k/4 \rfloor$" constraint, just established in Lemma 51, implies that all $\hat{y}$ labels obtained while $k \leq \tilde{d}_f$ are consistent with the target function.

Lemma 52 *Consider running Meta-Algorithm 3 with a budget $n \in \mathbb{N}$, while f is the target function and $\mathcal{P}$ is the data distribution. There is an event $\tilde{H}_n^{(ii)}$ and $(\mathbb{C}, \mathcal{P}, f, \gamma)$-dependent constants $\tilde{c}_1^{(ii)}, \tilde{c}_2^{(ii)} \in [1, \infty)$ such that, for any $\delta \in (0,1)$, if $n \geq \tilde{c}_1^{(ii)} \ln\left(\tilde{c}_2^{(ii)}/\delta\right)$, then $\mathbb{P}\left(\tilde{H}_n^{(i)}(\delta) \setminus \tilde{H}_n^{(ii)}\right) \leq \delta$, and on $\tilde{H}_n^{(i)}(\delta) \cap \tilde{H}_n^{(ii)}$, we have $V^{(\tilde{d}_f)} = V_{\hat{m}_{\tilde{d}_f}} = V_{\hat{m}_{\tilde{d}_f}}^\star$.*

Proof Define $\tilde{c}_1^{(ii)} = \max\left\{\tilde{c}_1^{(i)}, \frac{192d}{r_{(1-\gamma)/6}}, \frac{2^{11}}{\tilde{\delta}_f^{1/3}}\right\}$, $\tilde{c}_2^{(ii)} = \max\left\{\tilde{c}_2^{(i)}, \frac{8e}{r_{(1-\gamma)/6}}, c^{(ii)}, \exp\{\tau^*\}\right\}$, let $n^{(ii)}(\delta) = \tilde{c}_1^{(ii)} \ln\left(\tilde{c}_2^{(ii)}/\delta\right)$, suppose $n \geq n^{(ii)}(\delta)$, and define the event $\tilde{H}_n^{(ii)} = H_{\tilde{m}_n}^{(ii)}$.

By Lemma 41, since $n \geq n^{(ii)}(\delta) \geq 24 \cdot \max\{\tau((1-\gamma)/6; \delta), \tau^*\}$, on $\tilde{H}_n^{(i)}(\delta) \cap \tilde{H}_n^{(ii)}$, $\forall m \in \mathbb{N}$ and $k \in \left\{1, \ldots, \tilde{d}_f\right\}$ with either $k = 1$ or $m > \tilde{m}_n$,

$$\hat{\Delta}_m^{(k)}\left(X_m, W_2, V_{m-1}^\star\right) < \gamma \Rightarrow \hat{\Gamma}_m^{(k)}\left(X_m, -f(X_m), W_2, V_{m-1}^\star\right) < \hat{\Gamma}_m^{(k)}\left(X_m, f(X_m), W_2, V_{m-1}^\star\right). \tag{72}$$

Recall that $\tilde{m}_n \leq \min\{\lceil T_1/4\rceil, 2^n\} = \lceil\lceil 2n/3\rceil/4\rceil$. Therefore, $V_{\tilde{m}_n}$ is obtained purely by $\tilde{m}_n$ executions of Step 8 while $k=1$. Thus, for every m obtained in Meta-Algorithm 3, either $k=1$ or $m > \tilde{m}_n$. We now proceed by induction on m. We already know $V_0 = \mathbb{C} = V_0^\star$, so this serves as our base case. Now consider some value $m \in \mathbb{N}$ obtained in Meta-Algorithm 3 while $k \leq \tilde{d}_f$, and suppose every $m' < m$ has $V_{m'} = V_{m'}^\star$. But this means that $T_k = T_k^\star$ and the value of t upon obtaining this particular m has $t \leq \sum_{\ell=\hat{m}_{k-1}+1}^{m-1} I_{\ell k}^\star$. In particular, if $\hat{\Delta}_m^{(k)}(X_m, W_2, V_{m-1}) \geq \gamma$, then $I_{mk}^\star = 1$, so that $t < \sum_{\ell=\hat{m}_{k-1}+1}^{m} I_{mk}^\star$; by Lemma 51, on $\tilde{H}_n^{(i)}(\delta) \cap \tilde{H}_n^{(ii)}$, $\sum_{\ell=\hat{m}_{k-1}+1}^{m} I_{mk}^\star \leq \sum_{\ell=\hat{m}_{k-1}+1}^{\hat{m}_k} I_{mk}^\star \leq 3T_k^\star/4$, so that $t < 3T_k^\star/4$, and therefore $\hat{y} = Y_m = f(X_m)$; this implies $V_m = V_m^\star$. On the other hand, on $\tilde{H}_n^{(i)}(\delta) \cap \tilde{H}_n^{(ii)}$, if $\hat{\Delta}_m^{(k)}(X_m, W_2, V_{m-1}) < \gamma$, then (72) implies

$$\hat{y} = \operatorname*{argmax}_{y\in\{-1,+1\}} \hat{\Gamma}_m^{(k)}(X_m, y, W_2, V_{m-1}) = f(X_m),$$

so that again $V_m = V_m^\star$. Thus, by the principle of induction, on $\tilde{H}_n^{(i)}(\delta) \cap \tilde{H}_n^{(ii)}$, for every $m \in \mathbb{N}$ obtained while $k \leq \tilde{d}_f$, we have $V_m = V_m^\star$; in particular, this implies $V^{(\tilde{d}_f)} = V_{\hat{m}_{\tilde{d}_f}} = V^\star_{\hat{m}_{\tilde{d}_f}}$. The bound on $\mathbb{P}\left(\tilde{H}_n^{(i)}(\delta) \setminus \tilde{H}_n^{(ii)}\right)$ then follows from Lemma 41, as we have chosen $n^{(ii)}(\delta)$ sufficiently large so that (27) (with $\tau = \tilde{m}_n$) is at most δ. ■

Lemma 53 *Consider running Meta-Algorithm 3 with a budget $n \in \mathbb{N}$, while f is the target function and $\mathcal{P}$ is the data distribution. There exist $(\mathbb{C}, \mathcal{P}, f, \gamma)$-dependent constants $\tilde{c}_1^{(iii)}, \tilde{c}_2^{(iii)} \in [1,\infty)$ such that, for any $\delta \in (0, e^{-3})$, $\lambda \in [1,\infty)$, and $n \in \mathbb{N}$, there exists an event $\tilde{H}_n^{(iii)}(\delta,\lambda)$ having $\mathbb{P}\left(\tilde{H}_n^{(i)}(\delta) \cap \tilde{H}_n^{(ii)} \setminus \tilde{H}_n^{(iii)}(\delta,\lambda)\right) \leq \delta$ with the property that, if*

$$n \geq \tilde{c}_1^{(iii)} \tilde{\theta}_f(d/\lambda) \ln^2\left(\frac{\tilde{c}_2^{(iii)}\lambda}{\delta}\right),$$

then on $\tilde{H}_n^{(i)}(\delta) \cap \tilde{H}_n^{(ii)} \cap \tilde{H}_n^{(iii)}(\delta,\lambda)$, at the conclusion of Meta-Algorithm 3, $\left|\mathcal{L}_{\tilde{d}_f}\right| \geq \lambda$.

Proof Let $\tilde{c}_1^{(iii)} = \max\left\{\tilde{c}_1^{(i)}, \tilde{c}_1^{(ii)}, \frac{d\cdot\tilde{d}_f\cdot 4^{10+2\tilde{d}_f}}{\gamma^3\tilde{\delta}_f^3}, \frac{192d}{r_{(3/32)}}\right\}$, $\tilde{c}_2^{(iii)} = \max\left\{\tilde{c}_2^{(i)}, \tilde{c}_2^{(ii)}, \frac{8e}{r_{(3/32)}}\right\}$, fix any $\delta \in (0, e^{-3})$, $\lambda \in [1,\infty)$, let $n^{(iii)}(\delta,\lambda) = \tilde{c}_1^{(iii)}\tilde{\theta}_f(d/\lambda)\ln^2(\tilde{c}_2^{(iii)}\lambda/\delta)$, and suppose $n \geq n^{(iii)}(\delta,\lambda)$.

Define a sequence $\ell_i = 2^i$ for integers $i \geq 0$, and let $\hat{\imath} = \left\lceil \log_2\left(4^{2+\tilde{d}_f}\lambda/\gamma\tilde{\delta}_f\right)\right\rceil$. Also define $\tilde{\phi}(m,\delta,\lambda) = \max\{\phi(m;\delta/2\hat{\imath}), d/\lambda\}$, where ϕ is defined in Lemma 29. Then define the events

$$\tilde{H}^{(3)}(\delta,\lambda) = \bigcap_{i=1}^{\hat{\imath}} H_{\ell_i}(\delta/2\hat{\imath}), \quad \tilde{H}_n^{(iii)}(\delta,\lambda) = \tilde{H}^{(3)}(\delta,\lambda) \cap \left\{\check{m}_{\tilde{d}_f} \geq \ell_{\hat{\imath}}\right\}.$$

Note that $\hat{\imath} \leq n$, so that $\ell_{\hat{\imath}} \leq 2^n$, and therefore the truncation in the definition of $\check{m}_{\tilde{d}_f}$, which enforces $\check{m}_{\tilde{d}_f} \leq \max\left\{\tilde{d}_f \cdot 2^n + 1, \hat{m}_{k-1}\right\}$, will never be a factor in whether or not $\check{m}_{\tilde{d}_f} \geq \ell_{\hat{\imath}}$ is satisfied.

Since $n \geq n^{(iii)}(\lambda,\delta) \geq \tilde{c}_1^{(ii)} \ln\left(\tilde{c}_2^{(ii)}/\delta\right)$, Lemma 52 implies that on $\tilde{H}_n^{(i)}(\delta) \cap \tilde{H}_n^{(ii)}$, $V_{\hat{m}_{\tilde{d}_f}} = V^\star_{\hat{m}_{\tilde{d}_f}}$. Recall that this implies that all $\hat{y}$ values obtained while $m \leq \hat{m}_{\tilde{d}_f}$ are consistent with their respective

$f(X_m)$ values, so that every such m has $V_m = V^\star_m$ as well. In particular, $V_{\check{m}_{\tilde{d}_f}} = V^\star_{\check{m}_{\tilde{d}_f}}$. Also note that $n^{(iii)}(\delta,\lambda) \geq 24 \cdot \tau^{(iv)}(\gamma/16;\delta)$, so that $\tau^{(iv)}(\gamma/16;\delta) \leq \tilde{m}_n$, and recall we always have $\tilde{m}_n \leq \check{m}_{\tilde{d}_f}$. Thus, on $\tilde{H}^{(i)}_n(\delta) \cap \tilde{H}^{(ii)}_n \cap \tilde{H}^{(iii)}_n(\delta,\lambda)$, (taking $\hat{\Delta}^{(k)}$ as in Meta-Algorithm 3)

$$
\begin{aligned}
\hat{\Delta}^{(\tilde{d}_f)} &= \hat{\Delta}^{(\tilde{d}_f)}_{\check{m}_{\tilde{d}_f}}\left(W_1, W_2, V^\star_{\check{m}_{\tilde{d}_f}}\right) && \text{(Lemma 52)} \\
&\leq \mathcal{P}\left(x : p_x\left(\tilde{d}_f, \check{m}_{\tilde{d}_f}\right) \geq \gamma/8\right) + 4\check{m}^{-1}_{\tilde{d}_f} && \text{(Lemma 44)} \\
&\leq \frac{8\mathcal{P}^{\tilde{d}_f}\left(\mathcal{S}^{\tilde{d}_f}\left(V^\star_{\check{m}_{\tilde{d}_f}}\right)\right)}{\gamma\mathcal{P}^{\tilde{d}_f-1}\left(\mathcal{S}^{\tilde{d}_f-1}\left(V^\star_{\check{m}_{\tilde{d}_f}}\right)\right)} + 4\check{m}^{-1}_{\tilde{d}_f} && \text{(Markov's ineq.)} \\
&\leq \left(8/\gamma\tilde{\delta}_f\right)\mathcal{P}^{\tilde{d}_f}\left(\mathcal{S}^{\tilde{d}_f}\left(V^\star_{\check{m}_{\tilde{d}_f}}\right)\right) + 4\check{m}^{-1}_{\tilde{d}_f} && \text{(Lemma 35)} \\
&\leq \left(8/\gamma\tilde{\delta}_f\right)\mathcal{P}^{\tilde{d}_f}\left(\mathcal{S}^{\tilde{d}_f}\left(V^\star_{\ell_{\hat{\imath}}}\right)\right) + 4\ell^{-1}_{\hat{\imath}} && \text{(defn of } \tilde{H}^{(iii)}_n(\delta,\lambda)\text{)} \\
&\leq \left(8/\gamma\tilde{\delta}_f\right)\mathcal{P}^{\tilde{d}_f}\left(\mathcal{S}^{\tilde{d}_f}\left(\mathrm{B}\left(f, \tilde{\phi}\left(\ell_{\hat{\imath}},\delta,\lambda\right)\right)\right)\right) + 4\ell^{-1}_{\hat{\imath}} && \text{(Lemma 29)} \\
&\leq \left(8/\gamma\tilde{\delta}_f\right)\tilde{\theta}_f(d/\lambda)\tilde{\phi}\left(\ell_{\hat{\imath}},\delta,\lambda\right) + 4\ell^{-1}_{\hat{\imath}} && \text{(defn of } \tilde{\theta}_f(d/\lambda)\text{)} \\
&\leq \left(12/\gamma\tilde{\delta}_f\right)\tilde{\theta}_f(d/\lambda)\tilde{\phi}\left(\ell_{\hat{\imath}},\delta,\lambda\right) && (\tilde{\phi}\left(\ell_{\hat{\imath}},\delta,\lambda\right) \geq \ell^{-1}_{\hat{\imath}}) \\
&= \frac{12\tilde{\theta}_f(d/\lambda)}{\gamma\tilde{\delta}_f}\max\left\{2\frac{d\ln\left(2e\max\left\{\ell_{\hat{\imath}},d\right\}/d\right)+\ln\left(4\hat{\imath}/\delta\right)}{\ell_{\hat{\imath}}}, d/\lambda\right\}. && (73)
\end{aligned}
$$

Plugging in the definition of $\hat{\imath}$ and $\ell_{\hat{\imath}}$,

$$
\frac{d\ln\left(2e\max\left\{\ell_{\hat{\imath}},d\right\}/d\right)+\ln\left(4\hat{\imath}/\delta\right)}{\ell_{\hat{\imath}}} \leq (d/\lambda)\gamma\tilde{\delta}_f 4^{-1-\tilde{d}_f}\ln\left(4^{1+\tilde{d}_f}\lambda/\delta\gamma\tilde{\delta}_f\right) \leq (d/\lambda)\ln\left(\lambda/\delta\right).
$$

Therefore, (73) is at most $24\tilde{\theta}_f(d/\lambda)(d/\lambda)\ln\left(\lambda/\delta\right)/\gamma\tilde{\delta}_f$. Thus, since

$$
n^{(iii)}(\delta,\lambda) \geq \max\left\{\tilde{c}^{(i)}_1\ln\left(\tilde{c}^{(i)}_2/\delta\right), \tilde{c}^{(ii)}_1\ln\left(\tilde{c}^{(ii)}_2/\delta\right)\right\},
$$

Lemmas 51 and 52 imply that on $\tilde{H}^{(i)}_n(\delta) \cap \tilde{H}^{(ii)}_n \cap \tilde{H}^{(iii)}_n(\delta,\lambda)$,

$$
\begin{aligned}
\left|\mathcal{L}_{\tilde{d}_f}\right| = \left\lfloor T^\star_{\tilde{d}_f}/\left(3\hat{\Delta}^{(\tilde{d}_f)}\right)\right\rfloor &\geq \left\lfloor 4^{1-\tilde{d}_f}2n/\left(9\hat{\Delta}^{(\tilde{d}_f)}\right)\right\rfloor \\
&\geq \frac{4^{1-\tilde{d}_f}\gamma\tilde{\delta}_f n}{9\cdot 24\cdot\tilde{\theta}_f(d/\lambda)(d/\lambda)\ln\left(\lambda/\delta\right)} \geq \lambda\ln(\lambda/\delta) \geq \lambda.
\end{aligned}
$$

Now we turn to bounding $\mathbb{P}\left(\tilde{H}^{(i)}_n(\delta) \cap \tilde{H}^{(ii)}_n \setminus \tilde{H}^{(iii)}_n(\delta,\lambda)\right)$. By a union bound, we have

$$
1 - \mathbb{P}\left(\tilde{H}^{(3)}(\delta,\lambda)\right) \leq \sum_{i=1}^{\hat{\imath}}\left(1 - \mathbb{P}\left(H_{\ell_i}\left(\delta/2\hat{\imath}\right)\right)\right) \leq \delta/2. \tag{74}
$$

Thus, it remains only to bound $\mathbb{P}\left(\tilde{H}_n^{(i)}(\delta)\cap\tilde{H}_n^{(ii)}\cap\tilde{H}^{(3)}(\delta,\lambda)\cap\left\{\check{m}_{\tilde{d}_f}<\ell_i\right\}\right)$.

For each $i\in\{0,1,\ldots,\hat{\imath}-1\}$, let $\check{Q}_i=\left|\left\{m\in(\ell_i,\ell_{i+1}]\cap\check{\mathcal{U}}_{\tilde{d}_f}:I^\star_{m\tilde{d}_f}=1\right\}\right|$. Now consider the set $\mathcal{I}$ of all $i\in\{0,1,\ldots,\hat{\imath}-1\}$ with $\ell_i\geq\tilde{m}_n$ and $(\ell_i,\ell_{i+1}]\cap\check{\mathcal{U}}_{\tilde{d}_f}\neq\emptyset$. Note that $n^{(iii)}(\delta,\lambda)\geq 48$, so that $\ell_0<\tilde{m}_n$. Fix any $i\in\mathcal{I}$. Since $n^{(iii)}(\lambda,\delta)\geq 24\cdot\tau(1/6;\delta)$, we have $\tilde{m}_n\geq\tau(1/6;\delta)$, so that Lemma 49 implies that on $\tilde{H}_n^{(i)}(\delta)\cap\tilde{H}_n^{(ii)}\cap\tilde{H}^{(3)}(\delta,\lambda)$, letting $Q=2\cdot 4^{6+\tilde{d}_f}\left(d/\gamma^2\tilde{\delta}_f^2\right)\tilde{\theta}_f(d/\lambda)\ln(\lambda/\delta)$,

$$\mathbb{P}\left(\tilde{H}_n^{(i)}(\delta)\cap\tilde{H}_n^{(ii)}\cap\tilde{H}^{(3)}(\delta,\lambda)\cap\left\{\check{Q}_i>Q\right\}\middle|W_2,V^\star_{\ell_i}\right)$$
$$\leq\mathbb{P}\left(\left|\left\{m\in(\ell_i,\ell_{i+1}]\cap\mathbb{N}:\hat{\Delta}_m^{(\tilde{d}_f)}\left(X_m,W_2,V^\star_{\ell_i}\right)\geq 2\gamma/3\right\}\right|>Q\middle|W_2,V^\star_{\ell_i}\right). \quad (75)$$

For $m>\ell_i$, the variables $\mathbb{1}_{[2\gamma/3,\infty)}\left(\hat{\Delta}_m^{(\tilde{d}_f)}\left(X_m,W_2,V^\star_{\ell_i}\right)\right)$ are conditionally (given $W_2,V^\star_{\ell_i}$) independent, each with respective conditional distribution Bernoulli with mean $p_{2\gamma/3}\left(\tilde{d}_f,\ell_i,m\right)$. Since $n^{(iii)}(\delta,\lambda)\geq 24\cdot\tau(3/32;\delta)$, we have $\tilde{m}_n\geq\tau(3/32;\delta)$, so that Lemma 43 (with $\xi=2\gamma/3$, $\alpha=3/4$, and $\beta=3/32$) implies that on $\tilde{H}_n^{(i)}(\delta)\cap\tilde{H}_n^{(ii)}\cap\tilde{H}^{(3)}(\delta,\lambda)$, each of these m values has

$$\begin{aligned}
p_{2\gamma/3}\left(\tilde{d}_f,\ell_i,m\right)&\leq\mathcal{P}\left(x:p_x\left(\tilde{d}_f,\ell_i\right)\geq\gamma/2\right)+\exp\left\{-\tilde{M}(m)\gamma^2/256\right\}\\
&\leq\frac{2\mathcal{P}^{\tilde{d}_f}\left(\mathcal{S}^{\tilde{d}_f}\left(V^\star_{\ell_i}\right)\right)}{\gamma\mathcal{P}^{\tilde{d}_f-1}\left(\mathcal{S}^{\tilde{d}_f-1}\left(V^\star_{\ell_i}\right)\right)}+\exp\left\{-\tilde{M}(\ell_i)\gamma^2/256\right\} && \text{(Markov's ineq.)}\\
&\leq\left(2/\gamma\tilde{\delta}_f\right)\mathcal{P}^{\tilde{d}_f}\left(\mathcal{S}^{\tilde{d}_f}\left(V^\star_{\ell_i}\right)\right)+\exp\left\{-\tilde{M}(\ell_i)\gamma^2/256\right\} && \text{(Lemma 35)}\\
&\leq\left(2/\gamma\tilde{\delta}_f\right)\mathcal{P}^{\tilde{d}_f}\left(\mathcal{S}^{\tilde{d}_f}\left(\mathrm{B}\left(f,\tilde{\phi}(\ell_i,\delta,\lambda)\right)\right)\right)+\exp\left\{-\tilde{M}(\ell_i)\gamma^2/256\right\} && \text{(Lemma 29)}\\
&\leq\left(2/\gamma\tilde{\delta}_f\right)\tilde{\theta}_f(d/\lambda)\tilde{\phi}(\ell_i,\delta,\lambda)+\exp\left\{-\tilde{M}(\ell_i)\gamma^2/256\right\} && \text{(defn of } \tilde{\theta}_f(d/\lambda)).
\end{aligned}$$

Denote the expression in this last line by p_i, and let $\mathbf{B}(\ell_i,p_i)$ be a Binomial(ℓ_i,p_i) random variable. Noting that $\ell_{i+1}-\ell_i=\ell_i$, we have that on $\tilde{H}_n^{(i)}(\delta)\cap\tilde{H}_n^{(ii)}\cap\tilde{H}^{(3)}(\delta,\lambda)$, (75) is at most $\mathbb{P}\left(\mathbf{B}(\ell_i,p_i)>Q\right)$. Next, note that

$$\ell_i p_i=(2/\gamma\tilde{\delta}_f)\tilde{\theta}_f(d/\lambda)\ell_i\tilde{\phi}(\ell_i,\delta,\lambda)+\ell_i\cdot\exp\left\{-\ell_i^3\tilde{\delta}_f\gamma^2/512\right\}.$$

Since $u\cdot\exp\left\{-u^3\right\}\leq(3e)^{-1/3}$ for any u, letting $u=\ell_i\tilde{\delta}_f\gamma/8$ we have

$$\ell_i\cdot\exp\left\{-\ell_i^3\tilde{\delta}_f\gamma^2/512\right\}\leq\left(8/\gamma\tilde{\delta}_f\right)u\cdot\exp\left\{-u^3\right\}\leq 8/\left(\gamma\tilde{\delta}_f(3e)^{1/3}\right)\leq 4/\gamma\tilde{\delta}_f.$$

Therefore, since $\tilde{\phi}(\ell_i,\delta,\lambda) \geq \ell_i^{-1}$, we have that $\ell_i p_i$ is at most

$$\begin{aligned}
\frac{6}{\gamma\tilde{\delta}_f}\tilde{\theta}_f(d/\lambda)\ell_i\tilde{\phi}(\ell_i,\delta,\lambda) &\leq \frac{6}{\gamma\tilde{\delta}_f}\tilde{\theta}_f(d/\lambda)\max\left\{2d\ln\left(2e\ell_{\hat{\imath}}\right)+2\ln\left(\frac{4\hat{\imath}}{\delta}\right),\ell_{\hat{\imath}}d/\lambda\right\} \\
&\leq \frac{6}{\gamma\tilde{\delta}_f}\tilde{\theta}_f(d/\lambda)\max\left\{2d\ln\left(\frac{4^{3+\tilde{d}_f}e\lambda}{\gamma\tilde{\delta}_f}\right)+2\ln\left(\frac{4^{3+\tilde{d}_f}2\lambda}{\gamma\tilde{\delta}_f\delta}\right),\frac{d4^{3+\tilde{d}_f}}{\gamma\tilde{\delta}_f}\right\} \\
&\leq \frac{6}{\gamma\tilde{\delta}_f}\tilde{\theta}_f(d/\lambda)\max\left\{4d\ln\left(\frac{4^{3+\tilde{d}_f}\lambda}{\gamma\tilde{\delta}_f\delta}\right),\frac{d4^{3+\tilde{d}_f}}{\gamma\tilde{\delta}_f}\right\} \\
&\leq \frac{6}{\gamma\tilde{\delta}_f}\tilde{\theta}_f(d/\lambda)\cdot\frac{d4^{4+\tilde{d}_f}}{\gamma\tilde{\delta}_f}\ln\left(\frac{\lambda}{\delta}\right) \leq \frac{4^{6+\tilde{d}_f}d}{\gamma^2\tilde{\delta}_f^2}\tilde{\theta}_f(d/\lambda)\ln\left(\frac{\lambda}{\delta}\right) = Q/2.
\end{aligned}$$

Therefore, a Chernoff bound implies $\mathbb{P}\left(\mathbf{B}(\ell_i,p_i) > Q\right) \leq \exp\left\{-Q/6\right\} \leq \delta/2\hat{\imath}$, so that on $\tilde{H}_n^{(i)}(\delta)\cap\tilde{H}_n^{(ii)}\cap\tilde{H}^{(3)}(\delta,\lambda)$, (75) is at most $\delta/2\hat{\imath}$. The law of total probability implies there exists an event $\tilde{H}_n^{(4)}(i,\delta,\lambda)$ with $\mathbb{P}\left(\tilde{H}_n^{(i)}(\delta)\cap\tilde{H}_n^{(ii)}\cap\tilde{H}^{(3)}(\delta,\lambda)\setminus\tilde{H}_n^{(4)}(i,\delta,\lambda)\right) \leq \delta/2\hat{\imath}$ such that, on $\tilde{H}_n^{(i)}(\delta)\cap\tilde{H}_n^{(ii)}\cap\tilde{H}^{(3)}(\delta,\lambda)\cap\tilde{H}_n^{(4)}(i,\delta,\lambda)$, $\check{Q}_i \leq Q$.

Note that

$$\begin{aligned}
\hat{\imath}Q &\leq \log_2\left(4^{2+\tilde{d}_f}\lambda/\gamma\tilde{\delta}_f\right)\cdot 4^{7+\tilde{d}_f}\left(d/\gamma^2\tilde{\delta}_f^2\right)\tilde{\theta}_f(d/\lambda)\ln(\lambda/\delta) \\
&\leq \left(\tilde{d}_f 4^{9+\tilde{d}_f}/\gamma^3\tilde{\delta}_f^3\right)d\tilde{\theta}_f(d/\lambda)\ln^2(\lambda/\delta) \leq 4^{1-\tilde{d}_f}n/12. \qquad (76)
\end{aligned}$$

Since $\sum_{m\leq 2\tilde{m}_n} I^\star_{m\tilde{d}_f} \leq n/12$, if $\tilde{d}_f = 1$ then (76) implies that on the event $\tilde{H}_n^{(i)}(\delta)\cap\tilde{H}_n^{(ii)}\cap\tilde{H}^{(3)}(\delta,\lambda)\cap\bigcap_{i\in\mathcal{I}}\tilde{H}_n^{(4)}(i,\delta,\lambda)$, $\sum_{m\leq\ell_{\hat{\imath}}} I^\star_{m1} \leq n/12+\sum_{i\in\mathcal{I}}\check{Q}_i \leq n/12+\hat{\imath}Q \leq n/6 \leq \lceil T_1^\star/4\rceil$, so that $\check{m}_1 \geq \ell_{\hat{\imath}}$. Otherwise, if $\tilde{d}_f > 1$, then every $m \in \check{\mathcal{U}}_{\tilde{d}_f}$ has $m > 2\tilde{m}_n$, so that $\sum_{i\leq\hat{\imath}}\check{Q}_i = \sum_{i\in\mathcal{I}}\check{Q}_i$; thus, on $\tilde{H}_n^{(i)}(\delta)\cap\tilde{H}_n^{(ii)}\cap\tilde{H}^{(3)}(\delta,\lambda)\cap\bigcap_{i\in\mathcal{I}}\tilde{H}_n^{(4)}(i,\delta,\lambda)$, $\sum_{i\in\mathcal{I}}\check{Q}_i \leq \hat{\imath}Q \leq 4^{1-\tilde{d}_f}n/12$; Lemma 50 implies $4^{1-\tilde{d}_f}n/12 \leq \left\lceil T^\star_{\tilde{d}_f}/4\right\rceil$, so that again we have $\check{m}_{\tilde{d}_f} \geq \ell_{\hat{\imath}}$. Combined with a union bound, this implies

$$\begin{aligned}
&\mathbb{P}\left(\tilde{H}_n^{(i)}(\delta)\cap\tilde{H}_n^{(ii)}\cap\tilde{H}^{(3)}(\delta,\lambda)\cap\left\{\check{m}_{\tilde{d}_f} < \ell_{\hat{\imath}}\right\}\right) \\
&\quad\leq \mathbb{P}\left(\tilde{H}_n^{(i)}(\delta)\cap\tilde{H}_n^{(ii)}\cap\tilde{H}^{(3)}(\delta,\lambda)\setminus\bigcap_{i\in\mathcal{I}}\tilde{H}_n^{(4)}(i,\delta,\lambda)\right) \\
&\quad\leq \sum_{i\in\mathcal{I}}\mathbb{P}\left(\tilde{H}_n^{(i)}(\delta)\cap\tilde{H}_n^{(ii)}\cap\tilde{H}^{(3)}(\delta,\lambda)\setminus\tilde{H}_n^{(4)}(i,\delta,\lambda)\right) \leq \delta/2. \qquad (77)
\end{aligned}$$

Therefore, $\mathbb{P}\left(\tilde{H}_n^{(i)}(\delta)\cap\tilde{H}_n^{(ii)}\setminus\tilde{H}_n^{(iii)}(\delta,\lambda)\right) \leq \delta$, obtained by summing (77) and (74). ■

Proof [Theorem 16] If $\Lambda_p(\varepsilon/4,f,\mathcal{P}) = \infty$ then the result trivially holds. Otherwise, suppose $\varepsilon \in (0,10e^{-3})$, let $\delta = \varepsilon/10$, $\lambda = \Lambda_p(\varepsilon/4,f,\mathcal{P})$, $\tilde{c}_2 = \max\left\{10\tilde{c}_2^{(i)},10\tilde{c}_2^{(ii)},10\tilde{c}_2^{(iii)},10e(d+1)\right\}$, and $\tilde{c}_1 = \max\left\{\tilde{c}_1^{(i)},\tilde{c}_1^{(ii)},\tilde{c}_1^{(iii)},2\cdot 6^3(d+1)\tilde{d}\ln(e(d+1))\right\}$, and consider running Meta-Algorithm

3 with passive algorithm $\mathcal{A}_p$ and budget $n \geq \tilde{c}_1 \tilde{\theta}_f(d/\lambda)\ln^2(\tilde{c}_2\lambda/\varepsilon)$, while f is the target function and $\mathcal{P}$ is the data distribution. On the event $\tilde{H}_n^{(i)}(\delta) \cap \tilde{H}_n^{(ii)} \cap \tilde{H}_n^{(iii)}(\delta,\lambda)$, Lemma 53 implies $\left|\mathcal{L}_{\tilde{d}_f}\right| \geq \lambda$, while Lemma 52 implies $V^{(\tilde{d}_f)} = V^\star_{\hat{m}_{\tilde{d}_f}}$; recalling that Lemma 35 implies that $V^\star_{\hat{m}_{\tilde{d}_f}} \neq \emptyset$ on this event, we must have $\mathrm{er}_{\mathcal{L}_{\tilde{d}_f}}(f) = 0$. Furthermore, if $\hat{h}$ is the classifier returned by Meta-Algorithm 3, then Lemma 34 implies that $\mathrm{er}(\hat{h})$ is at most $2\mathrm{er}(\mathcal{A}_p(\mathcal{L}_{\tilde{d}_f}))$, on a high probability event (call it $\hat{E}_2$ in this context). Letting $\hat{E}_3(\delta) = \hat{E}_2 \cap \tilde{H}_n^{(i)}(\delta) \cap \tilde{H}_n^{(ii)} \cap \tilde{H}_n^{(iii)}(\delta,\lambda)$, a union bound implies the total failure probability $1 - \mathbb{P}(\hat{E}_3(\delta))$ from all of these events is at most $4\delta + e(d+1)\cdot\exp\left\{-\lfloor n/3\rfloor / \left(72\tilde{d}_f(d+1)\ln(e(d+1))\right)\right\} \leq 5\delta = \varepsilon/2$. Since, for $\ell \in \mathbb{N}$ with $\mathbb{P}\left(\left|\mathcal{L}_{\tilde{d}_f}\right| = \ell\right) > 0$, the sequence of X_m values appearing in $\mathcal{L}_{\tilde{d}_f}$ are conditionally distributed as $\mathcal{P}^\ell$ given $|\mathcal{L}_{\tilde{d}_f}| = \ell$, and this is the same as the (unconditional) distribution of $\{X_1, X_2, \ldots, X_\ell\}$, we have that

$$\begin{aligned}\mathbb{E}\left[\mathrm{er}\left(\hat{h}\right)\right] &\leq \mathbb{E}\left[2\mathrm{er}\left(\mathcal{A}_p\left(\mathcal{L}_{\tilde{d}_f}\right)\right)\mathbb{1}_{\hat{E}_3(\delta)}\right] + \varepsilon/2 = \mathbb{E}\left[\mathbb{E}\left[2\mathrm{er}\left(\mathcal{A}_p\left(\mathcal{L}_{\tilde{d}_f}\right)\right)\mathbb{1}_{\hat{E}_3(\delta)}\middle|\left|\mathcal{L}_{\tilde{d}_f}\right|\right]\right] + \varepsilon/2 \\ &\leq 2 \sup_{\ell \geq \Lambda_p(\varepsilon/4, f, \mathcal{P})} \mathbb{E}\left[\mathrm{er}(\mathcal{A}_p(\mathcal{Z}_\ell))\right] + \varepsilon/2 \leq \varepsilon.\end{aligned}$$

To specialize to the specific variant of Meta-Algorithm 3 stated in Section 5.2, take $\gamma = 1/2$. ■

Appendix E. Proofs Related to Section 6: Agnostic Learning

This appendix contains the proofs of our results on learning with noise. Specifically, Appendix E.1 provides the proof of the counterexample from Theorem 22, demonstrating that there is no activizer for the $\check{\mathcal{A}}_p$ passive learning algorithm described in Section 6.2 in the agnostic case. Appendix E.2 presents the proof of Lemma 26 from Section 6.7, bounding the label complexity of Algorithm 5 under Condition 1. Finally, Appendix E.3 presents a proof of Theorem 28, demonstrating that any active learning algorithm can be modified to trivialize the misspecified model case. The notation used throughout Appendix E is taken from Section 6.

E.1 Proof of Theorem 22: Negative Result for Agnostic Activized Learning

It suffices to show that $\check{\mathcal{A}}_p$ achieves a label complexity Λ_p such that, for any label complexity Λ_a achieved by any active learning algorithm $\mathcal{A}_a$, there exists a distribution $\mathcal{P}_{XY}$ on $\mathcal{X} \times \{-1,+1\}$ such that $\mathcal{P}_{XY} \in \mathrm{Nontrivial}(\Lambda_p;\mathbb{C})$ and yet $\Lambda_a(\nu + c\varepsilon, \mathcal{P}_{XY}) \neq o\left(\Lambda_p(\nu+\varepsilon,\mathcal{P}_{XY})\right)$ for every constant $c \in (0,\infty)$. Specifically, we will show that there is a distribution $\mathcal{P}_{XY}$ for which $\Lambda_p(\nu+\varepsilon,\mathcal{P}_{XY}) = \Theta(1/\varepsilon)$ and $\Lambda_a(\nu+\varepsilon,\mathcal{P}_{XY}) \neq o(1/\varepsilon)$.

Let $\mathcal{P}(\{0\}) = 1/2$, and for any measurable $A \subseteq (0,1]$, $\mathcal{P}(A) = \lambda(A)/2$, where λ is Lebesgue measure. Let $\mathbb{D}$ be the family of distributions $\mathcal{P}_{XY}$ on $\mathcal{X} \times \{-1,+1\}$ characterized by the properties that the marginal distribution on $\mathcal{X}$ is $\mathcal{P}$, $\eta(0;\mathcal{P}_{XY}) \in (1/8, 3/8)$, and $\forall x \in (0,1]$,

$$\eta(x;\mathcal{P}_{XY}) = \eta(0;\mathcal{P}_{XY}) + (x/2)\cdot(1 - \eta(0;\mathcal{P}_{XY})).$$

Thus, $\eta(x;\mathcal{P}_{XY})$ is a linear function. For any $\mathcal{P}_{XY} \in \mathbb{D}$, since the point $z^* = \frac{1-2\eta(0;\mathcal{P}_{XY})}{1-\eta(0;\mathcal{P}_{XY})}$ has $\eta(z^*;\mathcal{P}_{XY}) = 1/2$, we see that $f = h_{z^*}$ is a Bayes optimal classifier. Furthermore, for any $\eta_0 \in$

$[1/8,3/8]$,

$$\left|\frac{1-2\eta_0}{1-\eta_0}-\frac{1-2\eta(0;\mathcal{P}_{XY})}{1-\eta(0;\mathcal{P}_{XY})}\right| = \frac{|\eta(0;\mathcal{P}_{XY})-\eta_0|}{(1-\eta_0)(1-\eta(0;\mathcal{P}_{XY}))},$$

and since $(1-\eta_0)(1-\eta(0;\mathcal{P}_{XY})) \in (25/64,49/64) \subset (1/3,1)$, the value $z = \frac{1-2\eta_0}{1-\eta_0}$ satisfies

$$|\eta_0-\eta(0;\mathcal{P}_{XY})| \le |z-z^*| \le 3|\eta_0-\eta(0;\mathcal{P}_{XY})|. \tag{78}$$

Also note that under $\mathcal{P}_{XY}$, since $(1-2\eta(0;\mathcal{P}_{XY})) = (1-\eta(0;\mathcal{P}_{XY}))z^*$, any $z \in (0,1)$ has

$$\begin{aligned}\mathrm{er}(h_z)-\mathrm{er}(h_{z^*}) &= \int_z^{z^*}\big(1-2\eta(x;\mathcal{P}_{XY})\big)\mathrm{d}x = \int_z^{z^*}\big(1-2\eta(0;\mathcal{P}_{XY})-x(1-\eta(0;\mathcal{P}_{XY}))\big)\mathrm{d}x \\ &= (1-\eta(0;\mathcal{P}_{XY}))\int_z^{z^*}(z^*-x)\,\mathrm{d}x = \frac{(1-\eta(0;\mathcal{P}_{XY}))}{2}(z^*-z)^2,\end{aligned}$$

so that

$$\frac{5}{16}(z-z^*)^2 \le \mathrm{er}(h_z)-\mathrm{er}(h_{z^*}) \le \frac{7}{16}(z-z^*)^2. \tag{79}$$

Finally, note that any $x,x' \in (0,1]$ with $|x-z^*| < |x'-z^*|$ has

$$|1-2\eta(x;\mathcal{P}_{XY})| = |x-z^*|(1-\eta(0;\mathcal{P}_{XY})) < |x'-z^*|(1-\eta(0;\mathcal{P}_{XY})) = |1-2\eta(x';\mathcal{P}_{XY})|.$$

Thus, for any $q \in (0,1/2]$, there exists $z'_q \in [0,1]$ such that $z^* \in [z'_q, z'_q+2q] \subseteq [0,1]$, and the classifier $h'_q(x) = h_{z^*}(x)\cdot\left(1-2\mathbb{1}_{(z'_q,z'_q+2q]}(x)\right)$ has $\mathrm{er}(h) \ge \mathrm{er}(h'_q)$ for every classifier h with $h(0) = -1$ and $\mathcal{P}(x : h(x) \ne h_{z^*}(x)) = q$. Noting that $\mathrm{er}(h'_q)-\mathrm{er}(h_{z^*}) = \left(\lim_{z\downarrow z'_q}\mathrm{er}(h_z)-\mathrm{er}(h_{z^*})\right) + \left(\mathrm{er}(h_{z'_q+2q})-\mathrm{er}(h_{z^*})\right)$, (79) implies that $\mathrm{er}(h'_q)-\mathrm{er}(h_{z^*}) \ge \frac{5}{16}\left(\left(z'_q-z^*\right)^2+\left(z'_q+2q-z^*\right)^2\right)$, and since $\max\{z^*-z'_q, z'_q+2q-z^*\} \ge q$, this is at least $\frac{5}{16}q^2$. In general, any h with $h(0) = +1$ has $\mathrm{er}(h)-\mathrm{er}(h_{z^*}) \ge 1/2-\eta(0;\mathcal{P}_{XY}) > 1/8 \ge (1/8)\mathcal{P}(x : h(x) \ne h_{z^*}(x))^2$. Combining these facts, we see that any classifier h has

$$\mathrm{er}(h)-\mathrm{er}(h_{z^*}) \ge (1/8)\mathcal{P}\left(x : h(x) \ne h_{z^*}(x)\right)^2. \tag{80}$$

Lemma 54 *The passive learning algorithm $\check{\mathcal{A}}_p$ achieves a label complexity Λ_p such that, for every $\mathcal{P}_{XY} \in \mathbb{D}$, $\Lambda_p(\nu+\varepsilon,\mathcal{P}_{XY}) = \Theta(1/\varepsilon)$.*

Proof Consider the values $\hat{\eta}_0$ and $\hat{z}$ from $\check{\mathcal{A}}_p(\mathcal{Z}_n)$ for some $n \in \mathbb{N}$. Combining (78) and (79), we have $\mathrm{er}(h_{\hat{z}})-\mathrm{er}(h_{z^*}) \le \frac{7}{16}(\hat{z}-z^*)^2 \le \frac{63}{16}(\hat{\eta}_0-\eta(0;\mathcal{P}_{XY}))^2 \le 4(\hat{\eta}_0-\eta(0;\mathcal{P}_{XY}))^2$. Let $N_n = |\{i \in \{1,\ldots,n\} : X_i = 0\}|$, and $\eta_0 = N_n^{-1}|\{i \in \{1,\ldots,n\} : X_i = 0, Y_i = +1\}|$ if $N_n > 0$, or $\eta_0 = 0$ if $N_n = 0$. Note that $\hat{\eta}_0 = \left(\eta_0 \vee \frac{1}{8}\right) \wedge \frac{3}{8}$, and since $\eta(0;\mathcal{P}_{XY}) \in (1/8,3/8)$, we have $|\hat{\eta}_0-\eta(0;\mathcal{P}_{XY})| \le |\eta_0-\eta(0;\mathcal{P}_{XY})|$. Therefore, for any $\mathcal{P}_{XY} \in \mathbb{D}$,

$$\begin{aligned}\mathbb{E}\left[\mathrm{er}(h_{\hat{z}})-\mathrm{er}(h_{z^*})\right] &\le 4\mathbb{E}\left[(\hat{\eta}_0-\eta(0;\mathcal{P}_{XY}))^2\right] \le 4\mathbb{E}\left[(\eta_0-\eta(0;\mathcal{P}_{XY}))^2\right] \\ &\le 4\mathbb{E}\left[\mathbb{E}\left[(\eta_0-\eta(0;\mathcal{P}_{XY}))^2\middle|N_n\right]\mathbb{1}_{[n/4,n]}(N_n)\right] + 4\mathbb{P}(N_n < n/4).\end{aligned} \tag{81}$$

By a Chernoff bound, $\mathbb{P}(N_n < n/4) \le \exp\{-n/16\}$, and since the conditional distribution of $N_n\eta_0$ given N_n is Binomial$(N_n,\eta(0;\mathcal{P}_{XY}))$, (81) is at most

$$4\mathbb{E}\left[\frac{1}{N_n \vee n/4}\eta(0;\mathcal{P}_{XY})(1-\eta(0;\mathcal{P}_{XY}))\right] + 4\cdot\exp\{-n/16\} \le 4\cdot\frac{4}{n}\cdot\frac{15}{64}+4\cdot\frac{16}{n} < \frac{68}{n}.$$

For any $n \geq \lceil 68/\varepsilon \rceil$, this is at most ε. Therefore, $\check{\mathcal{A}}_p$ achieves a label complexity Λ_p such that, for any $\mathcal{P}_{XY} \in \mathbb{D}$, $\Lambda_p(\nu+\varepsilon, \mathcal{P}_{XY}) = \lceil 68/\varepsilon \rceil = \Theta(1/\varepsilon)$. ■

Next we establish a corresponding lower bound for any active learning algorithm. Note that this requires more than a simple minimax lower bound, since we must have an asymptotic lower bound for a *fixed* $\mathcal{P}_{XY}$, rather than selecting a different $\mathcal{P}_{XY}$ for each ε value; this is akin to the *strong* minimax lower bounds proven by Antos and Lugosi (1998) for passive learning in the realizable case. For this, we proceed by reduction from the task of estimating a binomial mean; toward this end, the following lemma will be useful.

Lemma 55 *For any nonempty $(a,b) \subset [0,1]$, and any sequence of estimators $\hat{p}_n : \{0,1\}^n \to [0,1]$, there exists $p \in (a,b)$ such that, if $B_1, B_2, \ldots$ are independent* Bernoulli(p) *random variables, also independent from every $\hat{p}_n$, then $\mathbb{E}\left[(\hat{p}_n(B_1,\ldots,B_n) - p)^2\right] \neq o(1/n)$.*

Proof We first establish the claim when $a = 0$ and $b = 1$. For any $p \in [0,1]$, let $B_1(p), B_2(p), \ldots$ be i.i.d. Bernoulli(p) random variables, independent from any internal randomness of the $\hat{p}_n$ estimators. We proceed by reduction from hypothesis testing, for which there are known lower bounds. Specifically, it is known (e.g., Wald, 1945; Bar-Yossef, 2003) that for any $p, q \in (0,1)$, $\delta \in (0, e^{-1})$, any (possibly randomized) $\hat{q} : \{0,1\}^n \to \{p,q\}$, and any $n \in \mathbb{N}$,

$$n < \frac{(1-8\delta)\ln(1/8\delta)}{8\mathrm{KL}(p\|q)} \implies \max_{p^* \in \{p,q\}} \mathbb{P}\left(\hat{q}(B_1(p^*),\ldots,B_n(p^*)) \neq p^*\right) > \delta,$$

where $\mathrm{KL}(p\|q) = p\ln(p/q) + (1-p)\ln((1-p)/(1-q))$. It is also known (e.g., Poland and Hutter, 2006) that for $p, q \in [1/4, 3/4]$, $\mathrm{KL}(p\|q) \leq (8/3)(p-q)^2$. Combining this with the above fact, we have that for $p, q \in [1/4, 3/4]$,

$$\max_{p^* \in \{p,q\}} \mathbb{P}\left(\hat{q}(B_1(p^*),\ldots,B_n(p^*)) \neq p^*\right) \geq (1/16) \cdot \exp\left\{-128(p-q)^2 n/3\right\}. \tag{82}$$

Given the estimator $\hat{p}_n$ from the lemma statement, we construct a sequence of hypothesis tests as follows. For $i \in \mathbb{N}$, let $\alpha_i = \exp\left\{-2^i\right\}$ and $n_i = \left\lfloor 1/\alpha_i^2 \right\rfloor$. Define $p_0^* = 1/4$, and for $i \in \mathbb{N}$, inductively define $\hat{q}_i(b_1,\ldots,b_{n_i}) = \mathrm{argmin}_{p \in \{p_{i-1}^*, p_{i-1}^* + \alpha_i\}} |\hat{p}_{n_i}(b_1,\ldots,b_{n_i}) - p|$ for $b_1,\ldots,b_{n_i} \in \{0,1\}$, and $p_i^* = \mathrm{argmax}_{p \in \{p_{i-1}^*, p_{i-1}^* + \alpha_i\}} \mathbb{P}\left(\hat{q}_i(B_1(p),\ldots,B_{n_i}(p)) \neq p\right)$. Finally, define $p^* = \lim_{i\to\infty} p_i^*$. Note that $\forall i \in \mathbb{N}$, $p_i^* < 1/2$, $p_{i-1}^*, p_{i-1}^* + \alpha_i \in [1/4, 3/4]$, and $0 \leq p^* - p_i^* \leq \sum_{j=i+1}^{\infty} \alpha_j < 2\alpha_{i+1} = 2\alpha_i^2$. We generally have

$$\begin{aligned}
\mathbb{E}\left[(\hat{p}_{n_i}(B_1(p^*),\ldots,B_{n_i}(p^*)) - p^*)^2\right] &\geq \frac{1}{3}\mathbb{E}\left[(\hat{p}_{n_i}(B_1(p^*),\ldots,B_{n_i}(p^*)) - p_i^*)^2\right] - (p^* - p_i^*)^2 \\
&\geq \frac{1}{3}\mathbb{E}\left[(\hat{p}_{n_i}(B_1(p^*),\ldots,B_{n_i}(p^*)) - p_i^*)^2\right] - 4\alpha_i^4.
\end{aligned}$$

Furthermore, note that for any $m \in \{0,\ldots,n_i\}$,

$$\begin{aligned}
\frac{(p^*)^m(1-p^*)^{n_i-m}}{(p_i^*)^m(1-p_i^*)^{n_i-m}} &\geq \left(\frac{1-p^*}{1-p_i^*}\right)^{n_i} \geq \left(\frac{1-p_i^*-2\alpha_i^2}{1-p_i^*}\right)^{n_i} \\
&\geq \left(1-4\alpha_i^2\right)^{n_i} \geq \exp\left\{-8\alpha_i^2 n_i\right\} \geq e^{-8},
\end{aligned}$$

so that the probability mass function of $(B_1(p^*),\ldots,B_{n_i}(p^*))$ is never smaller than e^{-8} times that of $(B_1(p_i^*),\ldots,B_{n_i}(p_i^*))$, which implies (by the law of the unconscious statistician)

$$\mathbb{E}\left[(\hat{p}_{n_i}(B_1(p^*),\ldots,B_{n_i}(p^*))-p_i^*)^2\right] \geq e^{-8}\mathbb{E}\left[(\hat{p}_{n_i}(B_1(p_i^*),\ldots,B_{n_i}(p_i^*))-p_i^*)^2\right].$$

By a triangle inequality, we have

$$\mathbb{E}\left[(\hat{p}_{n_i}(B_1(p_i^*),\ldots,B_{n_i}(p_i^*))-p_i^*)^2\right] \geq \frac{\alpha_i^2}{4}\mathbb{P}\left(\hat{q}_i(B_1(p_i^*),\ldots,B_{n_i}(p_i^*)) \neq p_i^*\right).$$

By (82), this is at least

$$\frac{\alpha_i^2}{4}(1/16)\cdot\exp\left\{-128\alpha_i^2 n_i/3\right\} \geq 2^{-6}e^{-43}\alpha_i^2.$$

Combining the above, we have

$$\mathbb{E}\left[(\hat{p}_{n_i}(B_1(p^*),\ldots,B_{n_i}(p^*))-p^*)^2\right] \geq 3^{-1}2^{-6}e^{-51}\alpha_i^2 - 4\alpha_i^4 \geq 2^{-9}e^{-51}n_i^{-1} - 4n_i^{-2}.$$

For $i \geq 5$, this is larger than $2^{-11}e^{-51}n_i^{-1}$. Since n_i diverges as $i\to\infty$, we have that

$$\mathbb{E}\left[(\hat{p}_{n_i}(B_1(p^*),\ldots,B_{n_i}(p^*))-p^*)^2\right] \neq o(1/n),$$

which establishes the result for $a=0$ and $b=1$.

To extend this result to general nonempty ranges (a,b), we proceed by reduction from the above problem. Specifically, suppose $p'\in(0,1)$, and consider the following independent random variables (also independent from the $B_i(p')$ variables and $\hat{p}_n$ estimators). For each $i\in\mathbb{N}$, $C_{i1}\sim$ Bernoulli(a), $C_{i2}\sim$ Bernoulli$((b-a)/(1-a))$. Then for $b_i\in\{0,1\}$, define $B_i'(b_i)=\max\{C_{i1},C_{i2}\cdot b_i\}$. For any given $p'\in(0,1)$, the random variables $B_i'(B_i(p'))$ are i.i.d. Bernoulli(p), with $p=a+(b-a)p'\in(a,b)$ (which forms a bijection between $(0,1)$ and (a,b)). Defining $\hat{p}_n'(b_1,\ldots,b_n)=(\hat{p}_n(B_1'(b_1),\ldots,B_n'(b_n))-a)/(b-a)$, we have

$$\mathbb{E}\left[(\hat{p}_n(B_1(p),\ldots,B_n(p))-p)^2\right] = (b-a)^2\cdot\mathbb{E}\left[\left(\hat{p}_n'(B_1(p'),\ldots,B_n(p'))-p'\right)^2\right]. \tag{83}$$

We have already shown there exists a value of $p'\in(0,1)$ such that the right side of (83) is not $o(1/n)$. Therefore, the corresponding value of $p=a+(b-a)p'\in(a,b)$ has the left side of (83) not $o(1/n)$, which establishes the result. ■

We are now ready for the lower bound result for our setting.

Lemma 56 *For any label complexity Λ_a achieved by any active learning algorithm $\mathcal{A}_a$, there exists a $\mathcal{P}_{XY}\in\mathbb{D}$ such that $\Lambda_a(\nu+\varepsilon,\mathcal{P}_{XY})\neq o(1/\varepsilon)$.*

Proof The idea here is to reduce from the task of estimating the mean of iid Bernoulli trials, corresponding to the Y_i values. Specifically, consider any active learning algorithm $\mathcal{A}_a$; we use $\mathcal{A}_a$ to construct an estimator for the mean of iid Bernoulli trials as follows. Suppose we have $B_1,B_2,\ldots,B_n$ i.i.d. Bernoulli(p), for some $p\in(1/8,3/8)$ and $n\in\mathbb{N}$. We take the sequence of $X_1,X_2,\ldots$ random

variables i.i.d. with distribution $\mathcal{P}$ defined above (independent from the B_j variables). For each i, we additionally have a random variable C_i with conditional distribution Bernoulli$(X_i/2)$ given X_i, where the C_i are conditionally independent given the X_i sequence, and independent from the B_i sequence as well.

We run $\mathcal{A}_a$ with this sequence of X_i values. For the t^{th} label request made by the algorithm, say for the Y_i value corresponding to some X_i, if it has previously requested this Y_i already, then we simply repeat the same answer for Y_i again, and otherwise we return to the algorithm the value $2\max\{B_t, C_i\} - 1$ for Y_i. Note that in the latter case, the conditional distribution of $\max\{B_t, C_i\}$ is Bernoulli$(p + (1-p)X_i/2)$, given the X_i that $\mathcal{A}_a$ requests the label of; thus, the Y_i response has the same conditional distribution given X_i as it would have for the $\mathcal{P}_{XY} \in \mathbb{D}$ with $\eta(0; \mathcal{P}_{XY}) = p$ (i.e., $\eta(X_i; \mathcal{P}_{XY}) = p + (1-p)X_i/2$). Since this Y_i value is conditionally (given X_i) independent from the previously returned labels and X_j sequence, this is distributionally equivalent to running $\mathcal{A}_a$ under the $\mathcal{P}_{XY} \in \mathbb{D}$ with $\eta(0; \mathcal{P}_{XY}) = p$.

Let $\hat{h}_n$ be the classifier returned by $\mathcal{A}_a(n)$ in the above context, and let $\hat{z}_n$ denote the value of $z \in [2/5, 6/7]$ with minimum $\mathcal{P}(x : h_z(x) \neq \hat{h}_n(x))$. Then define $\hat{p}_n = \frac{1-\hat{z}_n}{2-\hat{z}_n} \in [1/8, 3/8]$ and $z^* = \frac{1-2p}{1-p} \in (2/5, 6/7)$. By a triangle inequality, we have $|\hat{z}_n - z^*| = 2\mathcal{P}(x : h_{\hat{z}_n}(x) \neq h_{z^*}(x)) \leq 4\mathcal{P}(x : \hat{h}_n(x) \neq h_{z^*}(x))$. Combining this with (80) and (78) implies that

$$\operatorname{er}(\hat{h}_n) - \operatorname{er}(h_{z^*}) \geq \frac{1}{8}\mathcal{P}\left(x : \hat{h}_n(x) \neq h_{z^*}(x)\right)^2 \geq \frac{1}{128}\left(\hat{z}_n - z^*\right)^2 \geq \frac{1}{128}\left(\hat{p}_n - p\right)^2. \tag{84}$$

In particular, by Lemma 55, we can choose $p \in (1/8, 3/8)$ so that $\mathbb{E}\left[\left(\hat{p}_n - p\right)^2\right] \neq o(1/n)$, which, by (84), implies $\mathbb{E}\left[\operatorname{er}(\hat{h}_n)\right] - \nu \neq o(1/n)$. This means there is an increasing infinite sequence of values $n_k \in \mathbb{N}$, and a constant $c \in (0, \infty)$ such that $\forall k \in \mathbb{N}, \mathbb{E}\left[\operatorname{er}(\hat{h}_{n_k})\right] - \nu \geq c/n_k$. Supposing $\mathcal{A}_a$ achieves label complexity Λ_a, and taking the values $\varepsilon_k = c/(2n_k)$, we have $\Lambda_a(\nu + \varepsilon_k, \mathcal{P}_{XY}) > n_k = c/(2\varepsilon_k)$. Since $\varepsilon_k > 0$ and approaches 0 as $k \to \infty$, we have $\Lambda_a(\nu + \varepsilon, \mathcal{P}_{XY}) \neq o(1/\varepsilon)$. ■

Proof [of Theorem 22] The result follows from Lemmas 54 and 56. ■

E.2 Proof of Lemma 26: Label Complexity of Algorithm 5

The proof of Lemma 26 essentially runs parallel to that of Theorem 16, with variants of each lemma from that proof adapted to the noise-robust Algorithm 5.

As before, in this section we will fix a particular joint distribution $\mathcal{P}_{XY}$ on $\mathcal{X} \times \{-1, +1\}$ with marginal $\mathcal{P}$ on $\mathcal{X}$, and then analyze the label complexity achieved by Algorithm 5 for that particular distribution. For our purposes, we will suppose $\mathcal{P}_{XY}$ satisfies Condition 1 for some finite parameters μ and κ. We also fix any $f \in \bigcap_{\varepsilon > 0} \operatorname{cl}(\mathbb{C}(\varepsilon))$. Furthermore, we will continue using the notation of Appendix B, such as $\mathcal{S}^k(\mathcal{H})$, etc., and in particular we continue to denote $V_m^\star = \{h \in \mathbb{C} : \forall \ell \leq m, h(X_\ell) = f(X_\ell)\}$ (though note that in this case, we may sometimes have $f(X_\ell) \neq Y_\ell$, so that $V_m^\star \neq \mathbb{C}[\mathcal{Z}_m]$). As in the above proofs, we will prove a slightly more general result in which the "1/2" threshold in Step 5 can be replaced by an arbitrary constant $\gamma \in (0, 1)$.

For the estimators $\hat{P}_{4m}$ used in the algorithm, we take the same definitions as in Appendix B.1. To be clear, we assume the sequences W_1 and W_2 mentioned there are independent from the entire

$(X_1,Y_1),(X_2,Y_2),\ldots$ sequence of data points; this is consistent with the earlier discussion of how these W_1 and W_2 sequences can be constructed in a preprocessing step.

We will consider running Algorithm 5 with label budget $n \in \mathbb{N}$ and confidence parameter $\delta \in (0,e^{-3})$, and analyze properties of the internal sets V_i. We will denote by $\hat{V}_i$, $\hat{\mathcal{L}}_i$, and $\hat{i}_k$, the *final* values of V_i, $\mathcal{L}_i$, and i_k, respectively, for each i and k in Algorithm 5. We also denote by $\hat{m}^{(k)}$ and $\hat{V}^{(k)}$ the final values of m and V_{i_k+1}, respectively, obtained while k has the specified value in Algorithm 5; $\hat{V}^{(k)}$ may be smaller than $\hat{V}_{\hat{i}_k}$ when $\hat{m}^{(k)}$ is not a power of 2. Additionally, define $\mathcal{L}_i^\star = \{(X_m,Y_m)\}_{m=2^{i-1}+1}^{2^i}$. After establishing a few results concerning these, we will show that for n satisfying the condition in Lemma 26, the conclusion of the lemma holds. First, we have a few auxiliary definitions. For $\mathcal{H} \subseteq \mathbb{C}$, and any $i \in \mathbb{N}$, define

$$\phi_i(\mathcal{H}) = \mathbb{E}\sup_{h_1,h_2\in\mathcal{H}} \left|\left(\mathrm{er}(h_1)-\mathrm{er}_{\mathcal{L}_i^\star}(h_1)\right)-\left(\mathrm{er}(h_2)-\mathrm{er}_{\mathcal{L}_i^\star}(h_2)\right)\right|$$

$$\text{and}\quad \tilde{U}_i(\mathcal{H},\delta) = \min\left\{\tilde{K}\left(\phi_i(\mathcal{H})+\sqrt{\mathrm{diam}(\mathcal{H})\frac{\ln(32i^2/\delta)}{2^{i-1}}}+\frac{\ln(32i^2/\delta)}{2^{i-1}}\right),1\right\},$$

where for our purposes we can take $\tilde{K}=8272$. It is known (see, e.g., Massart and Nédélec, 2006; Giné and Koltchinskii, 2006) that for some universal constant $c' \in [2,\infty)$,

$$\phi_{i+1}(\mathcal{H}) \le c'\max\left\{\sqrt{\mathrm{diam}(\mathcal{H})2^{-i}d\log_2\frac{2}{\mathrm{diam}(\mathcal{H})}}, 2^{-i}di\right\}. \tag{85}$$

We also generally have $\phi_i(\mathcal{H}) \le 2$ for every $i \in \mathbb{N}$. The next lemma is taken from the work of Koltchinskii (2006) on data-dependent Rademacher complexity bounds on the excess risk.

Lemma 57 *For any $\delta \in (0,e^{-3})$, any $\mathcal{H} \subseteq \mathbb{C}$ with $f \in \mathrm{cl}(\mathcal{H})$, and any $i \in \mathbb{N}$, on an event K_i with $\mathbb{P}(K_i) \ge 1-\delta/4i^2$, $\forall h \in \mathcal{H}$,*

$$\begin{aligned}\mathrm{er}_{\mathcal{L}_i^\star}(h) - \min_{h'\in\mathcal{H}}\mathrm{er}_{\mathcal{L}_i^\star}(h') &\le \mathrm{er}(h)-\mathrm{er}(f)+\hat{U}_i(\mathcal{H},\delta)\\ \mathrm{er}(h)-\mathrm{er}(f) &\le \mathrm{er}_{\mathcal{L}_i^\star}(h)-\mathrm{er}_{\mathcal{L}_i^\star}(f)+\hat{U}_i(\mathcal{H},\delta)\\ \min\left\{\hat{U}_i(\mathcal{H},\delta),1\right\} &\le \tilde{U}_i(\mathcal{H},\delta).\end{aligned}$$

Lemma 57 essentially follows from a version of Talagrand's inequality. The details of the proof may be extracted from the proofs of Koltchinskii (2006), and related derivations have previously been presented by Hanneke (2011) and Koltchinskii (2010). The only minor twist here is that f need only be in $\mathrm{cl}(\mathcal{H})$, rather than in $\mathcal{H}$ itself, which easily follows from Koltchinskii's original results, since the Borel-Cantelli lemma implies that with probability one, every $\varepsilon > 0$ has some $g \in \mathcal{H}(\varepsilon)$ (very close to f) with $\mathrm{er}_{\mathcal{L}_i^\star}(g) = \mathrm{er}_{\mathcal{L}_i^\star}(f)$.

For our purposes, the important implications of Lemma 57 are summarized by the following lemma.

Lemma 58 *For any $\delta \in (0,e^{-3})$ and any $n \in \mathbb{N}$, when running Algorithm 5 with label budget n and confidence parameter δ, on an event $J_n(\delta)$ with $\mathbb{P}(J_n(\delta)) \ge 1-\delta/2$, $\forall i \in \{0,1,\ldots,\hat{i}_{d+1}\}$, if $V_{2^i}^\star \subseteq \hat{V}_i$*

then $\forall h \in \hat{V}_i$,

$$\begin{aligned}
\mathrm{er}_{\mathcal{L}^\star_{i+1}}(h) - \min_{h' \in \hat{V}_i} \mathrm{er}_{\mathcal{L}^\star_{i+1}}(h') &\le \mathrm{er}(h) - \mathrm{er}(f) + \hat{U}_{i+1}(\hat{V}_i, \delta) \\
\mathrm{er}(h) - \mathrm{er}(f) &\le \mathrm{er}_{\mathcal{L}^\star_{i+1}}(h) - \mathrm{er}_{\mathcal{L}^\star_{i+1}}(f) + \hat{U}_{i+1}(\hat{V}_i, \delta) \\
\min\left\{\hat{U}_{i+1}(\hat{V}_i, \delta), 1\right\} &\le \tilde{U}_{i+1}(\hat{V}_i, \delta).
\end{aligned}$$

Proof For each i, consider applying Lemma 57 under the conditional distribution given $\hat{V}_i$. The set $\mathcal{L}^\star_{i+1}$ is independent from $\hat{V}_i$, as are the Rademacher variables in the definition of $\hat{R}_{i+1}(\hat{V}_i)$. Furthermore, by Lemma 35, on H', $f \in \mathrm{cl}\left(V^\star_{2^i}\right)$, so that the conditions of Lemma 57 hold. The law of total probability then implies the existence of an event J_i of probability $\mathbb{P}(J_i) \ge 1 - \delta/4(i+1)^2$, on which the claimed inequalities hold for that value of i if $i \le \hat{i}_{d+1}$. A union bound over values of i then implies the existence of an event $J_n(\delta) = \bigcap_i J_i$ with probability $\mathbb{P}(J_n(\delta)) \ge 1 - \sum_i \delta/4(i+1)^2 \ge 1 - \delta/2$ on which the claimed inequalities hold for all $i \le \hat{i}_{d+1}$. ■

Lemma 59 *For some* $(\mathbb{C}, \mathcal{P}_{XY}, \gamma)$*-dependent constants* $c, c^* \in [1, \infty)$*, for any* $\delta \in (0, e^{-3})$ *and integer* $n \ge c^* \ln(1/\delta)$*, when running Algorithm 5 with label budget* n *and confidence parameter* δ*, on event* $J_n(\delta) \cap H_n^{(i)} \cap H_n^{(ii)}$*, every* $i \in \{0, 1, \ldots, \hat{i}_{\tilde{d}_f}\}$ *satisfies*

$$V^\star_{2^i} \subseteq \hat{V}_i \subseteq \mathbb{C}\left(c\left(\frac{di + \ln(1/\delta)}{2^i}\right)^{\frac{\kappa}{2\kappa-1}}\right),$$

and furthermore $V^\star_{\hat{m}^{(\tilde{d}_f)}} \subseteq \hat{V}^{(\tilde{d}_f)}$.

Proof Define $c = \left(24\tilde{K}c'\sqrt{\mu}\right)^{\frac{2\kappa}{2\kappa-1}}$, $c^* = \max\left\{\tau^*, 8d\left(\frac{\mu c^{1/\kappa}}{r_{(1-\gamma)/6}}\right)^{\frac{1}{2\kappa-1}} \log_2\left(\frac{4\mu c^{1/\kappa}}{r_{(1-\gamma)/6}}\right)\right\}$, and suppose $n \ge c^* \ln(1/\delta)$. We now proceed by induction. As the right side equals $\mathbb{C}$ for $i = 0$, the claimed inclusions are certainly true for $\hat{V}_0 = \mathbb{C}$, which serves as our base case. Now suppose some $i \in \{0, 1, \ldots, \hat{i}_{\tilde{d}_f}\}$ satisfies

$$V^\star_{2^i} \subseteq \hat{V}_i \subseteq \mathbb{C}\left(c\left(\frac{di + \ln(1/\delta)}{2^i}\right)^{\frac{\kappa}{2\kappa-1}}\right). \tag{86}$$

In particular, Condition 1 implies

$$\mathrm{diam}(\hat{V}_i) \le \mathrm{diam}\left(\mathbb{C}\left(c\left(\frac{di + \ln(1/\delta)}{2^i}\right)^{\frac{\kappa}{2\kappa-1}}\right)\right) \le \mu c^{\frac{1}{\kappa}}\left(\frac{di + \ln(1/\delta)}{2^i}\right)^{\frac{1}{2\kappa-1}}. \tag{87}$$

If $i < \hat{i}_{\tilde{d}_f}$, then let k be the integer for which $\hat{i}_{k-1} \le i < \hat{i}_k$, and otherwise let $k = \tilde{d}_f$. Note that we certainly have $\hat{i}_1 \ge \lfloor \log_2(n/2) \rfloor$, since $m = \lfloor n/2 \rfloor \ge 2^{\lfloor \log_2(n/2) \rfloor}$ is obtained while $k = 1$. Therefore, if $k > 1$,

$$\frac{di + \ln(1/\delta)}{2^i} \le \frac{4d\log_2(n) + 4\ln(1/\delta)}{n},$$

so that (87) implies

$$\operatorname{diam}\left(\hat{V}_i\right) \leq \mu c^{\frac{1}{\kappa}} \left(\frac{4d\log_2(n)+4\ln(1/\delta)}{n}\right)^{\frac{1}{2\kappa-1}}.$$

By our choice of c^*, the right side is at most $r_{(1-\gamma)/6}$. Therefore, since Lemma 35 implies $f \in \operatorname{cl}\left(V^\star_{2^i}\right)$ on $H_n^{(i)}$, we have $\hat{V}_i \subseteq \mathrm{B}\left(f, r_{(1-\gamma)/6}\right)$ when $k > 1$. Combined with (86), we have that $V^\star_{2^i} \subseteq \hat{V}_i$, and either $k = 1$, or $\hat{V}_i \subseteq \mathrm{B}(f, r_{(1-\gamma)/6})$ and $4m > 4\lfloor n/2 \rfloor \geq n$. Now consider any m with $2^i + 1 \leq m \leq \min\left\{2^{i+1}, \hat{m}^{(\tilde{d}_f)}\right\}$, and for the purpose of induction suppose $V^\star_{m-1} \subseteq V_{i+1}$ upon reaching Step 5 for that value of m in Algorithm 5. Since $V_{i+1} \subseteq \hat{V}_i$ and $n \geq \tau^*$, Lemma 41 (with $\ell = m-1$) implies that on $H_n^{(i)} \cap H_n^{(ii)}$,

$$\hat{\Delta}^{(k)}_{4m}(X_m, W_2, V_{i+1}) < \gamma \implies \hat{\Gamma}^{(k)}_{4m}(X_m, -f(X_m), W_2, V_{i+1}) < \hat{\Gamma}^{(k)}_{4m}(X_m, f(X_m), W_2, V_{i+1}),$$

so that after Step 8 we have $V^\star_m \subseteq V_{i+1}$. Since (86) implies that the $V^\star_{m-1} \subseteq V_{i+1}$ condition holds if Algorithm 5 reaches Step 5 with $m = 2^i + 1$ (at which time $V_{i+1} = \hat{V}_i$), we have by induction that on $H_n^{(i)} \cap H_n^{(ii)}$, $V^\star_m \subseteq V_{i+1}$ upon reaching Step 9 with $m = \min\left\{2^{i+1}, \hat{m}^{(\tilde{d}_f)}\right\}$. This establishes the final claim of the lemma, given that the first claim holds. For the remainder of this inductive proof, suppose $i < \hat{i}_{\tilde{d}_f}$. Since Step 8 enforces that, upon reaching Step 9 with $m = 2^{i+1}$, every $h_1, h_2 \in V_{i+1}$ have $\operatorname{er}_{\hat{\mathcal{L}}_{i+1}}(h_1) - \operatorname{er}_{\hat{\mathcal{L}}_{i+1}}(h_2) = \operatorname{er}_{\mathcal{L}^\star_{i+1}}(h_1) - \operatorname{er}_{\mathcal{L}^\star_{i+1}}(h_2)$, on $J_n(\delta) \cap H_n^{(i)} \cap H_n^{(ii)}$ we have

$$\begin{aligned}
\hat{V}_{i+1} &\subseteq \left\{h \in \hat{V}_i : \operatorname{er}_{\mathcal{L}^\star_{i+1}}(h) - \min_{h' \in V^\star_{2^{i+1}}} \operatorname{er}_{\mathcal{L}^\star_{i+1}}(h') \leq \hat{U}_{i+1}\left(\hat{V}_i, \delta\right)\right\} \\
&\subseteq \left\{h \in \hat{V}_i : \operatorname{er}_{\mathcal{L}^\star_{i+1}}(h) - \operatorname{er}_{\mathcal{L}^\star_{i+1}}(f) \leq \hat{U}_{i+1}\left(\hat{V}_i, \delta\right)\right\} \\
&\subseteq \hat{V}_i \cap \mathbb{C}\left(2\hat{U}_{i+1}\left(\hat{V}_i, \delta\right)\right) \subseteq \mathbb{C}\left(2\tilde{U}_{i+1}\left(\hat{V}_i, \delta\right)\right), \qquad (88)
\end{aligned}$$

where the second line follows from Lemma 35 and the last two inclusions follow from Lemma 58. Focusing on (88), combining (87) with (85) (and the fact that $\phi_{i+1}(\hat{V}_i) \leq 2$), we can bound the value of $\tilde{U}_{i+1}\left(\hat{V}_i, \delta\right)$ as follows.

$$\begin{aligned}
\sqrt{\operatorname{diam}(\hat{V}_i)\frac{\ln(32(i+1)^2/\delta)}{2^i}} &\leq \sqrt{\mu} c^{\frac{1}{2\kappa}} \left(\frac{di+\ln(1/\delta)}{2^i}\right)^{\frac{1}{4\kappa-2}} \left(\frac{\ln(32(i+1)^2/\delta)}{2^i}\right)^{\frac{1}{2}} \\
&\leq \sqrt{\mu} c^{\frac{1}{2\kappa}} \left(\frac{2di+2\ln(1/\delta)}{2^{i+1}}\right)^{\frac{1}{4\kappa-2}} \left(\frac{8(i+1)+2\ln(1/\delta)}{2^{i+1}}\right)^{\frac{1}{2}} \\
&\leq 4\sqrt{\mu} c^{\frac{1}{2\kappa}} \left(\frac{d(i+1)+\ln(1/\delta)}{2^{i+1}}\right)^{\frac{\kappa}{2\kappa-1}},
\end{aligned}$$

$$\begin{aligned}
\phi_{i+1}(\hat{V}_i) &\leq c'\sqrt{\mu} c^{\frac{1}{2\kappa}} \left(\frac{di+\ln(1/\delta)}{2^i}\right)^{\frac{1}{4\kappa-2}} \left(\frac{d(i+2)}{2^i}\right)^{\frac{1}{2}} \\
&\leq 4c'\sqrt{\mu} c^{\frac{1}{2\kappa}} \left(\frac{d(i+1)+\ln(1/\delta)}{2^{i+1}}\right)^{\frac{\kappa}{2\kappa-1}},
\end{aligned}$$

and thus

$$\tilde{U}_{i+1}(\hat{V}_i,\delta) \leq \min\left\{8\tilde{K}c'\sqrt{\mu}c^{\frac{1}{2\kappa}}\left(\frac{d(i+1)+\ln(1/\delta)}{2^{i+1}}\right)^{\frac{\kappa}{2\kappa-1}} + \tilde{K}\frac{\ln(32(i+1)^2/\delta)}{2^i}, 1\right\}$$
$$\leq 12\tilde{K}c'\sqrt{\mu}c^{\frac{1}{2\kappa}}\left(\frac{d(i+1)+\ln(1/\delta)}{2^{i+1}}\right)^{\frac{\kappa}{2\kappa-1}} = (c/2)\left(\frac{d(i+1)+\ln(1/\delta)}{2^{i+1}}\right)^{\frac{\kappa}{2\kappa-1}}.$$

Combining this with (88) now implies

$$\hat{V}_{i+1} \subseteq \mathbb{C}\left(c\left(\frac{d(i+1)+\ln(1/\delta)}{2^{i+1}}\right)^{\frac{\kappa}{2\kappa-1}}\right).$$

To complete the inductive proof, it remains only to show $V^\star_{2^{i+1}} \subseteq \hat{V}_{i+1}$. Toward this end, recall we have shown above that on $H_n^{(i)} \cap H_n^{(ii)}$, $V^\star_{2^{i+1}} \subseteq V_{i+1}$ upon reaching Step 9 with $m = 2^{i+1}$, and that every $h_1, h_2 \in V_{i+1}$ at this point have $\mathrm{er}_{\hat{\mathcal{L}}_{i+1}}(h_1) - \mathrm{er}_{\hat{\mathcal{L}}_{i+1}}(h_2) = \mathrm{er}_{\mathcal{L}^\star_{i+1}}(h_1) - \mathrm{er}_{\mathcal{L}^\star_{i+1}}(h_2)$. Consider any $h \in V^\star_{2^{i+1}}$, and note that any other $g \in V^\star_{2^{i+1}}$ has $\mathrm{er}_{\mathcal{L}^\star_{i+1}}(g) = \mathrm{er}_{\mathcal{L}^\star_{i+1}}(h)$. Thus, on $H_n^{(i)} \cap H_n^{(ii)}$,

$$\mathrm{er}_{\hat{\mathcal{L}}_{i+1}}(h) - \min_{h' \in V_{i+1}} \mathrm{er}_{\hat{\mathcal{L}}_{i+1}}(h') = \mathrm{er}_{\mathcal{L}^\star_{i+1}}(h) - \min_{h' \in V_{i+1}} \mathrm{er}_{\mathcal{L}^\star_{i+1}}(h')$$
$$\leq \mathrm{er}_{\mathcal{L}^\star_{i+1}}(h) - \min_{h' \in \hat{V}_i} \mathrm{er}_{\mathcal{L}^\star_{i+1}}(h') = \inf_{g \in V^\star_{2^{i+1}}} \mathrm{er}_{\mathcal{L}^\star_{i+1}}(g) - \min_{h' \in \hat{V}_i} \mathrm{er}_{\mathcal{L}^\star_{i+1}}(h'). \quad (89)$$

Lemma 58 and (86) imply that on $J_n(\delta) \cap H_n^{(i)} \cap H_n^{(ii)}$, the last expression in (89) is not larger than $\inf_{g \in V^\star_{2^{i+1}}} \mathrm{er}(g) - \mathrm{er}(f) + \hat{U}_{i+1}(\hat{V}_i, \delta)$, and Lemma 35 implies $f \in \mathrm{cl}\left(V^\star_{2^{i+1}}\right)$ on $H_n^{(i)}$, so that $\inf_{g \in V^\star_{2^{i+1}}} \mathrm{er}(g) = \mathrm{er}(f)$. We therefore have

$$\mathrm{er}_{\hat{\mathcal{L}}_{i+1}}(h) - \min_{h' \in V_{i+1}} \mathrm{er}_{\hat{\mathcal{L}}_{i+1}}(h') \leq \hat{U}_{i+1}(\hat{V}_i, \delta),$$

so that $h \in \hat{V}_{i+1}$ as well. Since this holds for any $h \in V^\star_{2^{i+1}}$, we have $V^\star_{2^{i+1}} \subseteq \hat{V}_{i+1}$. The lemma now follows by the principle of induction. ■

Lemma 60 *There exist $(\mathbb{C}, \mathcal{P}_{XY}, \gamma)$-dependent constants $c_1^*, c_2^* \in [1,\infty)$ such that, for any $\varepsilon, \delta \in (0, e^{-3})$ and integer*

$$n \geq c_1^* + c_2^* \tilde{\theta}_f\left(\varepsilon^{\frac{1}{\kappa}}\right)\varepsilon^{\frac{2}{\kappa}-2}\log_2^2\left(\frac{1}{\varepsilon\delta}\right),$$

when running Algorithm 5 with label budget n and confidence parameter δ, on an event $J_n^*(\varepsilon,\delta)$ with $\mathbb{P}(J_n^*(\varepsilon,\delta)) \geq 1-\delta$, we have $\hat{V}_{\hat{i}_{\tilde{d}_f}} \subseteq \mathbb{C}(\varepsilon)$.

Proof Define

$$c_1^* = \max\left\{2^{\tilde{d}_f+5}\left(\frac{\mu c^{1/\kappa}}{r_{(1-\gamma)/6}}\right)^{2\kappa-1} d\log_2\frac{d\mu c^{1/\kappa}}{r_{(1-\gamma)/6}}, \frac{2}{\tilde{\delta}_f^{1/3}}\ln\left(8c^{(i)}\right), \frac{120}{\tilde{\delta}_f^{1/3}}\ln\left(8c^{(ii)}\right)\right\}$$

and

$$c_2^* = \max\left\{c^*, 2^{\tilde{d}_f+5}\cdot\left(\frac{\mu c^{1/\kappa}}{r_{(1-\gamma)/6}}\right)^{2\kappa-1}, 2^{\tilde{d}_f+15}\cdot\frac{\mu c^2 d}{\gamma\tilde{\delta}_f}\log_2^2(4dc)\right\}.$$

Fix any $\varepsilon,\delta\in(0,e^{-3})$ and integer $n\geq c_1^*+c_2^*\tilde{\theta}_f\left(\varepsilon^{\frac{1}{\kappa}}\right)\varepsilon^{\frac{2}{\kappa}-2}\log_2^2\left(\frac{1}{\varepsilon\delta}\right)$.

For each $i\in\{0,1,\ldots\}$, let $\tilde{r}_i=\mu c^{\frac{1}{\kappa}}\left(\frac{di+\ln(1/\delta)}{2^i}\right)^{\frac{1}{2\kappa-1}}$. Also define

$$\tilde{i}=\left\lceil\left(2-\frac{1}{\kappa}\right)\log_2\frac{c}{\varepsilon}+\log_2\left[8d\log_2\frac{2dc}{\varepsilon\delta}\right]\right\rceil.$$

and let $\check{i}=\min\left\{i\in\mathbb{N}:\sup_{j\geq i}\tilde{r}_j<r_{(1-\gamma)/6}\right\}$. For any $i\in\left\{\check{i},\ldots,\hat{i}_{\tilde{d}_f}\right\}$, let

$$\mathcal{Q}_{i+1}=\left\{m\in\left\{2^i+1,\ldots,2^{i+1}\right\}:\hat{\Delta}_{4m}^{(\tilde{d}_f)}\left(X_m,W_2,\mathrm{B}\left(f,\tilde{r}_i\right)\right)\geq 2\gamma/3\right\}.$$

Also define

$$\tilde{\mathcal{Q}}=\frac{96}{\gamma\tilde{\delta}_f}\tilde{\theta}_f\left(\varepsilon^{\frac{1}{\kappa}}\right)\cdot 2\mu c^2\cdot\left(8d\log_2\frac{2dc}{\varepsilon\delta}\right)\cdot\varepsilon^{\frac{2}{\kappa}-2}.$$

By Lemma 59 and Condition 1, on $J_n(\delta)\cap H_n^{(i)}\cap H_n^{(ii)}$, if $i\leq\hat{i}_{\tilde{d}_f}$,

$$\hat{V}_i\subseteq\mathbb{C}\left(c\left(\frac{di+\ln(1/\delta)}{2^i}\right)^{\frac{\kappa}{2\kappa-1}}\right)\subseteq\mathrm{B}\left(f,\tilde{r}_i\right). \tag{90}$$

Lemma 59 also implies that, on $J_n(\delta)\cap H_n^{(i)}\cap H_n^{(ii)}$, for i with $\hat{i}_{\tilde{d}_f-1}\leq i\leq\hat{i}_{\tilde{d}_f}$, all of the sets V_{i+1} obtained in Algorithm 5 while $k=\tilde{d}_f$ and $m\in\left\{2^i+1,\ldots,2^{i+1}\right\}$ satisfy $V_{2^{i+1}}^\star\subseteq V_{i+1}\subseteq\hat{V}_i$. Recall that $\hat{i}_1\geq\lfloor\log_2(n/2)\rfloor$, so that we have either $\tilde{d}_f=1$ or else every $m\in\left\{2^i+1,\ldots,2^{i+1}\right\}$ has $4m>n$. Also recall that Lemma 49 implies that when the above conditions are satisfied, and $i\geq\check{i}$, on $H'\cap G_n^{(i)}$, $\hat{\Delta}_{4m}^{(\tilde{d}_f)}\left(X_m,W_2,V_{i+1}\right)\leq(3/2)\hat{\Delta}_{4m}^{(\tilde{d}_f)}\left(X_m,W_2,\mathrm{B}\left(f,\tilde{r}_i\right)\right)$, so that $|\mathcal{Q}_{i+1}|$ upper bounds the number of $m\in\left\{2^i+1,\ldots,2^{i+1}\right\}$ for which Algorithm 5 requests the label Y_m in Step 6 of the $k=\tilde{d}_f$ round. Thus, on $J_n(\delta)\cap H_n^{(i)}\cap H_n^{(ii)}$, $2^{\check{i}}+\sum_{i=\max\left\{\check{i},\hat{i}_{\tilde{d}_f-1}\right\}}^{\hat{i}_{\tilde{d}_f}}|\mathcal{Q}_{i+1}|$ upper bounds the total number of label requests by Algorithm 5 while $k=\tilde{d}_f$; therefore, by the constraint in Step 3, we know that either this quantity is at least as big as $\left\lfloor 2^{-\tilde{d}_f}n\right\rfloor$, or else we have $2^{\hat{i}_{\tilde{d}_f}+1}>\tilde{d}_f\cdot 2^n$. In particular, on this event, if we can show that

$$2^{\check{i}}+\sum_{i=\max\left\{\check{i},\hat{i}_{\tilde{d}_f-1}\right\}}^{\min\left\{\hat{i}_{\tilde{d}_f},\tilde{i}\right\}}|\mathcal{Q}_{i+1}|<\left\lfloor 2^{-\tilde{d}_f}n\right\rfloor\text{ and }2^{\tilde{i}+1}\leq\tilde{d}_f\cdot 2^n, \tag{91}$$

then it must be true that $\tilde{i}<\hat{i}_{\tilde{d}_f}$. Next, we will focus on establishing this fact.

Consider any $i\in\left\{\max\left\{\check{i},\hat{i}_{\tilde{d}_f-1}\right\},\ldots,\min\left\{\hat{i}_{\tilde{d}_f},\tilde{i}\right\}\right\}$ and any $m\in\left\{2^i+1,\ldots,2^{i+1}\right\}$. If $\tilde{d}_f=1$, then

$$\mathbb{P}\left(\hat{\Delta}_{4m}^{(\tilde{d}_f)}\left(X_m,W_2,\mathrm{B}\left(f,\tilde{r}_i\right)\right)\geq 2\gamma/3\middle|W_2\right)=\mathcal{P}^{\tilde{d}_f}\left(\mathcal{S}^{\tilde{d}_f}\left(\mathrm{B}\left(f,\tilde{r}_i\right)\right)\right).$$

Otherwise, if $\tilde{d}_f > 1$, then by Markov's inequality and the definition of $\hat{\Delta}_{4m}^{(\tilde{d}_f)}(\cdot,\cdot,\cdot)$ from (15),

$$\mathbb{P}\left(\hat{\Delta}_{4m}^{(\tilde{d}_f)}(X_m, W_2, \mathrm{B}(f,\tilde{r}_i)) \geq 2\gamma/3 \middle| W_2\right) \leq \frac{3}{2\gamma}\mathbb{E}\left[\hat{\Delta}_{4m}^{(\tilde{d}_f)}(X_m, W_2, \mathrm{B}(f,\tilde{r}_i)) \middle| W_2\right]$$
$$= \frac{3}{2\gamma}\frac{1}{M_{4m}^{(\tilde{d}_f)}(\mathrm{B}(f,\tilde{r}_i))}\sum_{s=1}^{(4m)^3}\mathbb{P}\left(S_s^{(\tilde{d}_f)} \cup \{X_m\} \in \mathcal{S}^{\tilde{d}_f}(\mathrm{B}(f,\tilde{r}_i)) \middle| S_s^{(\tilde{d}_f)}\right).$$

By Lemma 39, Lemma 59, and (90), on $J_n(\delta) \cap H_n^{(i)} \cap H_n^{(ii)}$, this is at most

$$\frac{3}{\tilde{\delta}_f\gamma}\frac{1}{(4m)^3}\sum_{s=1}^{(4m)^3}\mathbb{P}\left(S_s^{(\tilde{d}_f)} \cup \{X_m\} \in \mathcal{S}^{\tilde{d}_f}(\mathrm{B}(f,\tilde{r}_i)) \middle| S_s^{(\tilde{d}_f)}\right)$$
$$\leq \frac{24}{\tilde{\delta}_f\gamma}\frac{1}{4^3 2^{3i+3}}\sum_{s=1}^{4^3 2^{3i+3}}\mathbb{P}\left(S_s^{(\tilde{d}_f)} \cup \{X_m\} \in \mathcal{S}^{\tilde{d}_f}(\mathrm{B}(f,\tilde{r}_i)) \middle| S_s^{(\tilde{d}_f)}\right).$$

Note that this value is invariant to the choice of $m \in \left\{2^i+1,\ldots,2^{i+1}\right\}$. By Hoeffding's inequality, on an event $J_n^*(i)$ of probability $\mathbb{P}(J_n^*(i)) \geq 1-\delta/(16i^2)$, this is at most

$$\frac{24}{\tilde{\delta}_f\gamma}\left(\sqrt{\frac{\ln(4i/\delta)}{4^3 2^{3i+3}}} + \mathcal{P}^{\tilde{d}_f}\left(\mathcal{S}^{\tilde{d}_f}(\mathrm{B}(f,\tilde{r}_i))\right)\right). \tag{92}$$

Since $i \geq \hat{i}_1 > \log_2(n/4)$ and $n \geq \ln(1/\delta)$, we have

$$\sqrt{\frac{\ln(4i/\delta)}{4^3 2^{3i+3}}} \leq 2^{-i}\sqrt{\frac{\ln(4\log_2(n/4)/\delta)}{128n}} \leq 2^{-i}\sqrt{\frac{\ln(n/\delta)}{128n}} \leq 2^{-i}.$$

Thus, (92) is at most

$$\frac{24}{\tilde{\delta}_f\gamma}\left(2^{-i} + \mathcal{P}^{\tilde{d}_f}\left(\mathcal{S}^{\tilde{d}_f}(\mathrm{B}(f,\tilde{r}_i))\right)\right).$$

In either case ($\tilde{d}_f = 1$ or $\tilde{d}_f > 1$), by definition of $\tilde{\theta}_f\left(\varepsilon^{\frac{1}{\kappa}}\right)$, on $J_n(\delta) \cap H_n^{(i)} \cap H_n^{(ii)} \cap J_n^*(i)$, $\forall m \in \left\{2^i+1,\ldots,2^{i+1}\right\}$ we have

$$\mathbb{P}\left(\hat{\Delta}_{4m}^{(\tilde{d}_f)}(X_m, W_2, \mathrm{B}(f,\tilde{r}_i)) \geq 2\gamma/3 \middle| W_2\right) \leq \frac{24}{\tilde{\delta}_f\gamma}\left(2^{-i} + \tilde{\theta}_f\left(\varepsilon^{\frac{1}{\kappa}}\right)\cdot\max\left\{\tilde{r}_i, \varepsilon^{\frac{1}{\kappa}}\right\}\right). \tag{93}$$

Furthermore, the $\mathbb{1}_{[2\gamma/3,\infty)}\left(\hat{\Delta}_{4m}^{(\tilde{d}_f)}(X_m, W_2, \mathrm{B}(f,\tilde{r}_i))\right)$ indicators are conditionally independent given W_2, so that we may bound $\mathbb{P}\left(|Q_{i+1}| > \tilde{Q} \middle| W_2\right)$ via a Chernoff bound. Toward this end, note that on $J_n(\delta) \cap H_n^{(i)} \cap H_n^{(ii)} \cap J_n^*(i)$, (93) implies

$$\mathbb{E}\left[|Q_{i+1}| \middle| W_2\right] = \sum_{m=2^i+1}^{2^{i+1}}\mathbb{P}\left(\hat{\Delta}_{4m}^{(\tilde{d}_f)}(X_m, W_2, \mathrm{B}(f,\tilde{r}_i)) \geq 2\gamma/3 \middle| W_2\right)$$
$$\leq 2^i \cdot \frac{24}{\tilde{\delta}_f\gamma}\left(2^{-i} + \tilde{\theta}_f\left(\varepsilon^{\frac{1}{\kappa}}\right)\cdot\max\left\{\tilde{r}_i, \varepsilon^{\frac{1}{\kappa}}\right\}\right) \leq \frac{24}{\tilde{\delta}_f\gamma}\left(1 + \tilde{\theta}_f\left(\varepsilon^{\frac{1}{\kappa}}\right)\cdot\max\left\{2^i\tilde{r}_i, 2^{\tilde{i}}\varepsilon^{\frac{1}{\kappa}}\right\}\right). \tag{94}$$

Note that

$$\begin{aligned}2^i\tilde{r}_i &= \mu c^{\frac{1}{\kappa}}\left(di+\ln(1/\delta)\right)^{\frac{1}{2\kappa-1}}\cdot 2^{i\left(1-\frac{1}{2\kappa-1}\right)}\\ &\leq \mu c^{\frac{1}{\kappa}}\left(d\tilde{i}+\ln(1/\delta)\right)^{\frac{1}{2\kappa-1}}\cdot 2^{\tilde{i}\left(1-\frac{1}{2\kappa-1}\right)} \leq \mu c^{\frac{1}{\kappa}}\left(8d\log_2\frac{2dc}{\varepsilon\delta}\right)^{\frac{1}{2\kappa-1}}\cdot 2^{\tilde{i}\left(1-\frac{1}{2\kappa-1}\right)}.\end{aligned}$$

Then since $2^{-\tilde{i}\frac{1}{2\kappa-1}} \leq \left(\frac{\varepsilon}{c}\right)^{\frac{1}{\kappa}}\cdot\left(8d\log_2\frac{2dc}{\varepsilon\delta}\right)^{-\frac{1}{2\kappa-1}}$, we have that the rightmost expression in (94) is at most

$$\frac{24}{\gamma\tilde{\delta}_f}\left(1+\tilde{\theta}_f\left(\varepsilon^{\frac{1}{\kappa}}\right)\cdot\mu\cdot 2^{\tilde{i}}\varepsilon^{\frac{1}{\kappa}}\right) \leq \frac{24}{\gamma\tilde{\delta}_f}\left(1+\tilde{\theta}_f\left(\varepsilon^{\frac{1}{\kappa}}\right)\cdot 2\mu c^2\cdot\left(8d\log_2\frac{2dc}{\varepsilon\delta}\right)\cdot\varepsilon^{\frac{2}{\kappa}-2}\right) \leq \tilde{\mathcal{Q}}/2.$$

Therefore, a Chernoff bound implies that on $J_n(\delta)\cap H_n^{(i)}\cap H_n^{(ii)}\cap J_n^*(i)$, we have

$$\begin{aligned}\mathbb{P}\left(|\mathcal{Q}_{i+1}| > \tilde{\mathcal{Q}}\middle| W_2\right) &\leq \exp\left\{-\tilde{\mathcal{Q}}/6\right\} \leq \exp\left\{-8\log_2\left(\frac{2dc}{\varepsilon\delta}\right)\right\}\\ &\leq \exp\left\{-\log_2\left(\frac{48\log_2(2dc/\varepsilon\delta)}{\delta}\right)\right\} \leq \delta/(8\tilde{i}).\end{aligned}$$

Combined with the law of total probability and a union bound over i values, this implies there exists an event $J_n^*(\varepsilon,\delta)\subseteq J_n(\delta)\cap H_n^{(i)}\cap H_n^{(ii)}$ with

$$\mathbb{P}\left(J_n(\delta)\cap H_n^{(i)}\cap H_n^{(ii)}\setminus J_n^*(\varepsilon,\delta)\right) \leq \sum_{i=\check{i}}^{\tilde{i}}\left(\delta/(16i^2)+\delta/(8\tilde{i})\right) \leq \delta/4,$$

on which every $i\in\left\{\max\left\{\check{i},\hat{i}_{\tilde{d}_f-1}\right\},\ldots,\min\left\{\hat{i}_{\tilde{d}_f},\tilde{i}\right\}\right\}$ has $|\mathcal{Q}_{i+1}|\leq\tilde{\mathcal{Q}}$.

We have chosen c_1^* and c_2^* large enough that $2^{\tilde{i}+1}<\tilde{d}_f\cdot 2^n$ and $2^{\check{i}}<2^{-\tilde{d}_f-2}n$. In particular, this means that on $J_n^*(\varepsilon,\delta)$,

$$2^{\check{i}}+\sum_{i=\max\left\{\check{i},\hat{i}_{\tilde{d}_f-1}\right\}}^{\min\left\{\tilde{i},\hat{i}_{\tilde{d}_f}\right\}}|\mathcal{Q}_{i+1}| < 2^{-\tilde{d}_f-2}n+\tilde{i}\tilde{\mathcal{Q}}.$$

Furthermore, since $\tilde{i}\leq 3\log_2\frac{4dc}{\varepsilon\delta}$, we have

$$\begin{aligned}\tilde{i}\tilde{\mathcal{Q}} &\leq \frac{2^{13}\mu c^2 d}{\gamma\tilde{\delta}_f}\tilde{\theta}_f\left(\varepsilon^{\frac{1}{\kappa}}\right)\cdot\varepsilon^{\frac{2}{\kappa}-2}\cdot\log_2^2\frac{4dc}{\varepsilon\delta}\\ &\leq \frac{2^{13}\mu c^2 d\log_2^2(4dc)}{\gamma\tilde{\delta}_f}\tilde{\theta}_f\left(\varepsilon^{\frac{1}{\kappa}}\right)\cdot\varepsilon^{\frac{2}{\kappa}-2}\cdot\log_2^2\frac{1}{\varepsilon\delta} \leq 2^{-\tilde{d}_f-2}n.\end{aligned}$$

Combining the above, we have that (91) is satisfied on $J_n^*(\varepsilon,\delta)$, so that $\hat{i}_{\tilde{d}_f}>\tilde{i}$. Combined with Lemma 59, this implies that on $J_n^*(\varepsilon,\delta)$,

$$\hat{V}_{\hat{i}_{\tilde{d}_f}}\subseteq\hat{V}_{\tilde{i}}\subseteq\mathbb{C}\left(c\left(\frac{d\tilde{i}+\ln(1/\delta)}{2^{\tilde{i}}}\right)^{\frac{\kappa}{2\kappa-1}}\right),$$

and by definition of $\tilde{i}$ we have

$$c\left(\frac{d\tilde{i}+\ln(1/\delta)}{2^{\tilde{i}}}\right)^{\frac{\kappa}{2\kappa-1}} \leq c\left(8d\log_2\frac{2dc}{\varepsilon\delta}\right)^{\frac{\kappa}{2\kappa-1}}\cdot 2^{-\tilde{i}\frac{\kappa}{2\kappa-1}}$$
$$\leq c\left(8d\log_2\frac{2dc}{\varepsilon\delta}\right)^{\frac{\kappa}{2\kappa-1}}\cdot(\varepsilon/c)\cdot\left(8d\log_2\frac{2dc}{\varepsilon\delta}\right)^{-\frac{\kappa}{2\kappa-1}}=\varepsilon,$$

so that $\hat{V}_{\hat{i}_{\tilde{d}_f}} \subseteq \mathbb{C}(\varepsilon)$.

Finally, to prove the stated bound on $\mathbb{P}(J_n^*(\varepsilon,\delta))$, by a union bound we have

$$1-\mathbb{P}(J_n^*(\varepsilon,\delta)) \leq (1-\mathbb{P}(J_n(\delta)))+\left(1-\mathbb{P}\left(H_n^{(i)}\right)\right)+\mathbb{P}\left(H_n^{(i)}\setminus H_n^{(ii)}\right)$$
$$+\mathbb{P}\left(J_n(\delta)\cap H_n^{(i)}\cap H_n^{(ii)}\setminus J_n^*(\varepsilon,\delta)\right)$$
$$\leq 3\delta/4+c^{(i)}\cdot\exp\left\{-n^3\tilde{\delta}_f/8\right\}+c^{(ii)}\cdot\exp\left\{-n\tilde{\delta}_f^{1/3}/120\right\}\leq\delta.$$

■

We are now ready for the proof of Lemma 26.

Proof [Lemma 26] First, note that because we break ties in the argmax of Step 7 in favor of a $\hat{y}$ value with $V_{i_k+1}[(X_m,\hat{y})]\neq\emptyset$, if $V_{i_k+1}\neq\emptyset$ before Step 8, then this remains true after Step 8. Furthermore, the $\hat{U}_{i_k+1}$ estimator is nonnegative, and thus the update in Step 10 never removes from V_{i_k+1} the minimizer of $\mathrm{er}_{\hat{\mathcal{L}}_{i_k+1}}(h)$ among $h\in V_{i_k+1}$. Therefore, by induction we have $V_{i_k}\neq\emptyset$ at all times in Algorithm 5. In particular, $\hat{V}_{\hat{i}_{d+1}+1}\neq\emptyset$ so that the return classifier $\hat{h}$ exists. Also, by Lemma 60, for n as in Lemma 60, on $J_n^*(\varepsilon,\delta)$, running Algorithm 5 with label budget n and confidence parameter δ results in $\hat{V}_{\hat{i}_{\tilde{d}_f}}\subseteq\mathbb{C}(\varepsilon)$. Combining these two facts implies that for such a value of n, on $J_n^*(\varepsilon,\delta)$, $\hat{h}\in\hat{V}_{\hat{i}_{d+1}+1}\subseteq\hat{V}_{\hat{i}_{\tilde{d}_f}}\subseteq\mathbb{C}(\varepsilon)$, so that $\mathrm{er}\left(\hat{h}\right)\leq\nu+\varepsilon$. ■

E.3 The Misspecified Model Case

Here we present a proof of Theorem 28, including a specification of the method $\mathcal{A}'_a$ from the theorem statement.

Proof [Theorem 28] Consider a weakly universally consistent passive learning algorithm $\mathcal{A}_u$ (Devroye, Györfi, and Lugosi, 1996). Such a method must exist in our setting; for instance, Hoeffding's inequality and a union bound imply that it suffices to take $\mathcal{A}_u(\mathcal{L})=\mathrm{argmin}_{\mathbb{1}^{\pm}_{B_i}}\mathrm{er}_{\mathcal{L}}(\mathbb{1}^{\pm}_{B_i})+\sqrt{\frac{\ln(4i^2|\mathcal{L}|)}{2|\mathcal{L}|}}$, where $\{B_1,B_2,\ldots\}$ is a countable algebra that generates $\mathcal{F}_{\mathcal{X}}$.

Then $\mathcal{A}_u$ achieves a label complexity Λ_u such that for any distribution $\mathcal{P}_{XY}$ on $\mathcal{X}\times\{-1,+1\}$, $\forall\varepsilon\in(0,1)$, $\Lambda_u(\varepsilon+\nu^*(\mathcal{P}_{XY}),\mathcal{P}_{XY})<\infty$. In particular, if $\nu^*(\mathcal{P}_{XY})<\nu(\mathbb{C};\mathcal{P}_{XY})$, then we have $\Lambda_u((\nu^*(\mathcal{P}_{XY})+\nu(\mathbb{C};\mathcal{P}_{XY}))/2,\mathcal{P}_{XY})<\infty$.

Fix any $n\in\mathbb{N}$ and describe the execution of $\mathcal{A}'_a(n)$ as follows. In a preprocessing step, withhold the first $m_{un}=n-\lfloor n/2\rfloor-\lfloor n/3\rfloor\geq n/6$ examples $\{X_1,\ldots,X_{m_{un}}\}$ and request their labels $\{Y_1,\ldots,Y_{m_{un}}\}$. Run $\mathcal{A}_a(\lfloor n/2\rfloor)$ on the remainder of the sequence $\{X_{m_{un}+1},X_{m_{un}+2},\ldots\}$ (i.e., shift

any index references in the algorithm by m_{un}), and let h_a denote the classifier it returns. Also request the labels $Y_{m_{un}+1}, \ldots Y_{m_{un}+\lfloor n/3 \rfloor}$, and let

$$h_u = \mathcal{A}_u\left(\left\{(X_{m_{un}+1}, Y_{m_{un}+1}), \ldots, (X_{m_{un}+\lfloor n/3 \rfloor}, Y_{m_{un}+\lfloor n/3 \rfloor})\right\}\right).$$

If $\mathrm{er}_{m_{un}}(h_a) - \mathrm{er}_{m_{un}}(h_u) > n^{-1/3}$, return $\hat{h} = h_u$; otherwise, return $\hat{h} = h_a$. This method achieves the stated result, for the following reasons.

First, let us examine the final step of this algorithm. By Hoeffding's inequality, with probability at least $1 - 2 \cdot \exp\left\{-n^{1/3}/12\right\}$,

$$|(\mathrm{er}_{m_{un}}(h_a) - \mathrm{er}_{m_{un}}(h_u)) - (\mathrm{er}(h_a) - \mathrm{er}(h_u))| \leq n^{-1/3}.$$

When this is the case, a triangle inequality implies $\mathrm{er}(\hat{h}) \leq \min\{\mathrm{er}(h_a), \mathrm{er}(h_u) + 2n^{-1/3}\}$.

If $\mathcal{P}_{XY}$ satisfies the benign noise case, then for any

$$n \geq 2\Lambda_a(\varepsilon/2 + \nu(\mathbb{C}; \mathcal{P}_{XY}), \mathcal{P}_{XY}),$$

we have $\mathbb{E}[\mathrm{er}(h_a)] \leq \nu(\mathbb{C}; \mathcal{P}_{XY}) + \varepsilon/2$, so $\mathbb{E}[\mathrm{er}(\hat{h})] \leq \nu(\mathbb{C}; \mathcal{P}_{XY}) + \varepsilon/2 + 2 \cdot \exp\{-n^{1/3}/12\}$, which is at most $\nu(\mathbb{C}; \mathcal{P}_{XY}) + \varepsilon$ if $n \geq 12^3 \ln^3(4/\varepsilon)$. So in this case, we can take $\lambda(\varepsilon) = \lceil 12^3 \ln^3(4/\varepsilon) \rceil$.

On the other hand, if $\mathcal{P}_{XY}$ is not in the benign noise case (i.e., the misspecified model case), then for any $n \geq 3\Lambda_u((\nu^*(\mathcal{P}_{XY}) + \nu(\mathbb{C}; \mathcal{P}_{XY}))/2, \mathcal{P}_{XY})$, $\mathbb{E}[\mathrm{er}(h_u)] \leq (\nu^*(\mathcal{P}_{XY}) + \nu(\mathbb{C}; \mathcal{P}_{XY}))/2$, so that

$$\begin{aligned}\mathbb{E}[\mathrm{er}(\hat{h})] &\leq \mathbb{E}[\mathrm{er}(h_u)] + 2n^{-1/3} + 2 \cdot \exp\{-n^{1/3}/12\} \\ &\leq (\nu^*(\mathcal{P}_{XY}) + \nu(\mathbb{C}; \mathcal{P}_{XY}))/2 + 2n^{-1/3} + 2 \cdot \exp\{-n^{1/3}/12\}.\end{aligned}$$

Again, this is at most $\nu(\mathbb{C}; \mathcal{P}_{XY}) + \varepsilon$ if $n \geq \max\left\{12^3 \ln^3 \frac{2}{\varepsilon}, 64(\nu(\mathbb{C}; \mathcal{P}_{XY}) - \nu^*(\mathcal{P}_{XY}))^{-3}\right\}$. So in this case, we can take

$$\lambda(\varepsilon) = \left\lceil \max\left\{12^3 \ln^3 \frac{2}{\varepsilon}, 3\Lambda_u\left(\frac{\nu^*(\mathcal{P}_{XY}) + \nu(\mathbb{C}; \mathcal{P}_{XY})}{2}, \mathcal{P}_{XY}\right), \frac{64}{(\nu(\mathbb{C}; \mathcal{P}_{XY}) - \nu^*(\mathcal{P}_{XY}))^3}\right\}\right\rceil.$$

In either case, we have $\lambda(\varepsilon) \in \mathrm{Polylog}(1/\varepsilon)$. ■

References

N. Abe and H. Mamitsuka. Query learning strategies using boosting and bagging. In *Proceedings of the* 15th *International Conference on Machine Learning*, 1998.

M. Alekhnovich, M. Braverman, V. Feldman, A. Klivans, and T. Pitassi. Learnability and automatizability. In *Proceedings of the* 45th *Foundations of Computer Science*, 2004.

K. Alexander. Probability inequalities for empirical processes and a law of the iterated logarithm. *The Annals of Probability*, 4:1041–1067, 1984.

M. Anthony and P. L. Bartlett. *Neural Network Learning: Theoretical Foundations*. Cambridge University Press, 1999.

A. Antos and G. Lugosi. Strong minimax lower bounds for learning. *Machine Learning*, 30:31–56, 1998.

R. B. Ash and C. A. Doléans-Dade. *Probability & Measure Theory*. Academic Press, 2000.

M.-F. Balcan, A. Beygelzimer, and J. Langford. Agnostic active learning. In *Proceedings of the* 23rd *International Conference on Machine Learning*, 2006a.

M.-F. Balcan, A. Blum, and S. Vempala. Kernels as features: On kernels, margins, and low-dimensional mappings. *Machine Learning Journal*, 65(1):79–94, 2006b.

M.-F. Balcan, A. Broder, and T. Zhang. Margin based active learning. In *Proceedings of the* 20th *Conference on Learning Theory*, 2007.

M.-F. Balcan, A. Beygelzimer, and J. Langford. Agnostic active learning. *Journal of Computer and System Sciences*, 75(1):78–89, 2009.

M.-F. Balcan, S. Hanneke, and J. Wortman Vaughan. The true sample complexity of active learning. *Machine Learning*, 80(2–3):111–139, 2010.

J. Baldridge and A. Palmer. How well does active learning *actually* work? Time-based evaluation of cost-reduction strategies for language documentation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, 2009.

Z. Bar-Yossef. Sampling lower bounds via information theory. In *Proceedings of the* 35th *Annual ACM Symposium on the Theory of Computing*, 2003.

P. L. Bartlett, M. I. Jordan, and J. D. McAuliffe. Convexity, classification, and risk bounds. *Journal of the American Statistical Association*, 101(473):138–156, 2006.

A. Beygelzimer, S. Dasgupta, and J. Langford. Importance weighted active learning. In *Proceedings of the International Conference on Machine Learning*, 2009.

A. Beygelzimer, D. Hsu, J. Langford, and T. Zhang. Agnostic active learning without constraints. In *Advances in Neural Information Processing Systems 23*, 2010.

A. Blumer, A. Ehrenfeucht, D. Haussler, and M. Warmuth. Learnability and the Vapnik-Chervonenkis dimension. *Journal of the Association for Computing Machinery*, 36(4):929–965, 1989.

F. Bunea, A. B. Tsybakov, and M. Wegkamp. Sparsity oracle inequalities for the lasso. *Electronic Journal of Statistics*, 1:169–194, 2009.

C. Campbell, N. Cristianini, and A. Smola. Query learning with large margin classifiers. In *Proceedings of the* 17th *International Conference on Machine Learning*, 2000.

R. Castro and R. D. Nowak. Minimax bounds for active learning. *IEEE Transactions on Information Theory*, 54(5):2339–2353, 2008.

D. Cohn, L. Atlas, and R. Ladner. Improving generalization with active learning. *Machine Learning*, 15(2):201–221, 1994.

S. Dasgupta. Coarse sample complexity bounds for active learning. In *Advances in Neural Information Processing Systems 18*, 2005.

S. Dasgupta, A. T. Kalai, and C. Monteleoni. Analysis of perceptron-based active learning. In *Proceedings of the* 18th *Conference on Learning Theory*, 2005.

S. Dasgupta, D. Hsu, and C. Monteleoni. A general agnostic active learning algorithm. In *Advances in Neural Information Processing Systems 20*, 2007.

S. Dasgupta, A. T. Kalai, and C. Monteleoni. Analysis of perceptron-based active learning. *Journal of Machine Learning Research*, 10:281–299, 2009.

O. Dekel, C. Gentile, and K. Sridharan. Robust selective sampling from single and multiple teachers. In *Proceedings of the* 23rd *Conference on Learning Theory*, 2010.

L. Devroye, L. Györfi, and G. Lugosi. *A Probabilistic Theory of Pattern Recognition*. Springer-Verlag New York, Inc., 1996.

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

Y. Freund, H. S. Seung, E. Shamir, and N. Tishby. Selective sampling using the query by committee algorithm. *Machine Learning*, 28:133–168, 1997.

E. Friedman. Active learning for smooth problems. In *Proceedings of the* 22nd *Conference on Learning Theory*, 2009.

R. Gangadharaiah, R. D. Brown, and J. Carbonell. Active learning in example-based machine translation. In *Proceedings of the* 17th *Nordic Conference on Computational Linguistics*, 2009.

E. Giné and V. Koltchinskii. Concentration inequalities and asymptotic results for ratio type empirical processes. *The Annals of Probability*, 34(3):1143–1216, 2006.

S. A. Goldman and M. J. Kearns. On the complexity of teaching. *Journal of Computer and System Sciences*, 50:20–31, 1995.

S. Hanneke. Teaching dimension and the complexity of active learning. In *Proceedings of the* 20th *Conference on Learning Theory*, 2007a.

S. Hanneke. A bound on the label complexity of agnostic active learning. In *Proceedings of the* 24th *International Conference on Machine Learning*, 2007b.

S. Hanneke. Adaptive rates of convergence in active learning. In *Proceedings of the* 22nd *Conference on Learning Theory*, 2009a.

S. Hanneke. *Theoretical Foundations of Active Learning*. PhD thesis, Machine Learning Department, School of Computer Science, Carnegie Mellon University, 2009b.

S. Hanneke. Rates of convergence in active learning. *The Annals of Statistics*, 39(1):333–361, 2011.

S. Har-Peled, D. Roth, and D. Zimak. Maximum margin coresets for active and noise tolerant learning. In *Proceedings of the* 20th *International Joint Conference on Artificial Intelligence*, 2007.

D. Haussler. Decision theoretic generalizations of the PAC model for neural net and other learning applications. *Information and Computation*, 100:78–150, 1992.

D. Haussler, N. Littlestone, and M. Warmuth. Predicting $\{0,1\}$-functions on randomly drawn points. *Information and Computation*, 115:248–292, 1994.

T. Hegedüs. Generalized teaching dimension and the query complexity of learning. In *Proceedings of the* 8th *Conference on Computational Learning Theory*, 1995.

L. Hellerstein, K. Pillaipakkamnatt, V. Raghavan, and D. Wilkins. How many queries are needed to learn? *Journal of the Association for Computing Machinery*, 43(5):840–862, 1996.

D. Helmbold, R. Sloan, and M. Warmuth. Learning nested differences of intersection-closed concept classes. *Machine Learning*, 5:165–196, 1990.

S. C. H. Hoi, R. Jin, J. Zhu, and M. R. Lyu. Batch mode active learning and its application to medical image classification. In *Proceedings of the* 23rd *International Conference on Machine Learning*, 2006.

M. Kääriäinen. Active learning in the non-realizable case. In *Proceedings of the* 17th *International Conference on Algorithmic Learning Theory*, 2006.

N. Karmarkar. A new polynomial-time algorithm for linear programming. *Combinatorica*, 4:373–395, 1984.

M. J. Kearns and U. Vazirani. *An Introduction to Computational Learning Theory*. The MIT Press, 1994.

M. J. Kearns, R. E. Schapire, and L. M. Sellie. Toward efficient agnostic learning. *Machine Learning*, 17:115–141, 1994.

L. G. Khachiyan. A polynomial algorithm in linear programming. *Soviet Mathematics Doklady*, 20:191–194, 1979.

V. Koltchinskii. Local Rademacher complexities and oracle inequalities in risk minimization. *The Annals of Statistics*, 34(6):2593–2656, 2006.

V. Koltchinskii. Rademacher complexities and bounding the excess risk in active learning. *Journal of Machine Learning Research*, 11:2457–2485, 2010.

V. Koltchinskii. Oracle inequalities in empirical risk minimization and sparse recovery problems. In *École d'Été de Probabilités de Saint-Flour XXXVIII-2008*. Lecture Notes in Mathematics, 2033, Springer, 2011.

S. Li. Concise formulas for the area and volume of a hyperspherical cap. *Asian Journal of Mathematics and Statistics*, 4(1):66–70, 2011.

M. Lindenbaum, S. Markovitch, and D. Rusakov. Selective sampling for nearest neighbor classifiers. *Machine Learning*, 54:125–152, 2004.

T. Luo, K. Kramer, D. B. Goldgof, L. O. Hall, S. Samson, A. Remsen, and T. Hopkins. Active learning to recognize multiple types of plankton. *Journal of Machine Learning Research*, 6: 589–613, 2005.

S. Mahalanabis. A note on active learning for smooth problems. arXiv*:1103.3095*, 2011.

E. Mammen and A. B. Tsybakov. Smooth discrimination analysis. *The Annals of Statistics*, 27: 1808–1829, 1999.

P. Massart and É. Nédélec. Risk bounds for statistical learning. *The Annals of Statistics*, 34(5): 2326–2366, 2006.

A. McCallum and K. Nigam. Employing EM in pool-based active learning for text classification. In *Proceedings of the* 15th *International Conference on Machine Learning*, 1998.

P. Mitra, C. A. Murthy, and S. K. Pal. A probabilistic active support vector learning algorithm. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(3):413–418, 2004.

J. R. Munkres. *Topology*. Prentice Hall, Inc., 2nd edition, 2000.

I. Muslea, S. Minton, and C. A. Knoblock. Active + semi-supervised learning = robust multi-view learning. In *Proceedings of the* 19th *International Conference on Machine Learning*, 2002.

R. D. Nowak. Generalized binary search. In *Proceedings of the* 46th *Annual Allerton Conference on Communication, Control, and Computing*, 2008.

L. Pitt and L. G. Valiant. Computational limitations on learning from examples. *Journal of the Association for Computing Machinery*, 35(4):965–984, 1988.

J. Poland and M. Hutter. MDL convergence speed for Bernoulli sequences. *Statistics and Computing*, 16:161–175, 2006.

G. V. Rocha, X. Wang, and B. Yu. Asymptotic distribution and sparsistency for l1-penalized parametric M-estimators with applications to linear SVM and logistic regression. *arXiv:0908.1940v1*, 2009.

D. Roth and K. Small. Margin-based active learning for structured output spaces. In *European Conference on Machine Learning*, 2006.

N. Roy and A. McCallum. Toward optimal active learning through sampling estimation of error reduction. In *Proceedings of the* 18th *International Conference on Machine Learning*, 2001.

A. I. Schein and L. H. Ungar. Active learning for logistic regression: An evaluation. *Machine Learning*, 68(3):235–265, 2007.

G. Schohn and D. Cohn. Less is more: Active learning with support vector machines. In *Proceedings of the* 17th *International Conference on Machine Learning*, 2000.

B. Settles. Active learning literature survey. *http://active-learning.net*, 2010.

S. M. Srivastava. *A Course on Borel Sets*. Springer-Verlag, 1998.

S. Tong and D. Koller. Support vector machine active learning with applications to text classification. *Journal of Machine Learning Research*, 2, 2001.

A. B. Tsybakov. Optimal aggregation of classifiers in statistical learning. *The Annals of Statistics*, 32(1):135–166, 2004.

L. G. Valiant. A theory of the learnable. *Communications of the Association for Computing Machinery*, 27(11):1134–1142, 1984.

A. W. van der Vaart and J. A. Wellner. *Weak Convergence and Empirical Processes*. Springer, 1996.

V. Vapnik. *Estimation of Dependencies Based on Empirical Data*. Springer-Verlag, New York, 1982.

V. Vapnik. *Statistical Learning Theory*. John Wiley & Sons, Inc., 1998.

V. Vapnik and A. Chervonenkis. On the uniform convergence of relative frequencies of events to their probabilities. *Theory of Probability and its Applications*, 16:264–280, 1971.

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

L. Wang. Sufficient conditions for agnostic active learnable. In *Advances in Neural Information Processing Systems 22*, 2009.

L. Wang. Smoothness, disagreement coefficient, and the label complexity of agnostic active learning. *Journal of Machine Learning Research*, 12:2269–2292, 2011.

L. Wang and X. Shen. On L1-norm multiclass support vector machines. *Journal of the American Statistical Association*, 102(478):583–594, 2007.

L. Yang, S. Hanneke, and J. Carbonell. The sample complexity of self-verifying Bayesian active learning. In *Proceedings of the* 14th *International Conference on Artificial Intelligence and Statistics*, 2011.

Journal of Machine Learning Research 13 (2012) 1589-1608 Submitted 10/11; Revised 2/12; Published 5/12

A Model of the Perception of Facial Expressions of Emotion by Humans: Research Overview and Perspectives

Aleix Martinez ALEIX@ECE.OSU.EDU
Shichuan Du DUS@ECE.OSU.EDU
Department of Electrical and Computer Engineering
The Ohio State University
2015 Neil Avenue
Columbus, OH 43210, USA

Editors: Isabelle Guyon and Vassilis Athitsos

Abstract

In cognitive science and neuroscience, there have been two leading models describing how humans perceive and classify facial expressions of emotion—the continuous and the categorical model. The continuous model defines each facial expression of emotion as a feature vector in a face space. This model explains, for example, how expressions of emotion can be seen at different intensities. In contrast, the categorical model consists of C classifiers, each tuned to a specific emotion category. This model explains, among other findings, why the images in a morphing sequence between a happy and a surprise face are perceived as either happy or surprise but not something in between. While the continuous model has a more difficult time justifying this latter finding, the categorical model is not as good when it comes to explaining how expressions are recognized at different intensities or modes. Most importantly, both models have problems explaining how one can recognize combinations of emotion categories such as happily surprised versus angrily surprised versus surprise. To resolve these issues, in the past several years, we have worked on a revised model that justifies the results reported in the cognitive science and neuroscience literature. This model consists of C distinct continuous spaces. Multiple (compound) emotion categories can be recognized by linearly combining these C face spaces. The dimensions of these spaces are shown to be mostly configural. According to this model, the major task for the classification of facial expressions of emotion is precise, detailed detection of facial landmarks rather than recognition. We provide an overview of the literature justifying the model, show how the resulting model can be employed to build algorithms for the recognition of facial expression of emotion, and propose research directions in machine learning and computer vision researchers to keep pushing the state of the art in these areas. We also discuss how the model can aid in studies of human perception, social interactions and disorders.

Keywords: vision, face perception, emotions, computational modeling, categorical perception, face detection

1. Introduction

The face is an object of major importance in our daily lives. Faces tell us the identity of the person we are looking at and provide information on gender, attractiveness and age, among many others. Of primary interest is the production and recognition of facial expressions of emotion. Emotions play a fundamental role in human cognition (Damasio, 1995) and are thus essential in studies of cognitive science, neuroscience and social psychology. Facial expressions of emotion could also

play a pivotal role in human communication (Schmidt and Cohn, 2001). And, sign languages use facial expressions to encode part of the grammar (Wilbur, 2011). It has also been speculated that expressions of emotion were relevant in human evolution (Darwin, 1872). Models of the perception of facial expressions of emotion are thus important for the advance of many scientific disciplines.

A first reason machine learning and computer vision researchers are interested in creating computational models of the perception of facial expressions of emotion is to aid studies in the above sciences (Martinez, 2003). Furthermore, computational models of facial expressions of emotion are important for the development of artificial intelligence (Minsky, 1988) and are essential in human-computer interaction (HCI) systems (Pentland, 2000).

Yet, as much as we understand how facial expressions of emotion are produced, very little is known on how they are interpreted by the human visual system. Without proper models, the scientific studies summarized above as well as the design of intelligent agents and efficient HCI platforms will continue to elude us. A HCI system that can easily recognize expressions of no interest to the human user is of limited interest. A system that fails to recognize emotions readily identified by us is worse.

In the last several years, we have defined a computational model consistent with the cognitive science and neuroscience literature. The present paper presents an overview of this research and a perspective of future areas of interest. We also discuss how machine learning and computer vision should proceed to successfully emulate this capacity in computers and how these models can aid in studies of visual perception, social interactions and disorders such as schizophrenia and autism. In particular, we provide the following discussion.

- A model of human perception of facial expressions of emotion: We provide an overview of the cognitive science literature and define a computational model consistent with it.

- Dimensions of the computational space: Recent research has shown that human used mostly shape for the perception and recognition of facial expressions of emotion. In particular, we show that configural features are of much use in this process. A configural feature is defined as a non-rotation invariant modeling of the distance between facial components; for example, the vertical distance between eyebrows and mouth.

- We argue that to overcome the current problems of face recognition algorithms (including identity and expressions), the area should make a shift toward a more shape-based modeling. Under this model, the major difficulty for the design of computer vision and machine learning systems is that of precise detection of the features, rather than classification. We provide a perspective on how to address these problems.

The rest of the paper is organized as follows. Section 2 reviews relevant research on the perception of facial expressions of emotion by humans. Section 3 defines a computational model consistent with the results reported in the previous section. Section 4 illustrates the importance of configural and shape features for the recognition of emotions in face images. Section 5 argues that the real problem in machine learning and computer vision is a detection one and emphasizes the importance of research in this domain before we can move forward with improved algorithms of face recognition. In Section 6, we summarize some of the implications of the proposed model. We conclude in Section 7.

2. Facial Expressions: From Production to Perception

The human face is an engineering marvel. Underneath our skin, a large number of muscles allow us to produce many configurations. The face muscles can be summarized as Action Unit (AU) (Ekman and Friesen, 1976) defining positions characteristic of facial expressions of emotion. These face muscles are connected to the motor neurons in the cerebral cortex through the corticobulbar track. The top muscles are connected bilaterally, while the bottom ones are connected unilaterally to the opposite hemisphere. With proper training, one can learn to move most of the face muscles independently. Otherwise, facial expressions take on predetermined configurations.

There is debate on whether these predetermined configurations are innate or learned (nature vs. nurture) and whether the expressions of some emotions is universal (Izard, 2009). By universal, we mean that people from different cultures produce similar muscle movements when expressing some emotions. Facial expressions typically classified as universal are joy, surprise, anger, sadness, disgust and fear (Darwin, 1872; Ekman and Friesen, 1976). Universality of emotions is controversial, since it assumes facial expressions of emotion are innate (rather than culturally bound). It also favors a categorical perception of facial expressions of emotion. That is, there is a finite set of predefined classes such as the six listed above. This is known as the *categorical model*.

In the categorical model, we have a set of C classifiers. Each classifier is specifically designed to recognize a single emotion label, such as surprise. Several psychophysical experiments suggest the perception of emotions by humans is categorical (Ekman and Rosenberg, 2005). Studies in neuroscience further suggest that distinct regions (or pathways) in the brain are used to recognize different expressions of emotion (Calder et al., 2001).

An alternative to the categorical model is the *continuous model* (Russell, 2003; Rolls, 1990). Here, each emotion is represented as a feature vector in a multidimensional space given by some characteristics common to all emotions. One such model is Russell's 2-dimensional circumplex model (Russell, 1980), where the first basis measures pleasure-displeasure and the second arousal. This model can justify the perception of many expressions, whereas the categorical model needs to define a class (i.e., classifier) for every possible expression. It also allows for intensity in the perception of the emotion label. Whereas the categorical model would need to add an additional computation to achieve this goal (Martinez, 2003), in the continuous model the intensity is intrinsically defined in its representation. Yet, morphs between expressions of emotions are generally classified to the closest class rather than to an intermediate category (Beale and Keil, 1995). Perhaps more interestingly, the continuous model better explains the caricature effect (Rhodes et al., 1987; Calder et al., 1997), where the shape features of someone's face are exaggerated (e..g, making a long nose longer). This is because the farther the feature vector representing that expression is from the mean (or center of the face space), the easier it is to recognize it (Valentine, 1991).

In neuroscience, the multidimensional (or continuous) view of emotions was best exploited under the limbic hypothesis (Calder et al., 2001). Under this model, there should be a neural mechanism responsible for the recognition of all facial expressions of emotion, which was assumed to take place in the limbic system. Recent results have however uncovered dissociated networks for the recognition of most emotions. This is not necessarily proof of a categorical model, but it strongly suggests that there are at least distinct groups of emotions, each following distinct interpretations.

Furthermore, humans are only very good at recognizing a number of facial expressions of emotion. The most readily recognized emotions are happiness and surprise. It has been shown that joy and surprise can be robustly identified extremely accurately at almost any resolution (Du and Mar-

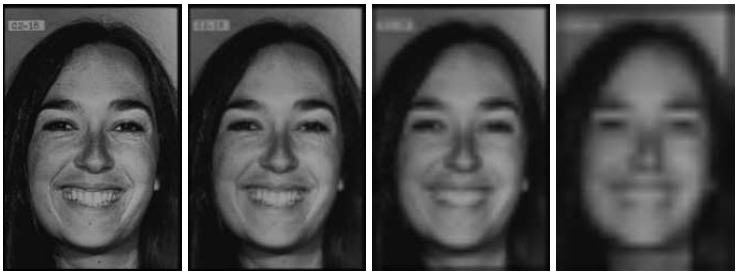


Figure 1: Happy faces at four different resolutions. From left o right: 240 by 160, 120 by 80, 60 by 40, and 30 by 20 pixels. All images have been resized to a common image size for visualization.

tinez, 2011). Figure 1 shows a happy expression at four different resolutions. The reader should not have any problem recognizing the emotion in display even at the lowest of resolutions. However, humans are not as good at recognizing anger and sadness and are even worse at fear and disgust.

A major question of interest is the following. Why are some facial configurations more easily recognizable than others? One possibility is that expressions such as joy and surprise involve larger face transformations than the others. This has recently proven not to be the case (Du and Martinez, 2011). While surprise does have the largest deformation, this is followed by disgust and fear (which are poorly recognized). Learning why some expressions are so readily classified by our visual system should facilitate the definition of the form and dimensions of the computational model of facial expressions of emotion.

The search is on to resolve these two problems. First, we need to determine the *form* of the computational space (e.g., a continuous model defined by a multidimensional space). Second, we ought to define the *dimensions* of this model (e.g., the dimensions of this multidimensional face space are given by configural features). In the following sections we overview the research we have conducted in the last several years leading to a solution to the above questions. We then discuss on the implications of this model. In particular, we provide a perspective on how machine learning and computer vision researcher should move forward if they are to define models based on the perception of facial expressions of emotion by humans.

3. A Model of the Perception of Facial Expressions of Emotion

In cognitive science and neuroscience researchers have been mostly concerned with models of the perception and classification of the six facial expressions of emotion listed above. Similarly, computer vision and machine learning algorithms generally employ a face space to represent these six emotions. Sample feature vectors or regions of this feature space are used to represent each of these six emotion labels. This approach has a major drawback—it can only detect one emotion from a single image. In machine learning, this is generally done by a winner-takes-all approach (Torre and Cohn, 2011). This means that when a new category wants to be included, one generally needs to provide labeled samples of it to the learning algorithm.

Yet, everyday experience demonstrates that we can perceive more than one emotional category in a single image (Martinez, 2011), even if we have no prior experience with it. For example,

Figure 2: Faces expressing different surprise. From left to right: happily surprised, sadly surprised, angrily surprised, fearfully surprised, disgustedly surprised, and surprise.

Figure 2 shows images of faces expressing different surprises—happily surprised, angrily surprised, fearfully surprised, disgustedly surprised and the typically studied surprise.

If we were to use a continuous model, we would need to have a very large number of labels represented all over the space; including all possible types of surprises. This would require a very large training set, since each possible combination of labels would have to be learned. But this is the same problem a categorical model would face. In such a case, dozens if not hundreds of sample images for each possible category would be needed. Alternatively, Susskind et al. (2007) have shown that the appearance of a continuous model may be obtained from a set of classifiers defining a small number of categories.

If we define an independent computational (face) space for a small number of emotion labels, we will only need sample faces of those few facial expressions of emotion. This is indeed the approach we have taken. Details of this model are given next.

Key to this model is to note that we can define new categories as linear combinations of a small set of categories. Figure 3 illustrates this approach. In this figure, we show how we can obtain the above listed different surprises as a linear combination of known categories. For instance, happily surprised can be defined as expressing 40% joy plus 60% surprise, that is, expression = .4 happy + .6 surprise. A large number of such expressions exist that are a combination of the six emotion categories listed above and, hence, the above list of six categories is a potential set of basic emotion classes. Also, there is some evidence form cognitive science to suggest that these are important categories for humans (Izard, 2009) Of course, one needs not base the model on this set of six emotions. This is an area that will undoubtedly attract lots of interest. A question of particular interest is to determine not only which basic categories to include in the model but how many. To this end both, cognitive studies with humans and computational extensions of the proposed model will be necessary, with the results of one area aiding the research of the other.

The approach described in the preceding paragraph would correspond to a categorical model. However, we now go one step further and define each of these face spaces as continuous feature spaces, Figure 3. This allows for the perception of each emotion at different intensities, for example, less happy to exhilarant (Neth and Martinez, 2010). Less happy would correspond to a feature vector (in the left most face space in the figure) closer to the mean (or origin of the feature space). Feature vectors farther from the mean would be perceived as happier. The proposed model also explains the caricature effect, because within each category the face space is continuous and exaggerating the

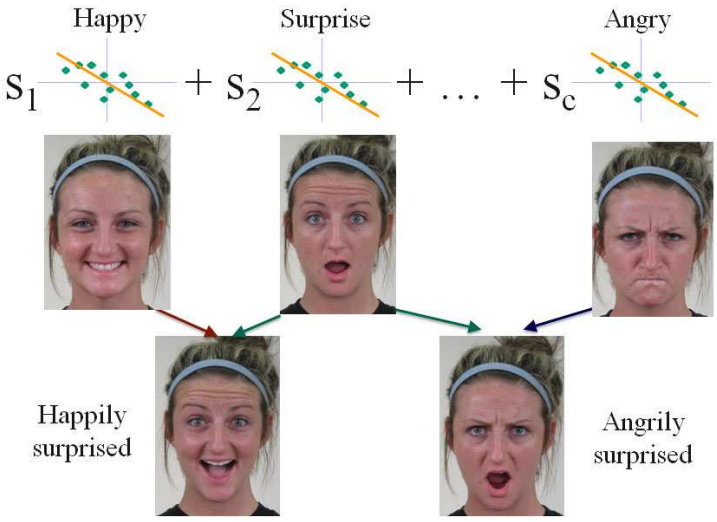


Figure 3: This figure shows how to construct linear combinations of known categories. At the top of the figure, we have the known or learned categories (emotions). The coefficients s_i determine the contribution of each of these categories to the final perception of the emotion.

expression will move the feature vector representing the expression further from the mean of that category.

Furthermore, the proposed model can define new terms, for example, "hatred" which is defined as having a small percentage of disgust and a larger percentage of anger; still linear. In essence, the intensity observed in this *continuous representation* defines the weight of the contribution of each basic category toward the final decision (classification). It also allows for the representation and recognition of a very large number of emotion categories without the need to have a categorical space for each or having to use many samples of each expression as in the continuous model.

The proposed model thus bridges the gap between the categorical and continuous ones and resolves most of the debate facing each of the models individually. To complete the definition of the model, we need to specify what defines each of the dimensions of the continuous spaces representing each category. We turn to this problem in the next section.

4. Dimensions of the Model

In the early years of computer vision, researchers derived several feature- and shape-based algorithms for the recognition of objects and faces (Kanade, 1973; Marr, 1976; Lowe, 1983). In these methods, geometric, shape features and edges were extracted from an image and used to build a model of the face. This model was then fitted to the image. Good fits determined the class and position of the face.

Later, the so-called appearance-based approach, where faces are represented by their pixel-intensity maps or the response of some filters (e.g., Gabors), was studied (Sirovich and Kirby,

1987). In this alternative texture-based approach, a metric is defined to detect and recognize faces in test images (Turk and Pentland, 1991). Advances in pattern recognition and machine learning have made this the preferred approach in the last two decades (Brunelli and Poggio, 1993).

Inspired by this success, many algorithms developed in computer vision for the recognition of expressions of emotion have also used the appearance-based model (Torre and Cohn, 2011). The appearance-based approach has also gained momentum in the analysis of AUs from images of faces. The main advantage of the appearance-based model is that one does not need to predefine a feature or shape model as in the earlier approaches. Rather, the face model is inherently given by the training images.

The appearance-based approach does provide good results from near-frontal images of a reasonable quality, but it suffers from several major inherent problems. The main drawback is its sensitivity to image manipulation. Image size (scale), illumination changes and pose are all examples of this. Most of these problems are intrinsic to the definition of the approach since this cannot generalize well to conditions not included in the training set. One solution would be to enlarge the number of training images (Martinez, 2002). However, learning from very large data sets (in the order of millions of samples) is, for the most part, unsolved (Lawrence, 2005). Progress has been made in learning complex, non-linear decision boundaries, but most algorithms are unable to accommodate large amounts of data—either in space (memory) or time (computation).

This begs the question as to how the human visual system solves the problem. One could argue that, throughout evolution, the homo genus (and potentially before it) has been exposed to trillions of faces. This has facilitated the development of simple, yet robust algorithms. In computer vision and machine learning, we wish to define algorithms that take a shorter time to learn a similarly useful image representation. One option is to decipher the algorithm used by our visual system. Research in face recognition of identity suggests that the algorithm used by the human brain is not appearance-based (Wilbraham et al., 2008). Rather, it seems that, over time, the algorithm has identified a set of robust features that facilitate rapid categorization (Young et al., 1987; Hosie et al., 1988; Barlett and Searcy, 1993).

This is also the case in the recognition of facial expressions of emotion (Neth and Martinez, 2010). Figure 4 shows four examples. These images all bear a neutral expression, that is, an expression associated to no emotion category. Yet, human subjects perceive them as expressing sadness, anger, surprise and disgust. The most striking part of this illusion is that these faces do not and cannot express any emotion, since all relevant AUs are inactive. This effect is called over-generalization (Zebrowitz et al., 2010), since human perception is generalizing the learned features defining these face spaces over to images with a different label.

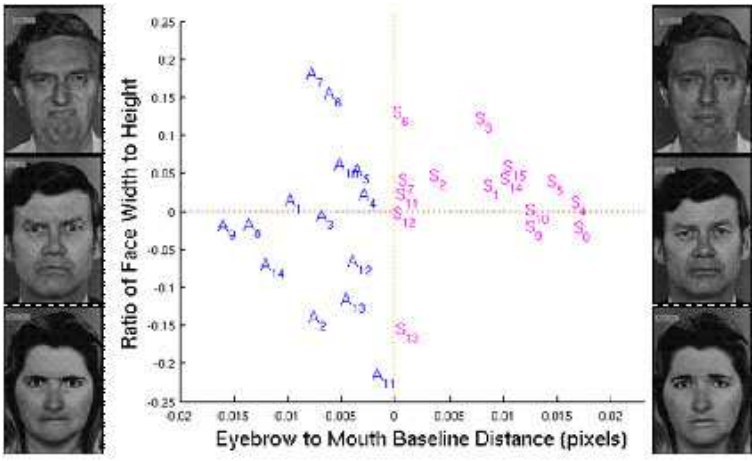
The images in Figure 4 do have something in common though—they all include a configural transformation. What the human visual system has learned is that faces do not usually look like those in the image. Rather the relationship (distances) between brows, nose, mouth and the contour of the face is quite standard. They follow a Gaussian distribution with small variance (Neth and Martinez, 2010). The images shown in this figure however bear uncanny distributions of the face components. In the sad-looking example, the distance between the brows and mouth is larger than normal (Neth and Martinez, 2009) and the face is thinner than usual (Neth and Martinez, 2010). This places this sample face, most likely, outside the 99% confidence interval of all Caucasian faces on these two measures. The angry-looking face has a much-shorter-than-average brow to mouth distance and a wide face. While the surprise-looking face has a large distance between eyes and

Figure 4: The four face images and schematics shown above all correspond to neutral expressions (i.e., the sender does not intend to convey any emotion to the receiver). Yet, most human subjects interpret these faces as conveying anger, sadness, surprise and disgust. Note that although these faces look very different from one another, three of them are actually morphs from the same (original) image.

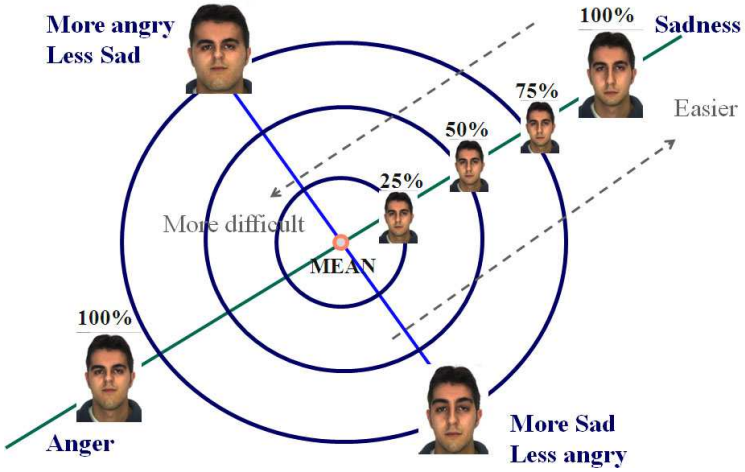
brows and a thinner face. The disgust-looking face has a shorter distance between brows, eyes, nose and mouth. These effects are also clear in the schematic faces shown in the figure.

Yet, configural cues alone are not sufficient to create an impressive, lasting effect. Other shape changes are needed. For example, the curvature of the mouth in joy or the opening of the eyes—showing additional sclera—in surprise. Note how the surprise-looking face in Figure 4 appears to also express disinterest or sleepiness. Wide-open eyes would remove these perceptions. But this can only be achieved with a shape change. Hence, our face spaces should include both, configural and shape features. It is important to note that configural features can be obtained from an appropriate representation of shape. Expressions such as fear and disgust seem to be mostly (if not solely) based on shape features, making recognition less accurate and more susceptible to image manipulation. We have previously shown (Neth and Martinez, 2010) that configural cues are amongst the most discriminant features in a classical (Procrustes) shape representation, which can be made invariant to 3D rotations of the face (Hamsici and Martinez, 2009a).

Thus, each of the six categories of emotion (happy, sad, surprise, angry, fear and disgust) is represented in a shape space given by classical statistical shape analysis. First the face and the shape of the major facial components are automatically detected. This includes delineating the brows, eyes, nose, mouth and jaw line. The shape is then sample with d equally spaced landmark points. The mean (center of mass) of all the points is computed. The $2d$-dimensional shape feature vector is given by the x and y coordinates of the d shape landmarks subtracted by the mean and divided by its norm. This provides invariance to translation and scale. 3D rotation invariance can be achieved with the inclusion of a kernel as defined in Hamsici and Martinez (2009a). The dimensions of each emotion category can now be obtained with the use of an appropriate discriminant analysis method. We use the algorithm defined by Hamsici and Martinez (2008) because it minimizes the Bayes classification error.



(a)



(b)

Figure 5: (a) Shown here are the two most discriminant dimensions of the face shape vectors. We also plot the images of anger and sadness of Ekman and Friesen (1976). In dashed are simple linear boundaries separating angry and sad faces according to the model. The first dimension (distance between brows and mouth) successfully classifies 100% of the sample images. This continuous model is further illustrated in (b). Note that, in the proposed computational model, the face space defining sadness corresponds to the right-bottom quadrant, while that of anger is given by the left-top quadrant. The dashed arrows in the figure reflect the fact that as we move away from the "mean" (or norm) face, recognition of that emotion become easier.

As an example, the approach detailed in this section identifies the distance between the brows and mouth and the width of the face as the two most important shape features of anger and sadness. It is important to note that, if we reduce the computational spaces of anger and sadness to 2-dimensions, they are almost indistinguishable. Thus, it is possible that these two categories are in fact connected by a more general one. This goes back to our question of the number of basic categories used by the human visual system. The face space of anger and sadness is illustrated in Figure 5, where we have also plotted the feature vectors of the face set of Ekman and Friesen (1976).

As in the above, we can use the shape space defined above to find the two most discriminant dimensions separating each of the six categories listed earlier. The resulting face spaces are shown in Figure 6. In each space, a simple linear classifier in these spaces can successfully classify each emotion very accurately. To test this, we trained a linear support vector machine (Vapnik, 1998) and use the leave-one-out test on the data set of images of Ekman and Friesen (1976). Happiness is correctly classified 99% of the time. Surprise and disgust 95% of the time. Sadness 90% and anger 94%. While fear is successfully classified at 92%. Of course, adding additional dimensions in the feature space and using nonlinear classifiers can readily achieve perfect classification (i.e., 100%). The important point from these results is to note that simple configural features can *linearly* discriminate most of the samples in each emotion. These features are very robust to image degradation and are thus ideal for recognition in challenging environments (e.g., low resolution)—a message to keep in mind for the development of machine learning and computer vision systems.

5. Precise Detection of Faces and Facial Features

As seen thus far, human perception is extremely tuned to small configural and shape changes. If we are to develop computer vision and machine learning systems that can emulate this capacity, the real problem to be addressed by the community is that of *precise detection of faces and facial features* (Ding and Martinez, 2010). Classification is less important, since this is embedded in the detection process; that is, we want to precisely detect changes that are important to recognize emotions.

Most computer vision algorithms defined to date provide, however, inaccurate detections. One classical approach to detection is template matching. In this approach, we first define a template (e.g., the face or the right eye or the left corner of the mouth or any other feature we wish to detect). This template is learned from a set of sample images; for example, estimating the distribution or manifold defining the appearance (pixel map) of the object (Yang et al., 2002). Detection of the object is based on a window search. That is, the learned template is compared to all possible windows in the image. If the template and the window are similar according to some metric, then the bounding box defining this window marks the location and size (scale) of the face. The major drawback of this approach is that it yields imprecise detections of the learned object, because a window of an non-centered face is more similar to the learned template than a window with background (say, a tree). An example of this result is shown in Figure 7.

A solution to the above problem is to learn to discriminate between non-centered windows of the objects and well centered ones (Ding and Martinez, 2010). In this alternative, a non-linear classifier (or some density estimator) is employed to discriminate the region of the feature space defining well-centered windows of the objects and non-centered ones. We call these non-centered windows the context of the object, in the sense that these windows provide the information typically found around the object but do not correspond to the actual face. This features versus context idea is illustrated in Figure 8. This approach can be used to precisely detect faces, eyes, mouth, or any

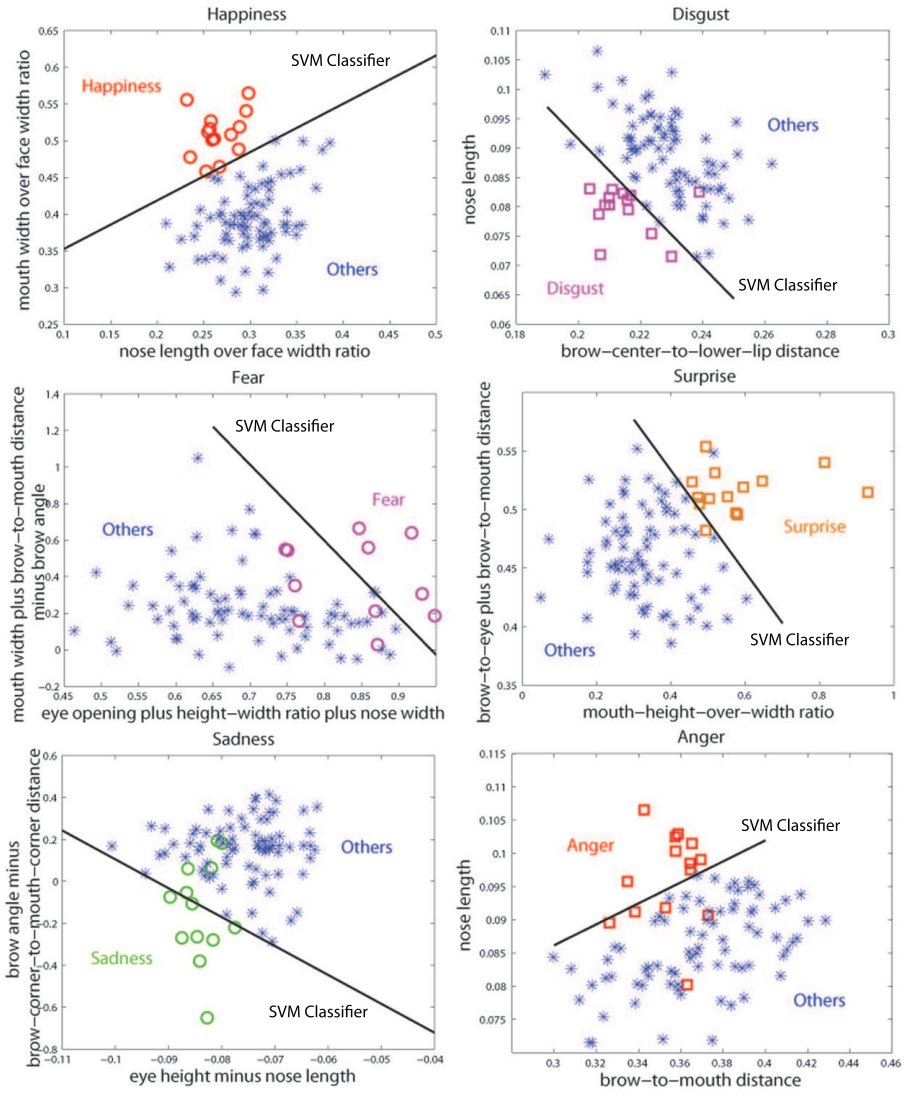


Figure 6: Shown in the above are the six feature spaces defining each of the six basic emotion categories. A simple linear Support Vector Machine (SVM) can achieve high classification accuracies; where we have used a one-versus-all strategy to construct each classifier and tested it using the leave-one-out strategy. Here, we only used two features (dimensions) for clarity of presentation. Higher accuracies are obtained if we include additional dimensions and training samples.

Figure 7: Two example of imprecise detections of a face with a state of the art algorithm.

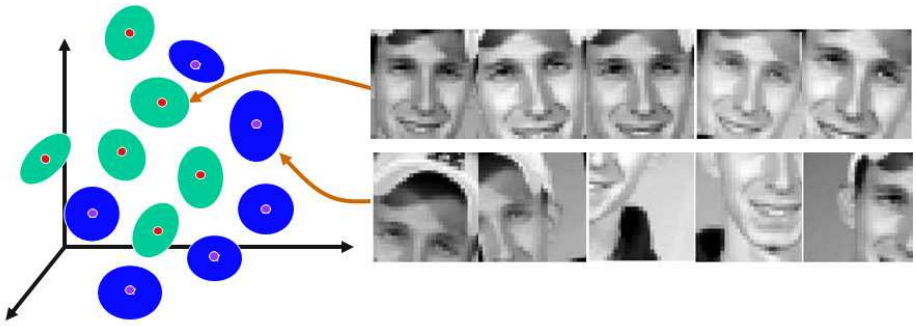


Figure 8: The idea behind the features versus context approach is to learn to discriminate between the feature we wish to detect (e.g., a face, an eye, etc.) and poorly detected versions of it. This approach eliminates the classical overlapping of multiple detections around the object of interest at multiple scales. At the same time, it increases the accuracy of the detection because we are moving away from poor detections and toward precise ones.

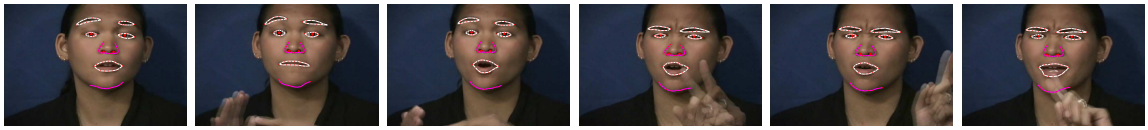


Figure 9: Precise detections of faces and facial features using the algorithm of (Ding and Martinez, 2010).

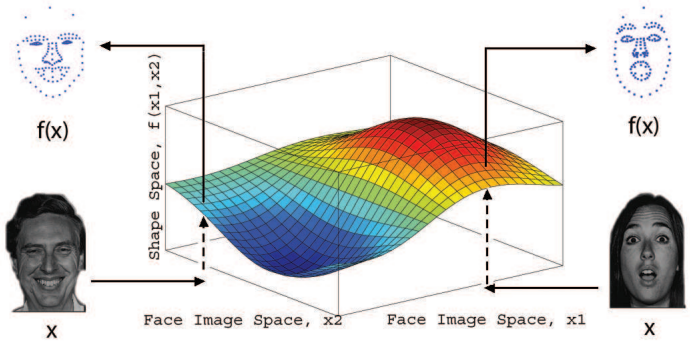


Figure 10: Manifold learning is ideal for learning mappings between face (object) images and their shape description vectors.

other facial feature where there is a textural discrimination between it and its surroundings. Figure 9 shows some sample results of accurate detection of faces and facial features with this approach.

The same features versus context idea can be applied to other detection and modeling algorithms, such as Active Appearance Models (AAM) (Cootes et al., 2001). AAM use a linear model—usually based on Principal Component Analysis (PCA)—to learn the relationship between the shape of an object (e.g., a face) and its texture. One obvious limitation is that the learned model is linear. A solution to this problem is to employ a kernel map. Kernel PCA is one option. Once we have introduced a kernel we can move one step further and use it to address additional issues of interest. A first capability we may like to add to a AAM is the possibility to work with three-dimensions. The second could be to omit the least-squares iterative nature of the Procrustes alignment required in most statistical shape analysis methods such as AAM. An approach that successfully addresses these problem uses a set of kernels called Rotation Invariant Kernels (RIK) (Hamsici and Martinez, 2009a). RIK add yet another important advantage to shape analysis: they provide rotation invariance. Thus, once the shape is been mapped to the RIK space, objects (e.g., faces) are invariant to translation, scale and rotation. These kernels are thus very attractive for the design of AAM algorithms (Hamsici and Martinez, 2009b).

By now we know that humans are very sensitive to small changes. But we do not yet know how sensitive (or accurate). Of course, it is impossible to be pixel accurate when marking the boundaries of each facial feature, because edges blur over several pixels. This can be readily observed by zooming in the corner of an eye. To estimate the accuracy of human subjects, we performed the following experiment. First, we designed a system that allows users to zoom in at any specified location to facilitate delineation of each of the facial features manually. Second, we asked three people (herein referred to as judges) to manually delineate each of the facial components of close to $4,000$ images of faces. Third, we compared the markings of each of the three judges. The within-judge variability was (on average) 3.8 pixels, corresponding to a percentage of error of 1.2% in terms of the size of the face. This gives us an estimate of the accuracy of the manual detections. The average error of the algorithm of Ding and Martinez (2010) is 7.3 pixels (or 2.3%), very accurate but still far short of what humans can achieve. Thus, further research is needed to develop computer vision algorithms that can extract even more accurate detection of faces and its components.

Figure 11: Shape detection examples at different resolutions. Note how the shape estimation is almost as good regardless of the resolution of the image.

Another problem is what happens when the resolution of the image diminishes. Humans are quite robust to these image manipulations (Du and Martinez, 2011). One solution to this problem is to use manifold learning. In particular, we wish to define a non-linear mapping $f(.)$ between the image of a face and its shape. This is illustrated in Figure 10. That is, given enough sample images and their shape feature vectors described in the preceding section, we need to find the function which relates the two. This can be done, for example, using kernel regression methods (Rivera and Martinez, 2012). One of the advantages of this approach is that this function can be defined to detect shape from very low resolution images or even under occlusions. Occlusions can be "learned" by adding synthetic occlusions or missing data in the training samples but leaving the shape feature vector undisturbed (Martinez, 2002). Example detections using this approach are shown in Figure 11.

One can go one step further and recover the three-dimensional information when a video sequence is available (Gotardo and Martinez, 2011a). Recent advances in non-rigid structure from motion allow us to recover very accurate reconstructions of both the shape and the motion even under occlusion. A recent approach resolves the nonlinearity of the problem using kernel mappings (Gotardo and Martinez, 2011b).

Combining the two approaches to detection defined in this section should yield even more accurate results in low-resolution images and under occlusions or other image manipulations. We hope that more research will be devoted to this important topic in face recognition.

The approaches defined in this section are a good start, but much research is needed to make these systems comparable to human accuracies. We argue that research in machine learning should address these problems rather than the typical classification one. A first goal is to define algorithms

that can detect face landmarks very accurately even at low resolutions. Kernel methods and regression approaches are surely good solutions as illustrated above. But more targeted approaches are needed to define truly successful computational models of the perception of facial expressions of emotion.

6. Discussion

In the real world, occlusions and unavoidable imprecise detections of the fiducial points, among others, are known to affect recognition (Torre and Cohn, 2011; Martinez, 2003). Additionally, some expressions are, by definition, ambiguous. Most importantly though seems to be the fact that people are not very good at recognizing facial expressions of emotion even under favorable condition (Du and Martinez, 2011). Humans are very robust at detection joy and surprise from images of faces; regardless of the image conditions or resolution. However, we are not as good at recognizing anger and sadness and are worst at fear and disgust.

The above results suggest that there could be three groups of expressions of emotion. The first group is intended for conveying emotions to observers. These expressions have evolved a facial construct (i.e., facial muscle positions) that is distinctive and readily detected by an observer at short or large distances. Example expressions in this group are happiness and surprise. A computer vision system—especially a HCI—should make sure these expressions are accurately and robustly recognized across image degradation. Therefore, we believe that work needs to be dedicated to make systems very robust when recognizing these emotions.

The second group of expressions (e.g., anger and sadness) is reasonably recognized at close proximity only. A computer vision system should recognize these expressions in good quality images, but can be expected to fail as the image degrades due to resolution or other image manipulations. An interesting open question is to determine why this is the case and what can be learned about human cognition from such a result.

The third and final group of emotions constitutes those at which humans are not very good recognizers. This includes expressions such as fear and disgust. Early work (especially in evolutionary psychology) had assumed that recognition of fear was primal because it served as a necessary survival mechanism (LeDoux, 2000). Recent studies have demonstrated much the contrary. Fear is generally poorly recognized by healthy human subjects (Smith and Schyns, 2009; Du and Martinez, 2011). One hypothesis is that expressions in this group have evolved for other than communication reasons. For example, it has been proposed that fear opens sensory channels (i.e., breathing in and wide open eye), while disgust closes them (i.e., breathing out and closed eyes) (Susskind et al., 2008). Under this model, the receiver has learned to identify those face configurations to some extent, but without the involvement of the sender—modifying the expression to maximize transmission of information through a noisy environment—the recognition of these emotions has remained poor. Note that people can be trained to detect such changes quite reliably (Ekman and Rosenberg, 2005), but this is not the case for the general population.

Another area that will require additional research is to exploit other types of facial expressions. Facial expressions are regularly used by people in a variety of setting. More research is needed to understand these. Moreover, it will be important to test the model in natural occurring environments. Collection and handling of this data poses several challenges, but the research described in these pages serves as a good starting point for such studies. In such cases, it may be necessary to go beyond a linear combination of basic categories. However, without empirical proof for the need

of something more complex than linear combinations of basic emotion categories, such extensions are unlikely. The cognitive system has generally evolved the simplest possible algorithms for the analysis or processing of data. Strong evidence of more complex models would need to be collected to justify such extensions. One way to do this is by finding examples that cannot be parsed by the current model, suggesting a more complex structure is needed.

It is important to note that these results will have many applications in studies of agnosias and disorders. Of particular interest are studies of depression or anxiety disorders. Depression afflicts a large number of people in the developed countries. Models that can help us better understand its cognitive processes, behaviors and patterns could be of great importance for the design of coping mechanisms. Improvements may also be possible if it were to better understand how facial expressions of emotion affect these people. Other syndromes such as autism are also of great importance these days. More children than ever are being diagnosed with the disorder (CDC, 2012; Prior, 2003). We know that autistic children do not perceive facial expressions of emotion as others do (Jemel et al., 2006) (but see Castelli, 2005). A modified computational model of the perception of facial expressions of emotion in autism could help design better teaching tools for this group and may bring us closer to understanding the syndrome.

There are indeed many great possibilities for machine learning researchers to help move these studies forward. Extending or modifying the modeled summarized in the present paper is one way. Developing machine learning algorithms to detect face landmark more accurately is another. Developing statistical tools that more accurately represent the underlying manifold or distribution of the data is yet another great way to move the state of the art forward.

7. Conclusions

In the present work we have summarized the development of a model of the perception of facial expressions of emotion by humans. A key idea in this model is to linearly combine a set of face spaces defining some basic emotion categories. The model is consistent with our current understanding of human perception and can be successfully exploited to achieve great recognition results for computer vision and HCI applications. We have shown how, to be consistent with the literature, the dimensions of these computational spaces need to encode configural and shape features.

We conclude that to move the state of the art forward, face recognition research has to focus on a topic that has received little attention in recent years—precise, detailed detection of faces and facial features. Although we have focused our study on the recognition of facial expressions of emotion, we believe that the results apply to most face recognition tasks. We have listed a variety of ways in which the machine learning community can get involved in this research project and briefly discussed applications in the study of human perception and the better understanding of disorders.

Acknowledgments

This research was supported in part by the National Institutes of Health, grants R01 EY 020834 and R21 DC 011081.

References

J. C. Barlett and J. Searcy. Inversion and configuration of faces. *Cognitive Psychology*, 25(3): 281–316, 1993.

J. M. Beale and F. C. Keil. Categorical effects in the perception of faces. *Cognition*, 57:217–239, 1995.

R. Brunelli and T. Poggio. Face recognition: features versus templates. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15(10):1042–1052, 1993.

A. J. Calder, A. W. Young, D. Rowland, and D. I. Perrett. Computer-enhanced emotion in facial expressions. *Proceedings of the Royal Society of London B*, 264:919–925, 1997.

A. J. Calder, A. D. Lawrence, and A. W. Young. Neuropsychology of fear and loathing. *Nature Review Neuroscience*, 2:352–363, 2001.

F. Castelli. Understanding emotions from standardized facial expressions in autism and normal development. *Autism*, 9:428–449, 2005.

CDC. Center for Disease Control and Prevention. Prevalence of autism spectrum disorders autism and developmental disabilities monitoring network, 14 sites, united states, 2008. *Morbidity and Mortality Weekly Report (MMWR)*, 61, 2012.

T. F. Cootes, G. J. Edwards, and C. J. Taylor. Active appearance models. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 23(6):681–685, 2001.

A. R. Damasio. *Descartes' Error: Emotion, Reason, and the Human Brain*. G. P. Putnam's Sons, New York, 1995.

C. Darwin. *The Expression of the Emotions in Man and Animal*. J. Murray., London, 1872.

L. Ding and A. M. Martinez. Features versus context: An approach for precise and detailed detection and delineation of faces and facial features. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 32:2022–2038, 2010.

S. Du and A. M. Martinez. The resolution of facial expressions of emotion. *Journal of Vision*, 11(13):24, 2011.

P. Ekman and W.V. Friesen. *Pictures of Facial Affect*. Consulting Psychologists Press, Palo Alto, CA, 1976.

P. Ekman and E.L. Rosenberg. *What the Face Reveals: Basic and Applied Studies of Spontaneous Expression Using the Facial Action Coding System (FACS)*. Oxford University Press, New York, 2nd edition, 2005.

P. F. U. Gotardo and A. M. Martinez. Computing smooth time-trajectories for camera and deformable shape in structure from motion with occlusion. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(10):2051–2065, 2011a.

P. F. U. Gotardo and A. M. Martinez. Kernel non-rigid structure from motion. In *Proceedings of the IEEE International Conference on Computer Vision*, 2011b.

O. C. Hamsici and A. M. Martinez. Bayes optimality in linear discriminant analysis. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 30:647–657, 2008.

O. C. Hamsici and A. M. Martinez. Rotation invariant kernels and their application to shape analysis. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 31:1985–1999, 2009a.

O. C. Hamsici and A. M. Martinez. Active appearance models with rotation invariant kernels. In *IEEE Proc. International Conference on Computer Vision*, 2009b.

J. A. Hosie, H. D. Ellis, and N. D. Haig. The effect of feature displacement on the perception of well-known faces. *Perception*, 17(4):461–474, 1988.

C. E. Izard. Emotion theory and research: Highlights, unanswered questions, and emerging issues. *Annual Review of Psychology*, 60:1–25, 2009.

B. Jemel, L. Mottron, and M. Dawson. Impaired face processing in autism: Fact or artifact? *Journal of Autism and Developmental Disorders*, 36:91–106, 2006.

T. Kanade. *Picture Processing System by Computer Complex and Recognition of Human Faces*. PhD thesis, Kyoto University, Japan, 1973.

N. Lawrence. Probabilistic non-linear principal component analysis with gaussian process latent variable models. *Journal of Machine Learning Research*, (6):1783–1816, 2005.

J.E. LeDoux. Emotion circuits in the brain. *Annual Review of Nueroscience*, 23:155–184, 2000.

D. Lowe. Three-dimensional object recognition from single two-dimensional images. *Artificial Intelligence*, 31(3):355–395, 1983.

D. Marr. Early processing of visual information. *Philosophical Transactions of the Royal Society of London*, 275(942):483–519, 1976.

A. M. Martinez. Recognizing imprecisely localized, partially occluded and expression variant faces from a single sample per class. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 24(6): 748–763, 2002.

A. M. Martinez. Matching expression variant faces. *Vision Research*, 43:1047–1060, 2003.

A. M. Martinez. Deciphering the face. In *Proc. IEEE Conf. Computer Vision and Pattern Recognition, workshop*, 2011.

M. Minsky. *The Society of Mind*. Simon & Schuster, New York, N.Y., 1988.

D. Neth and A. M. Martinez. Emotion perception in emotionless face images suggests a norm-based representation. *Journal of Vision*, 9(1):1–11, 2009.

D. Neth and A. M. Martinez. A computational shape-based model of anger and sadness justifies a configural representation of faces. *Vision Research*, 50:1693–1711, 2010.

A. Pentland. Looking at people: Sensing for ubiquitous and wearable computing. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(1):107–119, 2000.

M. Prior. Is there an increase in the prevalence of autism spectrum disorders? *Journal of Paediatrics and Child Health*, 39:81–82, 2003.

G. Rhodes, S. Brennan, and S. Carey. Identification and ratings of caricatures: implications for mental representations of faces. *Cognitive Psychology*, 19:473–497, 1987.

S. Rivera and A. M. Martinez. Learning shape manifolds. *Pattern Recognition*, 45(4):1792–1801, 2012.

E. T. Rolls. A theory of emotion, and its application to understanding the neural basis of emotion. *Cognition and Emotion*, 4:161–190, 1990.

J. A. Russell. A circumplex model of affect. *J. Personality Social. Psych.*, 39:1161–1178, 1980.

J. A. Russell. Core affect and the psychological construction of emotion. *Psychological Review*, 110:145–172, 2003.

K.L. Schmidt and J.F. Cohn. Human facial expressions as adaptations: Evolutionary questions in facial expression. *Yearbook of Physical Anthropology*, 44:3–24, 2001.

L. Sirovich and M. Kirby. Low-dimensional procedure for the characterization of human faces. *J. Optical Soc. Am. A*, 4:519–524, 1987.

F.W. Smith and P.G. Schyns. Smile through your fear and sadness: Transmitting and identifying facial expression signals over a range of viewing distances. *Psychology Science*, 20(10):1202–1208, 2009.

J. Susskind, D. Lee, A. Cusi, R. Feinman, W. Grabski, and A.K. Anderson. Expressing fear enhances sensory acquisition. *Nature Neuroscience*, 11(7):843–850, 2008.

J.M. Susskind, G. Littlewort, M.S. Bartlett, and A.K. Anderson J. Movellanb. Human and computer recognition of facial expressions of emotion. *Neuropsychologia*, 45:152162, 2007.

F. Dela Torre and J. F. Cohn. Facial expression analysis. In Th. B. Moeslund, A. Hilton, V. Kruger, and L. Sigal, editors, *Guide to Visual Analysis of Humans: Looking at People*, pages 377–410. Springer, 2011.

M. Turk and A. Pentland. Eigenfaces for recognition. *J. Cognitive Neuroscience*, 3:71–86, 1991.

T. Valentine. A unified account of the effects of distinctiveness, inversion, and race in face recognition. *Quarterly Journal of Experimental Psychology A: Human Experimental Psychology*, 43: 161–204, 1991.

V. Vapnik. *Statistical Learning Theory*. John Wiley and Sons, New York, NY, 1998.

D. A. Wilbraham, J. C. Christensen, A. M. Martinez, and J. T. Todd. Can low level image differences account for the ability of human observers to discriminate facial identity? *Journal of Vision*, 8 (5):1–12, 2008.

R. B. Wilbur. Nonmanuals, semantic operators, domain marking, and the solution to two outstanding puzzles in asl. In *Nonmanuals in Sign Languages*. John Benjamins, 2011.

M.-H. Yang, D. J. Kriegman, and N. Ahuja. Detecting faces in images: A survey. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 24(1):34–58, 2002.

A. W. Young, D. Hellawell, and D. C. Hay. Configurational information in face perception. *Perception*, 16(6):747–759, 1987.

L.A. Zebrowitz, M. Kikuchi, and J.M. Fellous. Facial resemblance to emotions: Group differences, impression effects, and race stereotypes. *Journal of Personality and Social Psychology*, 98(2): 175–189, 2010.

Journal of Machine Learning Research 13 (2012) 1609-1638 Submitted 10/10; Revised 1/12; Published 5/12

A Unifying Probabilistic Perspective for Spectral Dimensionality Reduction: Insights and New Models

Neil D. Lawrence[*] N.LAWRENCE@DCS.SHEF.AC.UK
Department of Computer Science
University of Sheffield
Regent Court
Portobello
S1 4DP
United Kingdom

Editor: Tony Jebara

Abstract

We introduce a new perspective on spectral dimensionality reduction which views these methods as Gaussian Markov random fields (GRFs). Our unifying perspective is based on the maximum entropy principle which is in turn inspired by maximum variance unfolding. The resulting model, which we call maximum entropy unfolding (MEU) is a nonlinear generalization of principal component analysis. We relate the model to Laplacian eigenmaps and isomap. We show that parameter fitting in the locally linear embedding (LLE) is approximate maximum likelihood MEU. We introduce a variant of LLE that performs maximum likelihood exactly: Acyclic LLE (ALLE). We show that MEU and ALLE are competitive with the leading spectral approaches on a robot navigation visualization and a human motion capture data set. Finally the maximum likelihood perspective allows us to introduce a new approach to dimensionality reduction based on L1 regularization of the Gaussian random field via the graphical lasso.

1. Introduction

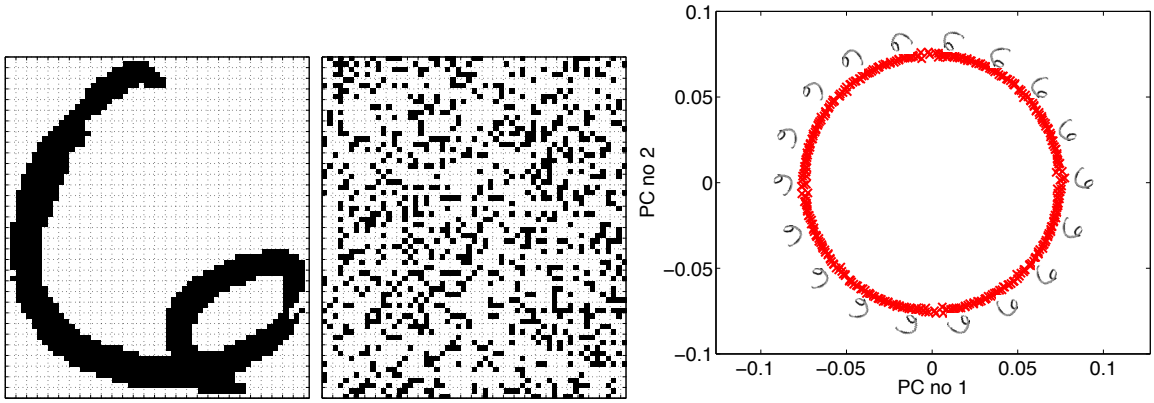
A representation of an object for processing by computer typically requires that object to be summarized by a series of features, represented by numbers. As the representation becomes more complex, the number of features required typically increases. Examples include: the characteristics of a customer in a database; the pixel intensities in an image; a time series of angles associated with data captured from human motion for animation; the energy at different frequencies (or across the cepstrum) as a time series for interpreting speech; the frequencies of given words as they appear in a set of documents; the level of expression of thousands of genes, across a time series, or for different diseases.

With the increasing complexity of the representation, the number of features that are stored also increases. Data of this type is known as high dimensional data.

Consider the simple example of a handwritten six. The six in Figure 1 is represented on a grid of pixels which is 64 rows by 57 columns, giving a datum with 3,648 dimensions. The space in which this digit sits contains far more than the digit. Imagine a simple probabilistic model of the digit which assumes that each pixel in the image is independent and is on with a given probability. We can sample from such a model (Figure 1(b)).

Even if we were to sample every nanosecond from now until the end of the universe we would be highly unlikely to see the original six. The space covered by this model is very large but fundamentally the data lies on low dimensional embedded space. This is illustrated in Figure 1(c). Here a data set has been constructed by rotating the digit 360 times in one degree intervals. The data is then projected onto its first two principal components. The spherical structure of the rotation is clearly visible in the projected data. Despite the

∗. Also in the Sheffield Institute of Translational Neuroscience.



(a) (b) (c)

Figure 1: The storage capacity of high dimensional spaces. (a) A six from the USPS digit data set. (b) A sample from a simple independent pixel model of the six. There are $2^{3,648}$ possible images. Even with an enormous number of samples from such a model we would never see the original six. (c) A data set generated by rotating the original six from (a) 360 times. The data is projected onto its first two principal components. These two principal components show us that the data lives on a circle in this high dimensional space. There is a small amount of noise due to interpolation used in the image rotation. Alongside the projected points we show some examples of the rotated sixes.

data being high dimensional, the underlying structure is low dimensional. The objective of dimensionality reduction is to recover this underlying structure.

Given a data set with n data points and p features associated with each data point, dimensionality reduction involves representing the data set using n points each with a reduced number, q, of features, with $q < p$. Dimensionality reduction is a popular approach to dealing with high dimensional data: the hope is that while many data sets seem high dimensional, it may be that their intrinsic dimensionality is low like the rotated six above.

1.1 Spectral Dimensionality Reduction

Spectral approaches to dimensionality reduction involve taking a data set containing n points and forming a matrix of size $n \times n$ from which eigenvectors are extracted to give a representation of the data in a low dimensional space. Several spectral methods have become popular in the machine learning community including isomap (Tenenbaum et al., 2000), locally linear embeddings (LLE, Roweis and Saul, 2000), Laplacian eigenmaps (Belkin and Niyogi, 2003) and maximum variance unfolding (MVU, Weinberger et al., 2004). These approaches (and kernel principal component analysis, kernel PCA, Schölkopf et al., 1998) are closely related. For a kernel perspective on the relationships see Ham et al. (2004) and Bengio et al. (2004b,a). Our focus in this work is unifying the methods from a classical multidimensional scaling (CMDS, Mardia et al., 1979) perspective.

In classical multidimensional scaling an $n \times n$ symmetric distance matrix, whose elements contain the distance between two data points, is converted to a similarity matrix and visualized through its principal eigenvectors. Viewed from the perspective of CMDS the main difference between the spectral approaches developed in the machine learning community is in the distance matrices they (perhaps implicitly) proscribe.

In this paper we introduce a probabilistic approach to constructing the distance matrix: maximum entropy unfolding (MEU). We describe how isomap, LLE, Laplacian eigenmaps and MVU are related to MEU using the unifying perspective of Gaussian random fields and CMDS.

The parameters of the model are fitted through maximum likelihood in a Gaussian Markov random field (GRF). The random field specifies dependencies between *data points* rather than the more typical approach which specifies dependencies between *data features*. We show that the locally linear embedding algorithm is an approximation to maximum entropy unfolding where pseudolikelihood is maximized as an approximation to the model likelihood. Our probabilistic perspective inspires new dimensionality reduction algorithms. We introduce an exact version of locally linear embedding based on an acyclic graph structure that maximizes the true model likelihood (acyclic locally linear embedding, ALLE). We also consider approaches to learning the structure of the GRF through graphical regression (Friedman et al., 2008). By L1 regularization of the dependencies we explore whether learning the graph structure (rather than prespecifying by nearest neighbour) improves performance. We call the algorithm Dimensionality reduction through Regularization of the Inverse covariance in the Log Likelihood (DRILL).

Our methods are based on maximum likelihood. Normally maximum likelihood algorithms specify a distribution which factorizes over the data points (each data point is independent given the model parameters). In our models the likelihood factorizes over the features (each feature from the data set is independent given the model parameters). This means that maximum likelihood in our model is consistent as the number of features increases, $p \to \infty$ rather than the number of data points. Alternatively, the parameters of our models become better determined as the number of features increase, rather than the number of data. This can be interpreted as a *blessing* of dimensionality rather than the more usual 'curse of dimensionality.' This has significant implications for learning in high dimensional data (known as the large p small n regime) which run counter to received wisdom.

In Section 2 we derive our model through using standard assumptions from the field of dimensionality reduction and the maximum entropy principle (Jaynes, 1986). We then relate the model to other popular spectral approaches for dimensionality reduction and show how the parameters of the model can be fitted through maximum likelihood. This allows us to regularize the system with sparse priors and seek MAP solutions that restrict the inter point dependencies. Finally, we demonstrate the model (with comparisons) on two real world data sets. First though, we will review classical multidimensional scaling which provides the general framework through which these approaches can be related (see also Ham et al., 2004; Bengio et al., 2004b,a).

1.2 Classical Multidimensional Scaling

Given an $n \times n$ matrix of similarities, $\mathbf{K}$, or dissimilarities, $\mathbf{D}$, between a set of data points, multidimensional scaling considers the problem of how to represent these data in a low dimensional space. One way of doing this is to associate a q dimensional latent vector with each data point, $\mathbf{y}_{i,:}$, and define a set of dissimilarities between each latent point, $\delta_{i,j} = \left\|\mathbf{x}_{i,:} - \mathbf{x}_{j,:}\right\|_2^2$ (where $\|\cdot\|_2$ represents the L2-norm) to give a matrix $\mathbf{\Delta}$. Here we have specified the squared distance between each point as the dissimilarity.[1]

If the error for the latent representation is then taken to be the sum of absolute values between the dissimilarity matrix entries,

$$E(\mathbf{X}) = \sum_{i=1}^{n}\sum_{j=1}^{i-1}\left\|d_{i,j} - \delta_{i,j}\right\|_1, \tag{1}$$

and we assume that the data dissimilarities also represent a squared Euclidean distance matrix (perhaps computed in some high, maybe infinite, dimensional space) then the optimal *linear* dimensionality reduction is given by the following procedure (Mardia et al., 1979, pg. 400),

1. Convert the matrix of dissimilarities to a matrix of similarities by taking $\mathbf{B} = -\frac{1}{2}\mathbf{HDH}$ where $\mathbf{H} = \mathbf{I} - n^{-1}\mathbf{1}\mathbf{1}^\top$ is a centering matrix.

2. Extract the first q principal eigenvectors of $\mathbf{B}$.

1. It is more usual to specify the distance directly as the dissimilarity, however, for our purposes it will be more convenient to work with squared distances.

3. Setting $\mathbf{X}$ to these principal eigenvectors (appropriately scaled) gives a global minimum for the error function (1).

The centering matrix $\mathbf{H}$ is so called because when applied to data in the form of a design matrix, $\mathbf{Y} \in \Re^{n\times p}$, that is, one where each row is a data point and each column is a data set feature, the centred data matrix is recovered,

$$\begin{aligned}\hat{\mathbf{Y}} =& \mathbf{YH}\\ =& \mathbf{Y} - n^{-1}\mathbf{Y}\mathbf{1}\mathbf{1}^\top,\\ =& \mathbf{Y} - \boldsymbol{\mu}\mathbf{1}^\top\end{aligned}$$

where $\boldsymbol{\mu} = n^{-1}\mathbf{Y}\mathbf{1}$ is the empirical mean of the data set.

2. Maximum Entropy Unfolding

Classical multidimensional scaling provides the optimal *linear* transformation of the space in which the squared distances are expressed. The key contribution of recently developed spectral approaches in machine learning is to compute these distances in a space which is nonlinearly related to the data thereby ensuring a *nonlinear* dimensionality reduction algorithm. From a machine learning perspective this is perhaps clearest for kernel PCA (Schölkopf et al., 1998). In kernel PCA the squared distances are computed between points in a Hilbert space and related to the original data through a kernel function,

$$d_{i,j} = k(\mathbf{y}_{i,:}, \mathbf{y}_{i,:}) - 2k(\mathbf{y}_{i,:}, \mathbf{y}_{j,:}) + k(\mathbf{y}_{j,:}, \mathbf{y}_{j,:}). \tag{2}$$

For the linear kernel function, $k(\mathbf{y}_{i,:}, \mathbf{y}_{j,:}) = \mathbf{y}_{i,:}^\top \mathbf{y}_{j,:}$ this reduces to the squared Euclidean distance, but for nonlinear kernel functions such as $k(\mathbf{y}_{i,:}, \mathbf{y}_{j,:}) = \exp(-\gamma \left\|\mathbf{y}_{i,:} - \mathbf{y}_{j,:}\right\|_2^2)$ the distances are nonlinearly related to the data space. They are recognized as squared distances which are computed in a "feature space" (see, e.g., Ham et al., 2004; Bengio et al., 2004b,a). If we equate the kernel matrix, $\mathbf{K}$, to the similarity matrix in CMDS then this equation is also known as the *standard transformation* between a similarity and distance (Mardia et al., 1979).

Kernel PCA (KPCA) recovers an $\mathbf{x}_{i,:}$ for each data point and a mapping from the data space to the $\mathbf{X}$ space. Under the CMDS procedure we outlined above the eigenvalue problem is performed on the centered kernel matrix,

$$\mathbf{B} = \mathbf{HKH},$$

where $\mathbf{K} = [k(\mathbf{y}_{i,:}, \mathbf{y}_{j,:})]_{i,j}$. This matches the procedure for the KPCA algorithm (Schölkopf et al., 1998).[2] However, for the commonly used exponentiated quadratic kernel,

$$k(y_{i,:}, y_{j,:}) = \exp(-\gamma \left\|\mathbf{y}_{i,:} - \mathbf{y}_{j,:}\right\|_2^2),$$

KPCA actually *expands* the feature space rather than reducing the dimension (see Weinberger et al., 2004, for some examples of this). Unless data points are repeated the exponentiated quadratic kernel always leads to a full rank matrix, $\mathbf{K}$, and correspondingly a rank $n-1$ centred kernel matrix, $\mathbf{B}$. To exactly reconstruct the squared distances computed in feature space all but one of the eigenvectors of $\mathbf{B}$ need to be retained for our latent representation, $\mathbf{X}$. If the dimensionality of the data, p, is smaller than the number of data points, n, then we have a latent representation for our data which has higher dimensionality than the original data.

The observation that KPCA does not reduce the data dimensionality motivated the maximum variance unfolding algorithm (MVU, Weinberger et al., 2004). The idea in MVU is to learn a kernel matrix that will allow for dimensionality reduction. This is achieved by only considering *local relationships* in the data. A set

2. For stationary kernels, kernel PCA also has an interpretation as a particular form of *metric* multidimensional scaling, see Williams (2001) for details.

of neighbors is defined (e.g., by k-nearest neighbors) and only distances between neighboring data points are respected. These distances are specified as constraints, and the other elements of the kernel matrix are filled in by maximizing its trace, $\mathrm{tr}(\mathbf{K})$, that is, the *total variance* of the data in feature space, while respecting the distance constraints and keeping the resulting matrix centered. Maximizing $\mathrm{tr}(\mathbf{K})$ maximizes the interpoint squared distances for all points that are unconnected in the neighborhood graph, thereby unravelling the manifold.

In this paper we consider an alternative maximum entropy formalism of this problem. Since entropy is related to variance, we might expect a similar result in the quality of the resulting algorithm, but since maximum entropy also provides a probability distribution we should also obtain a probabilistic model with all the associated advantages (dealing with missing data, extensions to mixture models, fitting parameters by Bayesian methods, combining with other probabilistic models). Importantly, our interpretation will also enable us to relate our algorithm to other well known spectral techniques as they each turn out to approximate maximum entropy unfolding in some way.

2.1 Constraints from D Lead to a Density on Y

The maximum entropy formalism (see, e.g., Jaynes, 1986) allows us to derive a probability density given only a set of constraints on expectations under that density. These constraints may be derived from observation. In our case the observations will be squared distances between data points, but we will derive a density over $\mathbf{Y}$ directly (not over the squared distances). We will do this by looking to constrain the expected squared inter-point distances, $d_{i,j}$, of any two samples, $\mathbf{y}_{i,:}$ and $\mathbf{y}_{j,:}$, from the density. This means that while our observations may be only of the squared distances, $d_{i,j}$, the corresponding density will be over the data space that gives rise to those distances, $p(\mathbf{Y})$. Of course, once we have found the form of probability density we are free to directly model in the space $\mathbf{Y}$ or make use only of the squared distance constraints. Direct modeling in $\mathbf{Y}$ turns out to be equivalent to maximum likelihood. However, since we do not construct a density over the squared distance matrix, modeling based on that information alone should be thought of as maximum entropy under distance constraints rather than maximum likelihood.

In the maximum entropy formalism, we specify the density by a free form maximization of the entropy subject to the imposed expectation constraints. The constraints we use will correspond to the constraints applied to maximum variance unfolding: the expectations of the squared distances between two neighboring data points sampled from the model.

2.2 Maximum Entropy in Continuous Systems

The entropy of a continuous density is normally defined as the limit of a discrete system. The continuous distribution is discretized and we consider the limit as the discrete bin widths approach zero. However, as that limit is taken the term dependent on the bin width approaches ∞. Normally this is dealt with by ignoring that term and referring to the remaining term as differential entropy. However, the maximum entropy solution for this differential entropy turns out to be undefined. Jaynes (1986) proposes an alternative *invariant measure* to the entropy. For maximum entropy in continuous systems we maximize the negative Kullback Leibler divergence (KL divergence, Kullback and Leibler, 1951) between a base density, $m(\mathbf{Y})$, and the density of interest, $p(\mathbf{Y})$,

$$H = -\int p(\mathbf{Y}) \log \frac{p(\mathbf{Y})}{m(\mathbf{Y})} \mathrm{d}\mathbf{Y}.$$

Maximizing this measure is equivalent to minimizing the KL divergence between $p(\mathbf{Y})$ and the base density, $m(\mathbf{Y})$. Any choice of base density can be made, but the solution will be pulled towards the base density (through the minimization of the KL divergence). We choose a base density to be a very broad, spherical, Gaussian density with covariance $\gamma^{-1}\mathbf{I}$. This adds a new parameter, γ, to the system, but it will turn out that this parameter has little affect on our analysis. Typically it can be taken to zero or assumed small. The density

that minimizes the KL divergence under the constraints on the expectations is then

$$p(\mathbf{Y}) \propto \exp\left(-\frac{1}{2}\text{tr}\left(\gamma\mathbf{Y}\mathbf{Y}^\top\right)\right)\exp\left(-\frac{1}{2}\sum_i \sum_{j\in\mathcal{N}(i)} \lambda_{i,j} d_{i,j}\right),$$

where $\mathcal{N}(i)$ represents the set of neighbors of data point i, and $\mathbf{Y} = [\mathbf{y}_{1,:}, \ldots, \mathbf{y}_{n,:}]^\top \in \Re^{n\times p}$ is a *design matrix* containing our data. Note that we have introduced a factor of $-1/2$ in front of our Lagrange multipliers,[3] $\{\lambda_{i,j}\}$, for later notational convenience. We now define the matrix $\mathbf{\Lambda}$ to contain $\lambda_{i,j}$ if i is a neighbor of j and zero otherwise. This allows us to write the distribution[4] as

$$p(\mathbf{Y}) \propto \exp\left(-\frac{1}{2}\text{tr}\left(\gamma\mathbf{Y}\mathbf{Y}^\top\right) - \frac{1}{4}\text{tr}\left(\mathbf{\Lambda}\mathbf{D}\right)\right).$$

We now introduce a matrix $\mathbf{L}$, which has the form of a graph Laplacian. It is symmetric and constrained to have a null space in the constant vector, $\mathbf{L}\mathbf{1} = \mathbf{0}$. Its off diagonal elements are given by $-\mathbf{\Lambda}$ and its diagonal elements are given by

$$\ell_{i,i} = \sum_{j\in\mathcal{N}(i)} \lambda_{i,j}$$

to enforce the null space constraint. The null space constraint enables us to write

$$p(\mathbf{Y}) = \frac{|\mathbf{L}+\gamma\mathbf{I}|^{\frac{1}{2}}}{\tau^{\frac{np}{2}}}\exp\left(-\frac{1}{2}\text{tr}\left((\mathbf{L}+\gamma\mathbf{I})\mathbf{Y}\mathbf{Y}^\top\right)\right), \tag{3}$$

where for convenience we have defined $\tau = 2\pi$. We arrive here because the distance matrix is zero along the diagonal. This allows us to set the diagonal elements of $\mathbf{L}$ as we please without changing the value of $\text{tr}(\mathbf{L}\mathbf{D})$. Our choice to set them as the sum of the off diagonals gives the matrix a null space in the constant vector enabling us to use the fact that

$$\mathbf{D} = \mathbf{1}\text{diag}\left(\mathbf{Y}\mathbf{Y}^\top\right)^\top - 2\mathbf{Y}\mathbf{Y}^\top + \text{diag}\left(\mathbf{Y}\mathbf{Y}^\top\right)\mathbf{1}^\top$$

(where the operator $\text{diag}(\mathbf{A})$ forms a vector from the diagonal of $\mathbf{A}$) to write

$$-\text{tr}(\mathbf{\Lambda}\mathbf{D}) = \text{tr}(\mathbf{L}\mathbf{D}) = \text{tr}\left(\mathbf{L}\mathbf{1}\text{diag}\left(\mathbf{Y}\mathbf{Y}^\top\right)^\top - 2\mathbf{L}\mathbf{Y}\mathbf{Y}^\top + \text{diag}\left(\mathbf{Y}\mathbf{Y}^\top\right)\mathbf{1}^\top\mathbf{L}\right) = -2\text{tr}\left(\mathbf{L}\mathbf{Y}\mathbf{Y}^\top\right),$$

which in turn allows us to recover (3). This probability distribution is a *Gaussian random field*. It can also be written as

$$p(\mathbf{Y}) = \prod_{j=1}^{p}\frac{|\mathbf{L}+\gamma\mathbf{I}|^{\frac{1}{2}}}{\tau^{\frac{n}{2}}}\exp\left(-\frac{1}{2}\mathbf{y}_{:,j}^\top(\mathbf{L}+\gamma\mathbf{I})\mathbf{y}_{:,j}\right),$$

which emphasizes the independence of the density across data features.

2.3 Gaussian Markov Random Fields

Multivariate Gaussian densities are specified by their mean, $\boldsymbol{\mu}$, and a covariance matrix, $\mathbf{C}$. A standard modeling assumption is that data is drawn independently from identical Gaussian densities. For this case the likelihood of the data, $p(\mathbf{Y})$, will be factorized across the individual data points,

$$p(\mathbf{Y}) = \prod_{i=1}^{n} p(\mathbf{y}_{i,:}) = \prod_{i=1}^{n}\mathcal{N}(\mathbf{y}_{i,:}|\boldsymbol{\mu},\mathbf{C})$$

3. We use λ for both Lagrange multipliers and eigenvalues, we hope that the meaning is clear from the context of use.
4. In our matrix notation the Lagrange multipliers and distances are appearing twice inside the trace, in matrices that are constrained symmetric, $\mathbf{\Lambda}$ and $\mathbf{D}$. The factor of $\frac{1}{4}$ replaces the factor of $\frac{1}{2}$ in the previous equation to account for this "double counting."

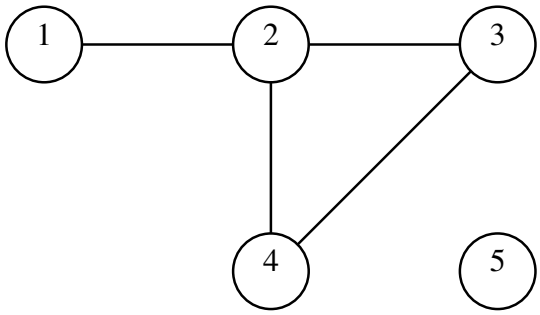


Figure 2: Graph representing conditional relationships between $p = 5$ features from a Gaussian Markov random field. Here the 5th feature is independent of the others. Feature 1 is conditionally dependent on 2, feature 2 is conditional dependent on 1, 3 and 4. Feature 3 is conditionally dependent on 2 and 4, and feature 4 is conditionally dependent on 2 and 3.

and the mean and covariance of the Gaussian are estimated by maximizing the log likelihood of the data. The covariance matrix is symmetric and positive definite. It contains $\frac{p(p+1)}{2}$ parameters. However, if the number of data points, n, is small relative to the number of features p, then the parameters may not be well determined. For this reason we might seek a representation of the covariance matrix which has fewer parameters. One option is a low rank representation,

$$\mathbf{C} = \mathbf{W}\mathbf{W}^\top + \mathbf{D},$$

where $\mathbf{D}$ is a diagonal matrix and $\mathbf{W} \in \Re^{p\times q}$. This is the representation underlying factor analysis, and if $\mathbf{D} = \sigma^2\mathbf{I}$, probabilistic principal component analysis (PPCA, Tipping and Bishop, 1999). For PPCA there are $pq+1$ parameters in the covariance representation.

An alternative approach, and one that is particularly popular in spatial systems, is to assume a sparse *inverse* covariance matrix, known as the precision matrix, or information matrix. In this representation we consider each feature to be a vertex in a graph. If two vertices are unconnected they are conditionally independent in the graph. In Figure 2 we show a simple example graph where the precision matrix is

$$\mathbf{K}^{-1} = \mathbf{J} = \begin{bmatrix} j_{1,1} & j_{1,2} & 0 & 0 & 0 \\ j_{2,1} & j_{2,2} & j_{2,3} & j_{2,4} & 0 \\ 0 & j_{3,2} & j_{3,3} & j_{3,4} & 0 \\ 0 & j_{4,2} & j_{4,3} & j_{4,4} & 0 \\ 0 & 0 & 0 & 0 & j_{5,5} \end{bmatrix}.$$

Zeros correspond to locations where there are no edges between vertices in the graph.

If each feature is constrained to only have K neighbors in the graph, then the inverse covariance (and correspondingly the covariance) is only parameterized by $Kp+p$ parameters. So the GRF provides an alternative approach to reducing the number of parameters in the covariance matrix.

2.4 Independence Over Data Features

The Gaussian Markov random field (GRF) for maximum entropy unfolding is unusual in that the independence is being expressed over data features (in the p-dimensional direction) instead of over data points (in the n-dimensional direction). This means that our model assumes that data *features* are independently and identically distributed (i.i.d.) given the model parameters. The standard assumption for Gaussian models is that data *points* we are expressing conditional probability densities between data points are i.i.d. given the parameters. This specification cannot be thought of as "the wrong way around" as it is merely a consequence

of the constraints we chose to impose on the maximum entropy solution. If those constraints are credible, then this model is also credible. This is not the first model proposed for which the independence assumptions are reversed. Such models have been proposed formerly in the context of semi-supervised learning (Zhu et al., 2003), probabilistic nonlinear dimensionality reduction (Lawrence, 2004, 2005) and in models that aim to discover structural form from data (Kemp and Tenenbaum, 2008).

2.5 Maximum Likelihood and Blessing of Dimensionality

Once the form of a maximum entropy density is determined, finding the Lagrange multipliers in the model is equivalent to maximizing the likelihood of the model, where the Lagrange multipliers are now considered to be parameters. The theory underpinning maximum likelihood is broad and well understood, but much of it relies on assuming independence across data points rather than data features. For example, maximum likelihood with independence across data points can be shown to be consistent by viewing the objective as a sample based approximation to the Kullback-Leibler (KL) divergence between the true data generating density, $\tilde{p}(\mathbf{y})$, and our approximation $p(\mathbf{y}|\boldsymbol{\theta})$ which in turn depends on parameters, $\boldsymbol{\theta}$. Taking the expectations under the generating density this KL divergence is written as

$$\mathrm{KL}\left(\tilde{p}(\mathbf{y}) \,\|\, p(\mathbf{y}|\boldsymbol{\theta})\right) = \int \tilde{p}(\mathbf{y}) \log \tilde{p}(\mathbf{y}) \mathrm{d}\mathbf{y} - \int \tilde{p}(\mathbf{y}) \log p(\mathbf{y}|\boldsymbol{\theta}) \mathrm{d}\mathbf{y}.$$

Given n sampled data points from $\tilde{p}(\mathbf{y})$, $\{\mathbf{y}_{i,:}\}$ we can write down a sample based approximation to the KL divergence in the form

$$\mathrm{KL}\left(\tilde{p}(\mathbf{y}) \,\|\, p(\mathbf{y})\right) \approx -\frac{1}{n} \sum_{i=1}^{n} \log p(\mathbf{y}_{i,:}|\boldsymbol{\theta}) + \text{const.},$$

where the constant term derives from the entropy of the generating density, which whilst unknown, does not depend on our model parameters. Since the sample based approximation is known to become exact in the large sample limit, and the KL divergence has a global minimum of zero *only* when the generating density and our approximation are identical, we know that, *if* the generating density falls within our chosen class of densities, maximum likelihood will reveal it in the large data limit. The global maximum of the likelihood will correspond to a global minimum of the KL divergence. Further, we can show that as we approach this limit, *if* the total number of parameters is fixed, our parameter values, $\boldsymbol{\theta}$, will become better determined (see, e.g., Wasserman, 2003, pg. 126). Since the number of parameters is often related to data dimensionality, p, this implies that for a given data dimensionality, p, we require a large number of data points, n, to have confidence we are approaching the large sample limit and our model's parameters will be well determined. We refer to this model set up as the *sampled-points* formalism.

The scenario described above does not apply for the situation where we have independence across data features. In this situation we construct an alternative consistency argument, but it is based around a density which describes correlation between data points instead of data features. This model is independent across data features. Models of this type can occur quite naturally. Consider the following illustrative example from cognitive science (Kemp and Tenenbaum, 2008). We wish to understand the relationship between different animals as more information about those animals' features is uncovered. There are 33 species in the group, and information is gained by unveiling features of the animals. A model which assumes independence over animals would struggle to incorporate additional feature information (such as whether or not the animal has feet, or whether or not it lives in the ocean). A model which assumes independence across features handles this situation naturally. However, to show the consistency of the model we must now think of our model as a generative model for data features, $\tilde{p}(\mathbf{y}')$, rather than data points. Our approximation to the KL divergence still applies,

$$\mathrm{KL}\left(\tilde{p}(\mathbf{y}') \,\|\, p(\mathbf{y}')\right) \approx -\frac{1}{p} \sum_{j=1}^{p} \log p(\mathbf{y}_{:,i}|\boldsymbol{\theta}) + \text{const.},$$

but now the sample based approximation is based on independent samples of features (in the animal example, whether or not it has a beak, or whether the animal can fly), instead of samples of data points. This model

will typically have a parameter vector that increases in size with the data set size, n (in that sense it is non-parametric), rather than the data dimensionality, p. The model is consistent as the number of features becomes large, rather than data points. For our Gaussian random field, the number of parameters increases linearly with the number of data points, but does not increase with the number of data (each datum requires $O(K)$ parameters to connect with K neighbors). However, as we increase features there is no corresponding increase in parameters. In other words as the number of features increases there is a clear *blessing of dimensionality*. We refer to this model set up as the *sampled-features* formalism.

There is perhaps a deeper lesson here in terms of how we should interpret such consistency results. In the sampled-points formalism, as we increase the number of data points, the parameters become better determined. In the sampled-features formalism, as we increase the number of features, the parameters become better determined. However, for consistency results to hold, the class of models we consider must include the actual model that generated the data. If we believe that "Essentially, all models are wrong, but some are useful" (Box and Draper, 1987, pg. 424) we may feel that encapsulating the right model within our class is a practical impossibility. Given that, we might pragmatically bias our choice somewhat to ensure utility of the resulting model. From this perspective, in the large p small n domain, the sampled-features formalism is attractive. A practical issue can arise though. If we wish to compute the likelihood of an out of sample data-point, we must first estimate the parameters associated with that new data point. This can be problematic. Of course, for the sampled-points formalism the same problem exists when you wish to include an out of sample data-feature in your model (such as in the animals example in Kemp and Tenenbaum, 2008). Unsurprisingly, addressing this issue for spectral methods is nontrivial (Bengio et al., 2004b).

2.5.1 PARAMETER GRADIENTS

We can find the parameters, $\mathbf{\Lambda}$, through maximum likelihood on the Gaussian Markov random field given in (3). Some algebra shows that the gradient of each Lagrange multiplier is given by,

$$\frac{\mathrm{d}\log p(\mathbf{Y})}{\mathrm{d}\lambda_{i,j}} = \frac{1}{2}\left\langle d_{i,j}\right\rangle_{p(\mathbf{Y})} - \frac{1}{2}d_{i,j},$$

where $\left\langle\right\rangle_{p(\cdot)}$ represents an expectation under the distribution $p(\cdot)$. This result is a consequence of the maximum entropy formulation: the Lagrange multipliers have a gradient of zero when the constraints are satisfied. To compute gradients we need the expectation of the squared distance given by

$$\left\langle d_{i,j}\right\rangle = \left\langle y_{i,:}^\top y_{i,:}\right\rangle - 2\left\langle y_{i,:}^\top y_{j,:}\right\rangle + \left\langle y_{j,:}^\top y_{j,:}\right\rangle,$$

which we can compute directly from the covariance matrix of the GRF, $\mathbf{K} = (\mathbf{L}+\gamma\mathbf{I})^{-1}$,

$$\left\langle d_{i,j}\right\rangle = \frac{p}{2}\left(k_{i,i} - 2k_{i,j} + k_{j,j}\right).$$

This is immediately recognized as a scaled version of the standard transformation between distances and similarities (see (2)). This relationship arises naturally in the probablistic model. Every GRF has an associated interpoint distance matrix. It is this matrix that is being used in CMDS. The machine learning community might interpret this as the relationship between distances in "feature space" and the kernel function. Note though that here (and also in MVU) each individual element of the kernel matrix *cannot* be represented only as a function of the corresponding two data points (i.e., we cannot represent them as $k_{i,j} = k(\mathbf{y}_{i,:}, \mathbf{y}_{j,:})$, where each $k_{i,j}$ is a function *only* of the i and jth data points). Given this we feel it is more correct to think of this matrix as a covariance matrix induced by our specification of the random field rather than a true Mercer kernel. We use the notation $k_{i,j}$ to denote an element of such a covariance (or similarity matrix) and only use $k(\cdot,\cdot)$ notation when the value of the similarity matrix can be explicitly represented as a Mercer kernel.

The Base Density Parameter. One role of the base density parameter, γ, is to ensure that the precision matrix is positive definite. Recall that the Laplacian has a null space in the constant vector, implying that $\mathbf{K1} = \gamma^{-1}$,

which becomes infinite as $\gamma \to 0$. This reflects an insensitivity of the covariance matrix to the data mean, and this in turn arises because that information is lost when we specify the expectation constraints only through interpoint distances. In practise though, $\mathbf{K}$ is always centred before its eigenvectors are extracted, $\mathbf{B} = \mathbf{HKH}$, resulting in $\mathbf{B1} = \mathbf{0}$ so γ has no effect on the final visualization. In some cases, it may be necessary to set γ to a small non-zero value to ensure stability of the inverse $\mathbf{L} + \gamma\mathbf{I}$. In these cases we set it to $\gamma = 1 \times 10^{-4}$ but in many of the comparisons we make to other spectral algorithms below we take it to be zero.

Number of Model Parameters. If K neighbors are used for each data point there are $O(Kn)$ parameters in the model, so the model is nonparametric in the sense that the number of parameters increases with the number of data. For the parameters to be well determined we require a large number of features, p, for each data point, otherwise we would need to regularize the model (see Section 3). This implies that the model is well primed for the so-called "large p small n domain."

Once the maximum likelihood solution is recovered the data can be visualized, as for MVU and kernel PCA, by looking at the eigenvectors of the centered covariance matrix $\mathbf{HKH}$. We call this algorithm maximum entropy unfolding (MEU).

Positive Definite Constraints. The maximum variance unfolding (MVU) algorithm maximizes the trace of the covariance matrix (given by the sum of its eigenvalues, $\{\lambda_i\}$),

$$\mathrm{tr}(\mathbf{K}) = \sum_{i=1}^{n} \lambda_i,$$

subject to constraints on the elements of $\mathbf{K}$ arising from the squared distances. These constraints are linear in the elements of $\mathbf{K}$. There is a further constraint on $\mathbf{K}$, that it should be positive semi-definite. This means MVU can be optimized through a a semi-definite program. In contrast MEU cannot be optimized through a semi-definite program because the objective is not linear in $\mathbf{K}$. This implies we need to find other approaches to maintaining the positive-definite constraint on $\mathbf{K}$. Possibilities include exploiting the fact that if the Lagrange multipliers are constrained to be positive the system is "attractive" and this guarantees a valid covariance (see, e.g., Koller and Friedman, 2009, pg. 255). Although now (as in a suggested variant of the MVU) the distance constraints would be inequalities. Another alternative would be to constrain $\mathbf{L}$ to be diagonally dominant through adjusting γ. We will also consider two further approaches in Section 2.7 and Section 3.

Non-linear Generalizations of PCA. Kernel PCA provides a non-linear generalization of PCA. This is achieved by 'kernelizing' the principal coordinate analysis algorithm: replacing data point inner products with a kernel function. Maximum variance unfolding and maximum entropy unfolding also provide non linear generalizations of PCA. For these algorithms, if we increase the neighborhood size to $K = n - 1$, then all squared distances implied by the GRF model are constrained to match the observed inter data point squared distances and $\mathbf{L}$ becomes non-sparse. Classical multidimensional scaling on the resulting squared distance matrix is known as principal coordinate analysis and is equivalent to principal component analysis (see Mardia et al., 1979).[5]

2.6 Relation to Laplacian Eigenmaps

Laplacian eigenmaps is a spectral algorithm introduced by Belkin and Niyogi (2003). In the Laplacian eigenmap procedure, a neighborhood is first defined in the data space. Typically this is done through nearest neighbor algorithms or defining all points within distance ε of each point to be neighbors. In Laplacian eigenmaps a symmetric sparse (possibly weighted) adjacency matrix, $\mathbf{A} \in \Re^{n \times n}$, is defined whose i, jth element, $a_{i,j}$ is non-zero if the ith and jth data points are neighbors. Belkin and Niyogi argue that a good *one*

5. In this case CMDS proceeds by computing the eigendecomposition of the centred negative squared distance matrix, which is the eigendecomposition of the centred inner product matrix as is performed for principal coordinate analysis.

dimensional embedding is one where the latent points, $\mathbf{X}$ minimize

$$E(\mathbf{X}) = \frac{1}{4}\sum_{i=1}^{n}\sum_{j=1}^{n} a_{i,j}(x_i - x_j)^2.$$

For a multidimensional embedding we can rewrite this objective in terms of the squared distance between two latent points, $\delta_{i,j} = \left\|\mathbf{x}_{i,:} - \mathbf{x}_{j,:}\right\|_2^2$, as

$$E(\mathbf{X}) = \frac{1}{4}\sum_{i=1}^{n}\sum_{j=1}^{n} a_{i,j}\delta_{i,j}.$$

The motivation behind this objective function is that neighboring points have non-zero entries in the adjacency matrix, therefore their inter point squared distances in latent space need to be minimized. In other words points which are neighbors in data space will be kept close together in the latent space. The objective function can be rewritten in matrix form as

$$E(\mathbf{X}) = \frac{1}{4}\mathrm{tr}\left(\mathbf{A}\boldsymbol{\Delta}\right).$$

Squared Euclidean distance matrices of this type can be rewritten in terms of the original vector space by introducing the Laplacian matrix. Introducing the degree matrix, $\mathbf{D}$, which is diagonal with entries, $d_{i,i} = \sum_j \mathbf{A}_{i,j}$ the Laplacian associated with the neighborhood graph can be written

$$\mathbf{L} = \mathbf{D} - \mathbf{A}$$

and the error function can now be written directly in terms of the latent coordinates,

$$E(\mathbf{X}) = \frac{1}{2}\mathrm{tr}\left(\mathbf{L}\mathbf{X}\mathbf{X}^\top\right)$$

by exploiting the null space of the Laplacian ($\mathbf{L1} = \mathbf{0}$) as we saw in Section 2.1.

Let us consider the properties of this objective. Since the error function is in terms of interpoint distances, it is insensitive to translations of the embeddings. The mean of the latent projections is therefore undefined. Further, there is a trivial solution for this objective. If the latent points are all placed on top of one another the interpoint distance matrix will be all zeros. To prevent this collapse Belkin and Niyogi suggest that each dimension of the latent representation is constrained,

$$\mathbf{x}_{:,i}^\top \mathbf{D}\mathbf{x}_{:,i} = 1.$$

Here the degree matrix, $\mathbf{D}$, acts to scale each data point so that points associated with a larger neighborhood are pulled towards the origin.

Given this constraint the objective function is minimized for a q dimensional space by the generalized eigenvalue problem,

$$\mathbf{L}\mathbf{u}_i = \lambda_i \mathbf{D}\mathbf{u}_i,$$

where λ_i is an eigenvalue and $\mathbf{u}_i$ is its associated eigenvector. The smallest eigenvalue is zero and is associated with the constant eigenvector. This eigenvector is discarded, whereas the eigenvectors associated with the next q smallest eigenvalues are retained for the embedding. So we have,

$$\mathbf{x}_{:,i} = \mathbf{u}_{i+1} \quad \text{for} \quad i = 1..q$$

if we assume that eigenvalues are ordered according to magnitude with the smallest first.

Note that the generalized eigenvalue problem underlying Laplacian eigenmaps can be readily converted to the related, symmetric, eigenvalue problem.

$$\hat{\mathbf{L}}\mathbf{v}_i = \lambda_i \mathbf{v}_i \tag{4}$$

where $\hat{\mathbf{L}}$ is the *normalized Laplacian matrix*,

$$\hat{\mathbf{L}} = \mathbf{D}^{-\frac{1}{2}}\mathbf{L}\mathbf{D}^{-\frac{1}{2}} = \mathbf{I} - \mathbf{D}^{-\frac{1}{2}}\mathbf{A}\mathbf{D}^{-\frac{1}{2}}$$

and the relationship between the eigenvectors is through scaling by the degree matrix, $\mathbf{v}_i = \mathbf{D}^{\frac{1}{2}}\mathbf{u}_i$ (implying $\mathbf{v}_i^\top \mathbf{v}_i = 1$). The eigenvalues remain unchanged in each case.

2.6.1 PARAMETERIZATION IN LAPLACIAN EIGENMAPS

In Laplacian eigenmaps the adjacency matrix can either be unweighted (Belkin and Niyogi refer to this as the simple-minded approach) or weighted according to the distance between two data points,

$$a_{i,j} = \exp\left(-\frac{\left\|\mathbf{y}_{i,:} - \mathbf{y}_{j,:}\right\|_2^2}{2\sigma^2}\right), \tag{5}$$

which is justified by analogy between the discrete graph Laplacian and its continuous equivalent, the Laplace Beltrami operator (Belkin and Niyogi, 2003).

2.6.2 RELATING LAPLACIAN EIGENMAPS TO MEU

The relationship of MEU to Laplacian eigenmaps is starting to become clear. In Laplacian eigenmaps a graph Laplacian is specified across the data points just as in maximum entropy unfolding. In classical multidimensional scaling, as applied in MEU and MVU, the eigenvectors associated with the largest eigenvalues of the centred covariance matrix,

$$\mathbf{B} = \mathbf{H}\left(\mathbf{L} + \gamma\mathbf{I}\right)^{-1}\mathbf{H} \tag{6}$$

are used for visualization. In Laplacian eigenmaps the smallest eigenvectors of $\mathbf{L}$ are used, disregarding the eigenvector associated with the null space.

Note that if we define the eigendecomposition of the covariance in the GRF as

$$\mathbf{K} = \mathbf{U}\mathbf{\Lambda}\mathbf{U}^\top$$

it is easy to show that the eigendecomposition of the associated Laplacian matrix is

$$\mathbf{L} = \mathbf{U}\left(\mathbf{\Lambda}^{-1} - \gamma\mathbf{I}\right)\mathbf{U}^\top.$$

We know that the smallest eigenvalue of $\mathbf{L}$ is zero with a constant eigenvector. That implies that the largest eigenvalue of $\mathbf{K}$ is γ^{-1} and is associated with a constant eigenvector. However, we do not use the eigenvectors of $\mathbf{K}$ directly. We first apply the centering operation in (6). This projects out the constant eigenvector, but leaves the remaining eigenvectors and eigenvalues intact.

To make the analogy with Laplacian eigenmaps direct we consider the formulation of its eigenvalue problem with the normalized graph Laplacian as given in (4). Substituting the normalized graph Laplacian into our covariance matrix, $\mathbf{K}$, we see that for Laplacian eigenmaps we are visualizing a Gaussian random field with a covariance as follows,

$$\mathbf{K} = (\hat{\mathbf{L}} + \gamma\mathbf{I})^{-1}.$$

Naturally we could also consider a variant of the algorithm which used the unnormalized Laplacian directly, $\mathbf{K} = (\mathbf{L} + \gamma\mathbf{I})^{-1}$. Under the Laplacian eigenmap formulation that would be equivalent to preventing the collapse of the latent points by constraining $\mathbf{x}_{:,i}^\top\mathbf{x}_{:,i} = 1$ instead of $\mathbf{x}_{:,i}^\top\mathbf{D}\mathbf{x}_{:,i} = 1$.

This shows the relationship between the eigenvalue problems for Laplacian eigenmaps and CMDS. The principal eigenvalues of $\mathbf{K}$ will be the smallest eigenvalues of $\mathbf{L}$. The very smallest eigenvalue of $\mathbf{L}$ is zero and associated with the constant eigenvector. However, in CMDS this would be removed by the centering operation and in Laplacian eigenmaps it is discarded. Once the parameters of the Laplacian have been set CMDS is being performed to recover the latent variables in Laplacian eigenmaps.

2.6.3 LAPLACIAN EIGENMAPS SUMMARY

The Laplacian eigenmaps procedure does not fit parameters through maximum likelihood. It uses analogies with the continuous Laplace Beltrami operator to set them via the Gaussian-like relationship in (5). This means that the local distance constraints are not a feature of Laplacian eigenmaps. The implied squared distance matrix used for CMDS will not preserve the interneighbor distances as it will for MVU and MEU. In

fact since the covariance matrix is never explicitly computed it is not possible to make specific statements about what these distances will be in general. However, Laplacian eigenmaps gains significant computational advantage by not representing the covariance matrix explicitly. No matrix inverses are required in the algorithm and the resulting eigenvalue problem is sparse. This means that Laplacian eigenmaps can be applied to much larger data sets than would be possible for MEU or MVU.

2.7 Relation of MEU to Locally Linear Embedding

The locally linear embedding (LLE Roweis and Saul, 2000) is a dimensionality reduction that was originally motivated by the idea that a non-linear manifold could be approximated by small linear patches. If the distance between data points is small relative to the curvature of the manifold at a particular point, then the manifold encircling a data point and its nearest neighbors may be approximated locally by a linear patch. This idea gave rise to the locally linear embedding algorithm. First define a local neighborhood for each data point and find a set of linear regression weights that allows each data point to be reconstructed by its neighbors. Considering the ith data point, $\mathbf{y}_{i,:}$ and a vector of reconstruction weights, $\mathbf{w}_{:,i}$, associated with that data point a standard least squares regression objective takes the form,

$$E(\mathbf{w}_{:,i}) = \frac{1}{2}\left\|\mathbf{y}_{i,:} - \sum_{j\in\mathcal{N}(i)} \mathbf{y}_{j,:} w_{j,i}\right\|_2^2, \tag{7}$$

for each data point. Here the sum over the reconstruction weights, $\mathbf{w}_{:,j}$ is restricted to data points, $\left\{\mathbf{y}_{j,:}\right\}_{j\in\mathcal{N}(()i)}$, which are in the neighborhood of the data point of interest, $\mathbf{y}_{i,:}$. Roweis and Saul point out that the objective function in (7) is invariant to rotation and rescaling of the data. If we rotate each data vector in (7) the objective does not change. If data are rescaled, for example, multiplied by a factor α, then the objective is simply rescaled by a factor α^2. However, the objective is not invariant to translation. For example if we were to translate the data, $\hat{\mathbf{y}}_{i,:} = \mathbf{y}_{i,:} - \boldsymbol{\mu}$, where $\boldsymbol{\mu}$ could be the sample mean of our data set (or any other translation), we obtain the following modified objective,

$$E(\mathbf{w}_{:,i}) = \frac{1}{2}\left\|\hat{\mathbf{y}}_{i,:} + \boldsymbol{\mu} - \sum_{j\in\mathcal{N}(i)} \hat{\mathbf{y}}_{j,:} w_{j,i} - \boldsymbol{\mu} \sum_{j\in\mathcal{N}(i)} w_{j,i}\right\|_2^2,$$

which retains a dependence on $\boldsymbol{\mu}$. Roweis and Saul point out that if we constrain $\sum_{j\in\mathcal{N}(i)} w_{j,i} = 1$ the terms involving $\boldsymbol{\mu}$ cancel and we recover the original objective. Imposing this constraint on the regression weights (which can also be written $\mathbf{w}_{:,i}^\top \mathbf{1} = 1$), ensures the objective is translation invariant.

To facilitate the comparison with the maximum entropy unfolding algorithm we now introduce an alternative approach to enforcing translation invariance. Our approach generalizes the LLE algorithm. First of all we introduce a new matrix $\mathbf{M}$. The sparsity pattern of this matrix should match that of $\mathbf{W}$ for off diagonal elements. We then set the diagonal elements of each row of $\mathbf{M}$ to be the negative sum of the off diagonal columns, so we have $m_{i,i} = -\sum_{j\in\mathcal{N}(i)} w_{j,i}$. We can then rewrite the objective in (7) as,

$$E(\mathbf{w}_{:,i}) = \frac{1}{2}\left\|\mathbf{Y}^\top \mathbf{m}_{:,i}\right\|_2^2 = \mathbf{m}_{:,i}^\top \mathbf{Y}\mathbf{Y}^\top \mathbf{m}_{:,i},$$

which is identical to (7) if $m_{i,i}$ is further constrained to 1. However, even if this constraint is not imposed, the translational invariance is retained. This is clear if we rewrite the objective in terms of the non-zero elements of $\mathbf{m}_{:,i}$,

$$\begin{aligned} E(\mathbf{m}_{:,i}) &= \frac{m_{i,i}^2}{2}\left\|\mathbf{y}_{i,:} + \sum_{j\in\mathcal{N}(i)} \mathbf{y}_{j,:}\frac{m_{j,i}}{m_{i,i}}\right\|_2^2 \\ &= \frac{m_{i,i}^2}{2}\left\|\mathbf{y}_{i,:} - \sum_{j\in\mathcal{N}(i)} \mathbf{y}_{j,:} w_{j,i}\right\|_2^2 \end{aligned}$$

where

$$w_{j,i} = -\frac{m_{j,i}}{m_{i,i}}$$

and by definition of $m_{i,i}$ we have $\sum_{j\in\mathcal{N}(i)} w_{j,i} = 1$. We now see that up to a scalar factor, $m_{i,i}^2$, this equation is identical to (7).

This form of the objective also shows us that $m_{i,i}$ has the role of scaling each data point's contribution to the overall objective function (rather like the degree, $d_{i,i}$ would do in the unnormalized variant of Laplacian eigenmaps we discussed in Section 2.6.2).

The objective function is a least squares formulation with particular constraints on the regression weights, $\mathbf{m}_{:,i}$. As with all least squares regressions, there is an underlying probabilistic interpretation of the regression which suggests Gaussian noise. In our objective function the variance of the Gaussian noise for the ith data point is given by $m_{i,i}^{-2}$. We can be a little more explicit about this by writing down the error as the negative log likelihood of the equivalent Gaussian model. This then includes a normalization term, $\log m_{i,i}^2$, which is zero in standard LLE where $m_{i,i}^2 = 1$,

$$\begin{aligned}
E(\mathbf{w}_{:,i}) &= -\log \mathcal{N}\left(\mathbf{y}_{i,:}|\sum_{j\in\mathcal{N}(i)} \mathbf{y}_{j,:}\hat{m}_{j,i}, m_{i,i}^{-2}\right)\\
&= \frac{m_{i,i}^2}{2}\left\|\mathbf{y}_{i,:} - \sum_{j\in\mathcal{N}(i)} \mathbf{y}_{j,:}\hat{m}_{j,i}\right\|_2^2 - \frac{1}{2}\log m_{i,i}^2 + \text{const}\\
&= \frac{1}{2}\mathbf{m}_{:,i}^\top \mathbf{Y}\mathbf{Y}^\top \mathbf{m}_{:,i} - \frac{1}{2}\log m_{i,i}^2 + \text{const}.
\end{aligned} \tag{8}$$

The overall objective is the sum of the objectives for each column of $\mathbf{W}$. Under the probabilistic interpretation this is equivalent to assuming independence between the individual regressions. The objective can be written in matrix form as

$$E(\mathbf{W}) = \frac{1}{2}\sum_{i=1}^{n} \mathbf{m}_{:,i}^\top \mathbf{Y}\mathbf{Y}^\top \mathbf{m}_{:,i} - \frac{1}{2}\sum_{i=1}^{n}\log m_{i,i}^2 + \text{const}. \tag{9}$$

Recalling that our definition of $\mathbf{M}$ was in terms of $\mathbf{W}$, we now make that dependence explicit by parameterizing the objective function only in terms of the non-zero elements of $\mathbf{W}$. To do this we introduce a 'croupier matrix' $\mathbf{S}_i \in \Re^{n\times k_i}$, where k_i is the size of the i data point's neighborhood. This matrix will distribute the non-zero elements of $\mathbf{W}$ appropriately into $\mathbf{M}$. It is defined in such a way that for the ith data point we have $\mathbf{m}_{:,i} = \mathbf{S}_i\mathbf{w}_i$, where we use the shorthand $\mathbf{w}_i = \mathbf{w}_{\mathcal{N}(i),i}$. In other words $\mathbf{w}_i$ is the vector of regression weights being used to reconstruct the ith data point. It contains the non-zero elements from the ith column of $\mathbf{W}$. The matrix $\mathbf{S}_i$ is constructed by setting all elements in its ith row to -1 (causing $m_{i,i}$ to be the negative sum of the elements of $\mathbf{w}_i$ as defined). Then we set $s_{\ell,j}$ to 1 if ℓ is the jth neighbor of the data point i and zero otherwise. We can then rewrite the objective function for the data set as

$$E(\mathbf{W}) = \frac{1}{2}\sum_{i=1}^{n} \mathbf{w}_i^\top \mathbf{S}_i^\top \mathbf{Y}\mathbf{Y}^\top \mathbf{S}_i\mathbf{w}_i - \frac{1}{2}\sum_{i=1}^{n}\log \mathbf{w}_i^\top \mathbf{1}\mathbf{1}^\top \mathbf{w}_i + \text{const},$$

A fixed point can be found by taking gradients with respect to $\mathbf{w}_i$,

$$\frac{\mathrm{d}E(\mathbf{W})}{\mathrm{d}\mathbf{w}_i} = \mathbf{S}_i^\top \mathbf{Y}\mathbf{Y}^\top \mathbf{S}_i\mathbf{w}_i - \frac{1}{\mathbf{w}_i^\top \mathbf{1}}\mathbf{1}$$

which implies that the direction of $\mathbf{w}_i$ is given by

$$\mathbf{w}_i \propto \mathbf{C}_i^{-1}\mathbf{1}$$

where $\mathbf{C}_i = \mathbf{S}_i^\top \mathbf{Y}\mathbf{Y}^\top \mathbf{S}_i$ has been called the "local covariance matrix" by Roweis and Saul (2000), removing the croupier matrix we can express the local covariance matrix in the same form given by Roweis and Saul (2000),

$$\mathbf{C}_i = \sum_{j \in \mathcal{N}(i)} (\mathbf{y}_{j,:} - \mathbf{y}_{i,:})(\mathbf{y}_{j,:} - \mathbf{y}_{i,:})^\top.$$

For standard LLE the magnitude of the vector $\mathbf{w}_i$ is set by the fact that $\mathbf{1}^\top \mathbf{w}_i = 1$. In our alternative formulation we can find the magnitude of the vector through differentiation of (8) with respect to $m_{i,i}^2$ leading to the following fixed point

$$m_{i,i}^{-2} = \left\| \mathbf{y}_{i,:} - \sum_{j \in \mathcal{N}(i)} \mathbf{y}_{j,:} \hat{m}_{j,i} \right\|_2^2,$$

where $\hat{m}_{j,i} = -m_{j,i}/m_{i,i}$. This update shows explicitly that $m_{i,i}$ estimates the precision with which each individual regression problem is solved.

2.7.1 DETERMINING THE EMBEDDING IN LLE

If the data is truly low dimensional, then we might expect that the local linear relationships between neighbors continue to hold for a data set $\mathbf{X}$, of lower dimensionality, $q < p$, than the original data $\mathbf{Y}$. The next step in the LLE procedure is to find this data set. We do this by minimizing the objective function in (9) with respect to this new, low dimensional data set. Writing the objective in terms of this reduced dimensional data set, $\mathbf{X}$, we have

$$\begin{aligned} E(\mathbf{X}) &= \frac{1}{2} \sum_{i=1}^{n} \mathbf{m}_{:,i}^\top \mathbf{X}\mathbf{X}^\top \mathbf{m}_{:,i} + \text{const} \\ &= \frac{1}{2} \text{tr}\left(\mathbf{M}\mathbf{M}^\top \mathbf{X}\mathbf{X} \right) + \text{const} \\ &= \frac{1}{2} \sum_{i=1}^{n} \mathbf{x}_{i,:}^\top \mathbf{M}\mathbf{M}^\top \mathbf{x}_{i,:} + \text{const}. \end{aligned}$$

Clearly the objective function is trivially minimized by setting $\mathbf{X} = \mathbf{0}$, so to avoid this solution a constraint is imposed that $\mathbf{X}^\top \mathbf{X} = \mathbf{I}$. This leads to an eigenvalue problem of the form

$$\mathbf{M}\mathbf{M}^\top \mathbf{u}_i = \lambda_i \mathbf{u}_i.$$

Here the smallest $q+1$ eigenvalues are extracted. The smallest eigenvector is the constant eigenvector and is associated with an eigenvalue of zero. This is because, by construction, we have set $\mathbf{M}\mathbf{M}^\top \mathbf{1} = \mathbf{0}$. The next q eigenvectors are retained to make up the low dimensional representation so we have

$$\mathbf{x}_{:,i} = \mathbf{u}_{i+1} \quad \text{for} \quad i = 1..q.$$

Extracting the latent coordinates in LLE is extremely similar to the process suggested in Laplacian eigenmaps, despite different motivations. Though in the LLE case the constraint on the latent embeddings is not scaled by the degree matrix. The procedure is also identical to that used in classical multidimensional scaling, and therefore matches that used in MVU and MEU, although again the motivation is different. Rather than distance matching, as suggested for CMDS, in LLE we are looking for a 'representative,' low dimensional, data set.

2.7.2 RELATING LLE TO MEU

We can see the similarity now between LLE and the Laplacian eigenmaps. If we interpret $\mathbf{M}\mathbf{M}^\top$ as a Laplacian we notice that the eigenvalue problem being solved for LLE to recover the embedding is similar to that being solved in Laplacian eigenmaps. The key difference between LLE and Laplacian eigenmaps is the manner in which the Laplacian is parameterized.

When introducing MEU we discussed how it is necessary to constrain the Laplacian matrix to be positive definite (see Section 2.5.1). One way of doing this is to assume the Laplacian factorizes as

$$\mathbf{L} = \mathbf{M}\mathbf{M}^\top$$

where $\mathbf{M}$ is non-symmetric. If $\mathbf{M}$ is constrained so that $\mathbf{M}^\top \mathbf{1} = \mathbf{0}$ then we will also have $\mathbf{L}\mathbf{1} = \mathbf{0}$. As we saw in the last section, this constraint is easily achieved by setting the diagonal elements $m_{i,i} = -\sum_{j \in \mathcal{N}(i)} m_{j,i}$. Then if we force $m_{j,i} = 0$ if $j \notin \mathcal{N}(i)$ we will have a Laplacian matrix which is positive semidefinite without need for any further constraint on $\mathbf{M}$. The sparsity pattern of $\mathbf{L}$ will, however, be different from the pattern of $\mathbf{M}$. The entry for $\ell_{i,j}$ will only be zero if there are no shared neighbors between i and j.

We described above how the parameters of LLE, $\mathbf{W}$, are chosen to reflect locally linear relationships between neighboring data points. Here we show that this algorithm is actually approximate maximum likelihood in the MEU model. Indeed LLE turns out to be the specific case of maximum entropy unfolding where:

1. The diagonal sums, $m_{i,i}$, are further constrained to unity.

2. The parameters of the model are optimized by maximizing the *pseudolikelihood* of the resulting GRF.

As we described in our introduction to LLE, traditionally the reconstruction weights, $\mathbf{w}_i$, are constrained to sum to 1. If this is the case then by our definition of $\mathbf{M}$ we can write $\mathbf{M} = \mathbf{I} - \mathbf{W}$. The sparsity pattern of $\mathbf{W}$ matches $\mathbf{M}$, apart from the diagonal of $\mathbf{W}$ which is set to zero. These constraints mean that $(\mathbf{I} - \mathbf{W})^\top \mathbf{1} = \mathbf{0}$. The LLE algorithm (Roweis and Saul, 2000) proscribes that the smallest eigenvectors of $(\mathbf{I} - \mathbf{W})(\mathbf{I} - \mathbf{W})^\top = \mathbf{M}\mathbf{M}^\top = \mathbf{L}$ are used with the constant eigenvector associated with the eigenvalue of 0 being discarded. This matches the CMDS procedure as applied to the MEU model, where the eigenvectors of $\mathbf{L}$ are computed with the smallest eigenvector discarded through the centering operation.

2.7.3 PSEUDOLIKELIHOOD APPROXIMATION

To see how pseudolikelihood in the MEU model results in the LLE procedure we firstly review the pseudolikelihood approximation (Besag, 1975).

The Hammersley-Clifford theorem (Hammersley and Clifford, 1971) states that for a Markov random field (of which our Gaussian random field is one example) the joint probability density can be represented as a factorization over the cliques of the graph. In the Gaussian random field underlying maximum entropy unfolding the cliques are defined by the neighbors of each data point and the relevant factorization is

$$p(\mathbf{Y}) \propto \prod_{i=1}^{n} p(\mathbf{y}_{i,:} | \mathbf{Y}_{\setminus i}), \tag{10}$$

where $\mathbf{Y}_{\setminus i}$ represents all data other than the ith point and in practice each conditional distribution is typically only dependent on a sub-set of $\mathbf{Y}_{\setminus i}$ (as defined by the neighborhood). As we will see, these conditional distributions are straightforward to write out for maximum entropy unfolding, particularly in the case where we have assumed the factorization of the Laplacian, $\mathbf{L} = \mathbf{M}\mathbf{M}^\top$.

The pseudolikelihood assumes that the proportionality in (10) can be ignored and that the approximation

$$p(\mathbf{Y}) \approx \prod_{i=1}^{n} p(\mathbf{y}_{i,:} | \mathbf{Y}_{\setminus i})$$

is valid.

To see how the decomposition into cliques applies in the factorizable MEU model first recall that

$$\mathrm{tr}\left(\mathbf{Y}\mathbf{Y}^\top \mathbf{M}\mathbf{M}^\top\right) = \sum_{i=1}^{n} \mathbf{m}_{:,i}^\top \mathbf{Y}\mathbf{Y}^\top \mathbf{m}_{:,i}$$

so for the MEU model we have[6]

$$p(\mathbf{Y}) \propto \exp\left(-\frac{1}{2}\mathrm{tr}\left(\mathbf{Y}\mathbf{Y}^\top\mathbf{M}\mathbf{M}^\top\right)\right) = \prod_{i=1}^{n}\exp\left(-\frac{1}{2}\mathbf{m}_{:,i}^\top\mathbf{Y}\mathbf{Y}^\top\mathbf{m}_{:,i}\right).$$

This provides the necessary factorization for each conditional density which can now be rewritten as

$$p(\mathbf{y}_{i,:}|\mathbf{Y}_{\setminus i}) = \left(\frac{m_{i,i}^2}{\tau}\right)^{\frac{p}{2}}\exp\left(-\frac{m_{i,i}^2}{2}\left\|\mathbf{y}_{i,:} + \sum_{j\in\mathcal{N}(i)}\frac{m_{j,i}}{m_{i,i}}\mathbf{y}_{j,:}\right\|_2^2\right).$$

Optimizing the pseudolikelihood is equivalent to optimizing the conditional density for each of these cliques independently,

$$\log p(\mathbf{Y}) \approx \sum_{i=1}^{n}\log p(\mathbf{y}_{i,:}|\mathbf{Y}_{\setminus i}),$$

which is equivalent to solving n independent regression problems with a constraint on the regression weights that they sum to one. This is exactly the optimization suggested in (9). In maximum entropy unfolding the constraint arises because the regression weights are constrained to be $w_{j,i}/m_{i,i}$ and $m_{i,i} = \sum_{j\in\mathcal{N}(i)} w_{j,i}$. In standard LLE a further constraint is placed that $m_{i,i} = 1$ which implies none of these regression problems should be solved to a greater precision than another. However, as we derived above, LLE is also applicable even if this further constraint is not imposed.

Locally linear embeddings make use of the pseudolikelihood approximation to parameter determination Gaussian random field. Underpinning this is a neat way of constraining the Laplacian to be positive semidefinite by assuming a factorized form. The pseudolikelihood also allows for relatively quick parameter estimation by ignoring the partition function from the actual likelihood. This again removes the need to invert to recover the covariance matrix and means that LLE can be applied to larger data sets than MEU or MVU. However, the sparsity pattern in the Laplacian for LLE will not match that used in the Laplacian for the other algorithms due to the factorized representation.

2.7.4 WHEN IS THE PSEUDOLIKELIHOOD VALID IN LLE?

The pseudolikelihood was motivated by Besag (1975) for computational reasons. However, it obtains speed ups whilst sacrificing accuracy: it does not make use of the correct form of the normalization of the Gaussian random field. For a Gaussian model the normalization is the determinant of the covariance matrix,

$$|\mathbf{K}| = |\mathbf{L} + \gamma\mathbf{I}|^{-1}.$$

However, under particular circumstances the approximation is exact. Here we quickly review an occasion when this occurs.

Imagine if we force $\mathbf{M}$ to be lower triangular, that is, we have a Cholesky form for our factorization of $\mathbf{L} = \mathbf{M}\mathbf{M}^\top$. The interpretation here is now that $\mathbf{M}$ is a weighted adjacency matrix from a *directed acyclic graph*. When constructing the LLE neighborhood the triangular form for this matrix can be achieved by first imposing an ordering on the data points. Then, when seeking the nearest K neighbors for i, we only consider a candidate data point j if $j > i$. In the resulting directed acyclic graph the neighbors of each data point are its *parents*.[7] The weighting of the edge between node j and its parent, i, is given by the (i,j)th element of $\mathbf{M}$. To enforce the constraint that $\mathbf{M}^\top\mathbf{1} = \mathbf{0}$ the diagonal elements of $\mathbf{M}$ are given by the negative sum of the off diagonal elements from each column (i.e., the sum of their parents). Note that the last data point (index n) has no parents and so the (n,n)th element of $\mathbf{M}$ is zero.

6. Here we have ignored the term arising from the base density, $\mathrm{tr}\left(\gamma\mathbf{Y}\mathbf{Y}^\top\right)$. It also factorizes, but it does not affect the dependence of the pseudolikelihood on $\mathbf{W}$.

7. Note that parents having a *lower* index than children is the reverse of the standard convention. However, here it is necessary to maintain the structure of the Cholesky decomposition.

Now we use the fact that the log determinant of $\mathbf{L}$ is given by $\log\left|\mathbf{M}\mathbf{M}^\top\right| = \sum_i \log m_{i,i}^2$ if $\mathbf{M}$ is lower triangular. This means that for the particular structure we have imposed on the covariance the true log likelihood *does* factorize into n independent regression problems,

$$\begin{aligned}\log p(\mathbf{Y}) &= \frac{\left|\mathbf{M}\mathbf{M}^\top\right|^{\frac{1}{2}}}{\tau^{\frac{n}{2}}}\exp\left(-\frac{1}{2}\mathrm{tr}\left(\mathbf{Y}\mathbf{Y}\mathbf{M}\mathbf{M}^\top\right)\right)\\ &= \prod_i^n \frac{m_{i,i}^2}{\tau^{\frac{1}{2}}}\exp\left(-\frac{m_{i,i}^2}{2}\left\|\mathbf{y}_{i,:}+\sum_{j\in\mathcal{N}(i)}\frac{m_{j,i}}{m_{i,i}}\mathbf{y}_{j,:}\right\|_2^2\right).\end{aligned}$$

The representation corresponds to a Gaussian random field which is constructed from specifying the directed relationship between the nodes in the graph. We can derive the Gaussian random field by considering a series of conditional relationships,

$$p(\mathbf{y}_{i,:}|\mathbf{Y}_{\setminus i}) = p(\mathbf{y}_{i,:}|\mathbf{Y}_{j>i,:})$$

where our notation here is designed to indicate that the model is constrained so that the density associated with each data point, $\mathbf{y}_{i,:}$, is only dependent on data points with an index greater than i, a matrix we denote with $\mathbf{Y}_{j>i,:}$. This constraint is enforced by our demand that the only potential neighbors (parents in the directed graph) are those data points with an index greater than i. The undirected system can now be produced by taking the conditional densities of each data point,

$$p(\mathbf{y}_{i,:}|\mathbf{Y}_{j>i,:}) = \mathcal{N}\left(\mathbf{y}_{i,:}|\mathbf{Y}_{j>i,:}^\top\mathbf{m}_{j>i,i}, m_{i,i}^{-2}\mathbf{I}\right),$$

and multiplying them together[8]

$$p(\mathbf{Y}) = \prod_{i=1}^n p(\mathbf{y}_{i,:}|\mathbf{Y}_{j>i,:}),$$

to form the joint density. Note that the nth data point has no parents so we can write $p(\mathbf{y}_{n,:}|\mathbf{Y}_{j>n,:}) = p(\mathbf{y}_{n,:})$. However, since we defined $m_{j,j} = -\sum_{i>j} m_{i,j}$ the model as it currently stands associates an infinite variance with this marginal density ($m_{n,n} = 0$). This is a consequence of the constraint $\mathbf{M}^\top\mathbf{1} = \mathbf{0}$. The problem manifests itself when computing the log determinant of $\mathbf{L}$, $\log\left|\mathbf{M}\mathbf{M}^\top\right| = \sum_i \log m_{i,i}^2$ to develop the log likelihood. The last term in this sum is now $\log m_{n,n}^2 = \log 0$. As for the standard model this is resolved if we include the $\gamma\mathbf{I}$ term from the base density when computing the determinant, but this destroys the separability of the determinant computation. If the likelihood is required the value $m_{n,n}$ could be set to a small value, or optimized, relaxing the constraint on $\mathbf{M}$.

We call the algorithm based on the above decomposition acyclic locally linear embedding (ALLE). A weakness for the ALLE is the need to specify an ordering for the data. The ordering specifies which points can be neighbors and different orderings will lead to different results. Ideally one might want to specify the sparsity pattern in $\mathbf{L}$ and derive the appropriate sparsity structure for $\mathbf{M}$. However, given a general undirected graph it is not possible, in general, to find an equivalent directed acyclic graph. This is because co-parents in the directed graph gain an edge in the undirected graph, but the weight associated with this edge cannot be set independently of the weights associated with the edges between those co-parents and their children.

2.7.5 LLE AND PCA

LLE is motivated by considering local linear embeddings of the data, although interestingly, as we increase the neighborhood size to $K = n-1$ we do not recover PCA, which is known to be the optimal linear embedding of the data under linear Gaussian constraints. The fact that LLE is optimizing the pseudolikelihood makes it clear why this is the case. In contrast the MEU algorithm, which LLE approximates, does recover PCA when $K = n-1$. The ALLE algorithm also recovers PCA.

8. The idea of modelling a joint distribution by approximating its conditionals was described by Li and Stephens (2003) in the context of haplotype modeling. Generic density estimators based on this idea appeared earlier (e.g., Frey et al., 1996), using sigmoid belief networks (Neal, 1992). Larochelle and Murray (2011) recently extended these type of models with latent variables for modeling binary data in the neural autoregressive distribution estimator.

2.8 Relation to Isomap

The isomap algorithm (Tenenbaum et al., 2000) more directly follows the CMDS framework. In isomap (Tenenbaum et al., 2000) a sparse graph of distances is created between all points considered to be neighbors. This graph is then filled in for all non-neighboring points by finding the shortest distance between any two neighboring points in the graph (along the edges specified by the neighbors). The resulting matrix is then element-wise squared to give a matrix of square distances which is then processed in the usual manner (centering and multiplying by -0.5) to provide a similarity matrix for multidimensional scaling. Compare this to the situation for MVU and MEU. Both MVU and MEU can be thought of as starting with a sparse graph of (squared) distances. The other distances are then filled in by either maximizing the trace of the associated covariance or maximizing the entropy. Importantly, though, the interneighbor distances in this graph are preserved (through constraints imposed by Lagrange multipliers) just like in isomap. For both MVU and MEU the covariance matrix, $\mathbf{K}$, is guaranteed positive semidefinite because the distances are implied by an underlying covariance matrix that is constrained positive definite. For isomap the shortest path algorithm is effectively approximating the distances between non-neighboring points. This can lead to an implied covariance matrix which has negative eigenvalues (see Weinberger et al., 2004). The algorithm is still slower than LLE and Laplacian eigenmaps because it requires a dense eigenvalue problem and the application of a shortest path algorithm to the graph provided by the neighbors.

3. Estimating Graph Structure

The relationship between spectral dimensionality reduction algorithms and Gaussian random fields now leads us to consider a novel approach to dimensionality reduction. Recently it has been shown that the structure of a Gaussian random field can be estimated through using L1 shrinkage on the parameters of the inverse covariance (see Hastie et al., 2009, Chapter 17). These sparse graph estimators are attractive as the regularization allows some structure determination. In other words, rather than relying entirely on the structure provided by the K nearest neighbors in data space, we can estimate this structure from the data. We call the resulting class of approaches Dimensionality reduction through Regularization of the Inverse covariance in the Log Likelihood (DRILL).

Before introducing the method, we need to first re-derive the maximum entropy approach by constraining the second moment of neighboring data points to equal the empirical observation instead of the expected inter data point squared distances. We first define the empirically observed second moment observation to be

$$\mathbf{S} = \mathbf{Y}\mathbf{Y}^\top$$

so if two points, i and j are neighbors then we constrain

$$s_{i,j} = \left\langle \mathbf{y}_{i,:}^\top \mathbf{y}_{j,:} \right\rangle,$$

where $s_{i,j}$ is the i,jth element of $\mathbf{S}$. If we then further constrain the diagonal moments,

$$\left\langle \mathbf{y}_{i,:}^\top \mathbf{y}_{i,:} \right\rangle = s_{i,i} \tag{11}$$

then the expected squared distance between two data points, will be given by

$$\left\langle d_{i,j} \right\rangle = \left\langle \mathbf{y}_{i,:}^\top \mathbf{y}_{i,:} \right\rangle - 2\left\langle \mathbf{y}_{i,:}^\top \mathbf{y}_{j,:} \right\rangle + \left\langle \mathbf{y}_{j,:}^\top \mathbf{y}_{j,:} \right\rangle = s_{i,i} - 2s_{i,j} + s_{j,j}.$$

So the expected interpoint squared distance will match the empirically observed interpoint squared distance from the data. In other words, whilst we have formulated the constraints slightly differently, the final model will respect the same interpoint squared distance constraints as our original formulation of maximum entropy unfolding.

The maximum entropy solution for the distribution has the form

$$p(\mathbf{Y}) \propto \exp\left(-\frac{1}{2}\mathrm{tr}\left(\mathbf{Y}\mathbf{Y}^\top(\mathbf{\Lambda}+\gamma\mathbf{I})\right)\right),$$

where now the matrix of Lagrange multipliers matches the sparsity structure of the underlying neighborhood graph but also contains diagonal elements to enforce the constraint from (11). Writing the full log likelihood in terms of the matrix $\mathbf{S}$ we have

$$\log p(\mathbf{Y}) = -\frac{pn}{2}\log\tau + \frac{p}{2}\log|\mathbf{\Lambda}+\gamma\mathbf{I}| - \frac{1}{2}\mathrm{tr}\left(\mathbf{S}(\mathbf{\Lambda}+\gamma\mathbf{I})\right),$$

Once again, maximum likelihood in this system is equivalent to finding the Lagrange multipliers so, given the structure from the neighborhood relationships, we simply need to maximize the likelihood to solve the system. That will lead to an implied covariance matrix,

$$\mathbf{K} = (\mathbf{\Lambda}+\gamma\mathbf{I})^{-1},$$

which once again should be centred, $\mathbf{B} = \mathbf{HKH}$, and the principal eigenvectors extracted to visualize the embedding. Here, though, we are proposing some additional structure learning. If elements of the inverse covariance are regularized appropriately the model can perform some additional structure learning. In particular recent work on application of L1 priors on the elements of the inverse covariance (see, e.g., Banerjee et al., 2007; Friedman et al., 2008) allows us to apply a L1 regularizer to the inverse covariance and learn the elements of $\mathbf{\Lambda}$ efficiently. The objective function for this system is now

$$E(\mathbf{\Lambda}) = -\log p(\mathbf{Y}) + \sum_{i<j}\left\|\lambda_{i,j}\right\|_1$$

There has been a great deal of recent work on maximizing objectives of this form. In our experiments we used the graphical lasso algorithm (Friedman et al., 2008) which converts the optimization into a series of iteratively applied lasso regressions.

4. Experiments

The models we have introduced are illustrative and draw on the connections between existing methods. The advantages of our approaches are in the unifying perspective they give and their potential to exploit the characteristics of the probabilistic formulation to explore extensions based on missing data, Bayesian formulations etc.. However, for illustrative purposes we conclude with a short experimental section.

For our experiments we consider two real world data sets. Code to recreate all our experiments is available online. We applied each of the spectral methods we have reviewed along with MEU using positive constraints on the Lagrange multipliers (denoted MEU) and the DRILL described in Section 3. To evaluate the quality of our embeddings we follow the suggestion of Harmeling (2007) and use the GP-LVM likelihood (Lawrence, 2005). The higher the likelihood the better the embedding. Harmeling conducted exhaustive tests over different manifold types (with known ground truth) and found the GP-LVM likelihood was the best indicator of the manifold quality amoungst all the measures he tried. Our first data set consists of human motion capture data.

4.1 Motion Capture Data

The data consists of a 3-dimensional point cloud of the location of 34 points from a subject performing a run. This leads to a 102 dimensional data set containing 55 frames of motion capture. The subject begins the motion from stationary and takes approximately three strides of run. We hope to see this structure in the visualization: a starting position followed by a series of loops. The data was made available by Ohio State University. The data is characterized by a cyclic pattern during the strides of run. However, the angle of

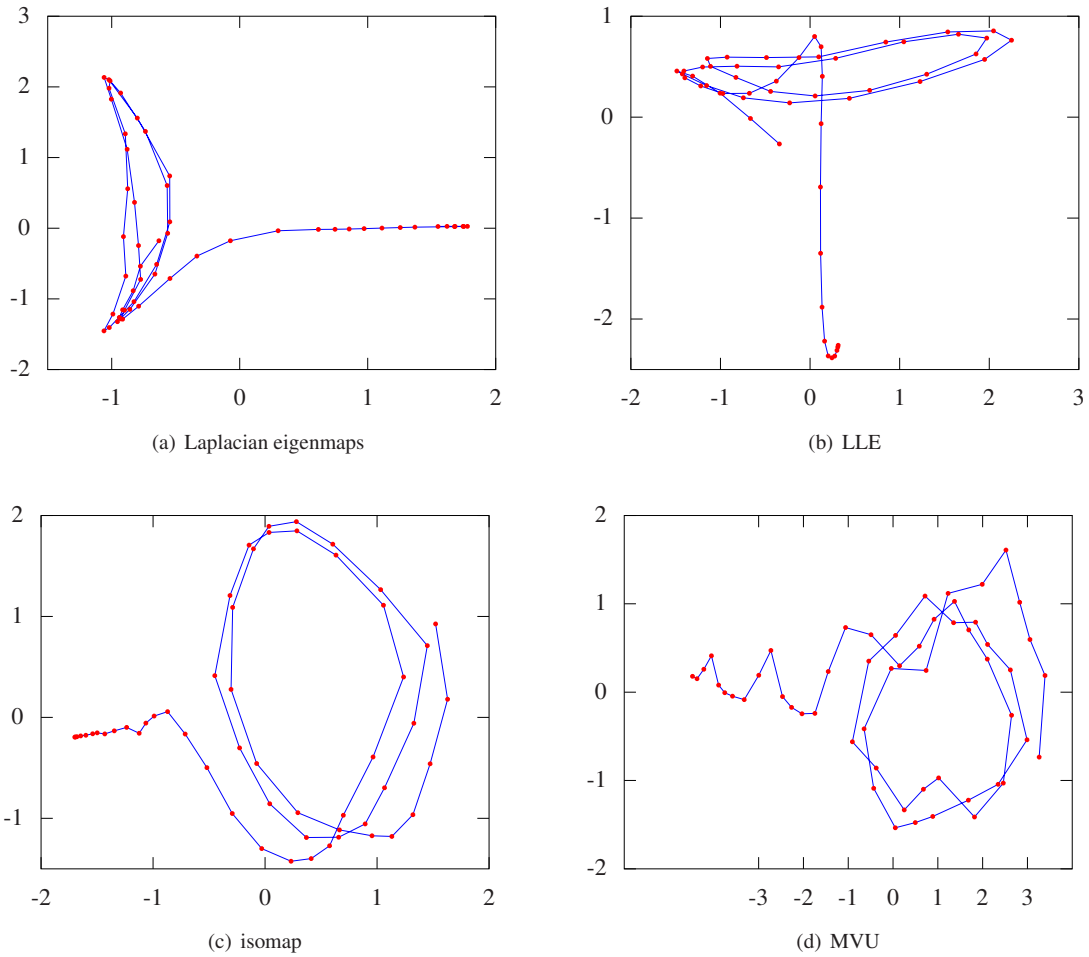
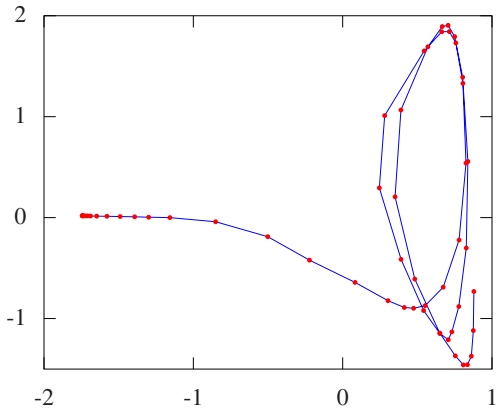


Figure 3: Motion capture data visualized in two dimensions for each algorithm we reviewed using 6 nearest neighbors. Models capture either the cyclic structure or the structure associated with the start of the run or both parts.

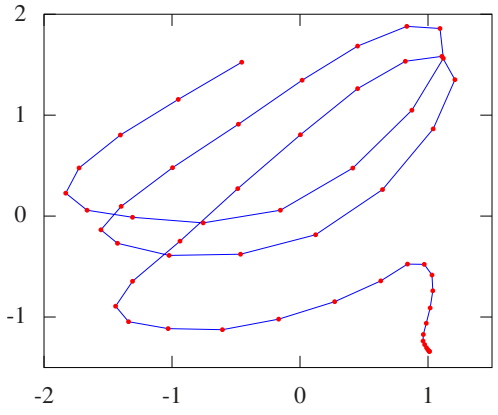
inclination during the run changes so there are slight differences for each cycle. The data is very low noise, as the motion capture rig is designed to extract the point locations of the subject to a high precision.

The two dominant eigenvectors are visualized in Figures 3–4 and the quality of the visualizations under the GP-LVM likelihood is given in Figure 7(a).

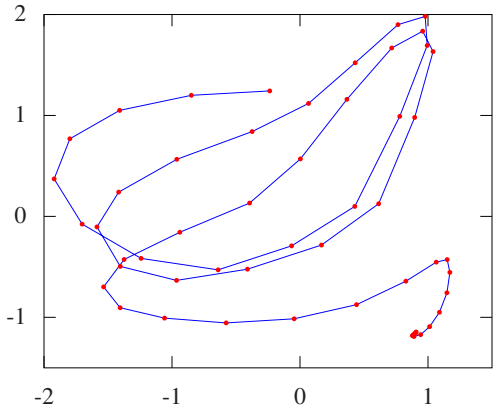
There is a clear difference in quality between the methods that constrain local distances (ALLE, MVU, isomap, MEU and DRILL) which are much better under the score than those that do not (Laplacian eigenmaps and LLE). Amongst the distance preserving methods isomap is the best performer under the GPLVM score, followed by ALLE, MVU, DRILL and MEU. The MEU model here preserves the positive definiteness of the covariance by constraining the Lagrange multipliers to be positive (an 'attractive' network as discussed in Section 2.5.1). It may be that this departure from the true maximum entropy framework explains its relatively poorer performance..



(a) MEU



(b) Acyclic LLE



(c) DRILL

Figure 4: Motion capture data visualized in two dimensions for models derived from the maximum entropy perspective. Again for each algorithm we used 6 nearest neighbors.

4.2 Robot Navigation Example

The second data set we use is a series of recordings from a robot as it traces a square path in a building. The robot records the strength of WiFi signals in an attempt to localize its position (see Ferris et al., 2007, for an application). Since the robot moves only in two dimensions, the inherent dimensionality of the data should be two: the reduced dimensional space should reflect the robot's movement. The WiFi signals are noisier than the motion capture data, so it makes an interesting contrast. The robot completes a single circuit after entering from a separate corridor, so it is expected to exhibit "loop closure" in the resulting map. The data consists of 215 frames of measurement, each frame consists of the WiFi signal strength of 30 access points.

The results for the range of spectral approaches are shown in Figures 5–6 with the quality of the methods scored in Figure 7(b). Both in the visualizations and in the GP-LVM scores we see a clear difference in quality for the methods that preserve local distances (i.e., again isomap, ALLE, MVU, MEU and DRILL are better than LLE and Laplacian eigenmaps). Amongst the methods that do preserve local distance relationships MEU seems to smooth the robot path more than the other three approaches. Given that it has the lowest score of the four distance preserving techniques this smoothing may be unwarranted. MVU appears to have an overly noisy representation of the path.

4.3 Learning the Neighborhood

Our final experiments test the ability of L1 regularization of the random field to learn the neighborhood. We firstly considered the motion capture data and used the DRILL with a large neighborhood size of 20 and L1 regularization on the parameters. As we varied the regularization coefficient we found a maximum under the GP-LVM score (Figure 8(a)). The visualization associated with this maximum is shown in Figure 8(b) this may be compared with Figure 4(c) which used 6 neighbors. Finally we investigated whether L1 regularization alone could recover a reasonable representation of the data. We again considered the motion capture data but initialized all points as neighbors. We then applied L1 regularization to learn a neighborhood structure. Again a maximum under the GP-LVM score was found (Figure 9(a)) and the visualization associated with this maximum is shown Figure 9(b).

The structural learning prior was able to improve the model fitted with 20 neighbors considerably until its performance was similar to that of the the six neighbor model shown in Figure 4(c). However, L1 regularization alone was not able to obtain such a good performance, and was unable to tease out the starting position from the rest of the run in the final visualization. It appears that structural learning using L1-priors for sparsity is not on its own enough to find an appropriate neighborhood structure for this data set.

5. Discussion and Conclusions

We have introduced a new perspective on dimensionality reduction algorithms based around maximum entropy. Our starting point was the maximum variance unfolding and our end point was a novel approach to dimensionality reduction based on Gaussian random fields and lasso based structure learning. We hope that this new perspective on dimensionality reduction will encourage new strands of research at the interface between these areas.

One feature that stands out from our unifying perspective (see also Ham et al., 2004; Bengio et al., 2004b,a) is the three separate stages used in existing spectral dimensionality algorithms.

1. A neighborhood between data points is selected. Normally k-nearest neighbors or similar algorithms are used.

2. Interpoint distances between neighbors are fed to the algorithms which provide a similarity matrix. The way the entries in the similarity matrix are computed is the main difference between the different algorithms.

3. The relationship between points in the similarity matrix is visualized using the eigenvectors of the similarity matrix.

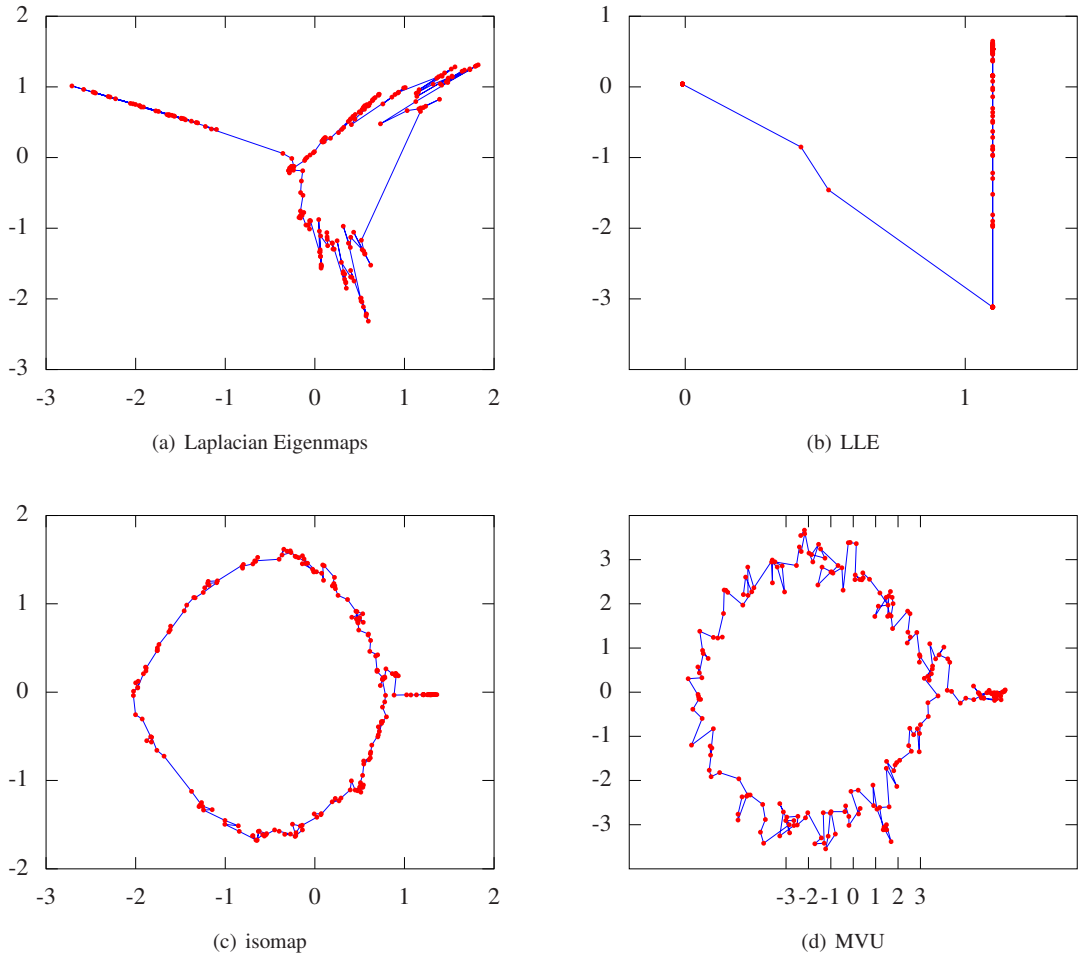
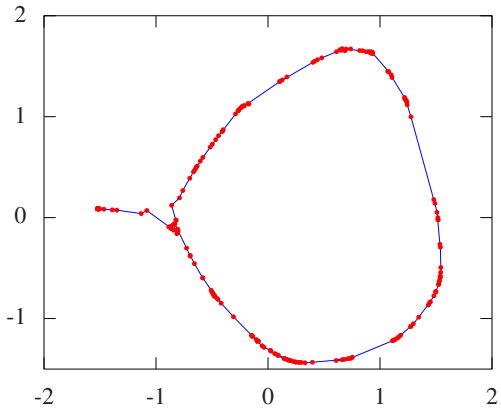


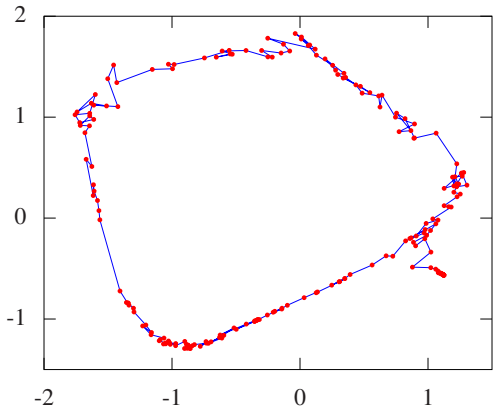
Figure 5: Visualization of the robot WiFi navigation data for different spectral algorithms we reviewed with seven neighbors used to construct graphs. LE and LLE struggle to captured the loop structure (perhaps because of the higher level of noise). Several of the models also show the noise present in the WiFi signals.

Our unifying perspective shows that actually each of these steps is somewhat orthogonal. The neighborhood relations need not come from nearest neighbors, we can use structural learning algorithms such as that suggested in DRILL to learn the interpoint structure. The main difference between the different approaches to spectral dimensionality reduction is how the entries of the similarity matrix are determined. Maximum variance unfolding looks to maximize the trace under the distance constraints from the neighbours. Our new algorithms maximize the entropy or, equivalently, the likelihood of the data. Locally linear embedding maximizes an approximation to our likelihood. Laplacian eigenmaps parameterize the inverse similarity through appealing to physical analogies. Finally, isomap uses shortest path algorithms to compute interpoint distances and centres the resulting matrix to give the similarities.

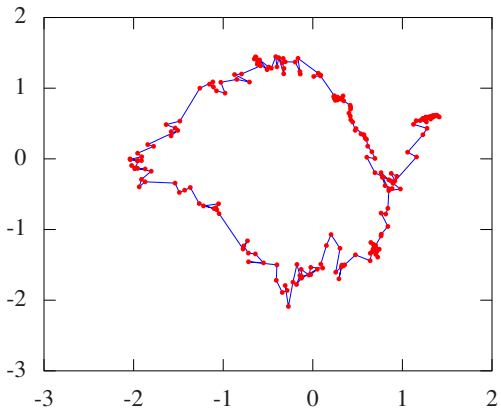
The final step of the algorithm attempts to visualize the similarity matrices using their eigenvectors. However, it simply makes use of one possible objective function to perform this visualization. Considering that underlying the similarity matrix, $\mathbf{K}$, is a sparse Laplacian matrix, $\mathbf{L}$, which represents a Gaussian-Markov



(a) MEU

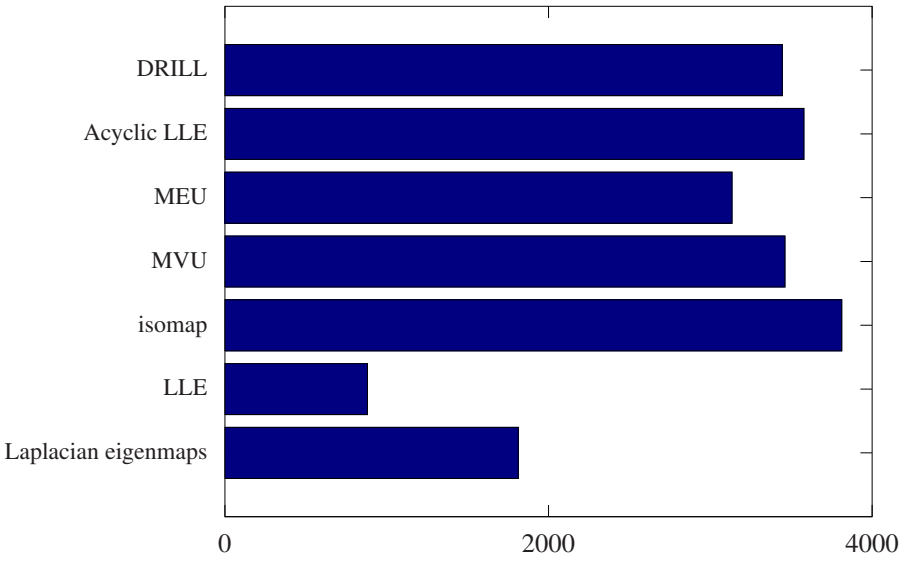


(b) Acyclic LLE

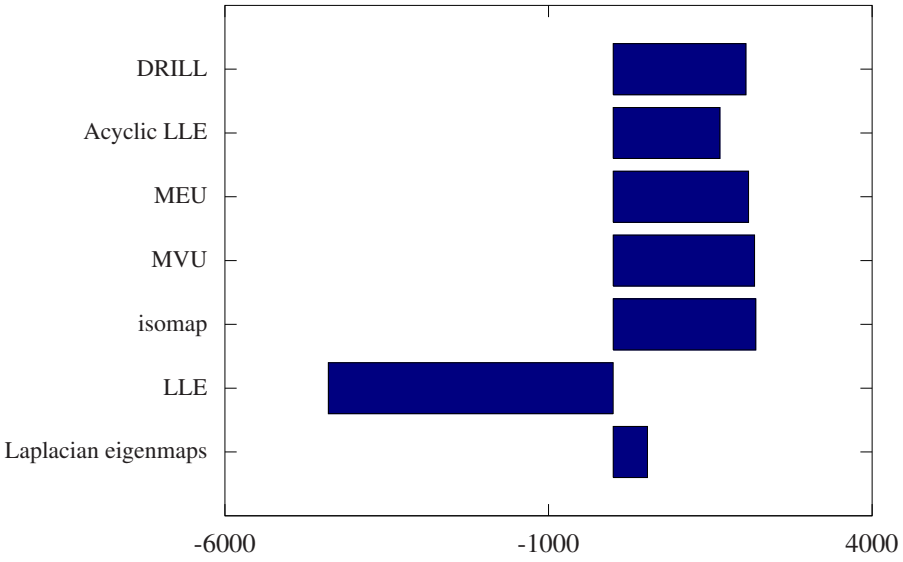


(c) DRILL

Figure 6: Visualization of the robot WiFi navigation data using algorithms based on maximum entropy. Again seven neighbors are used.

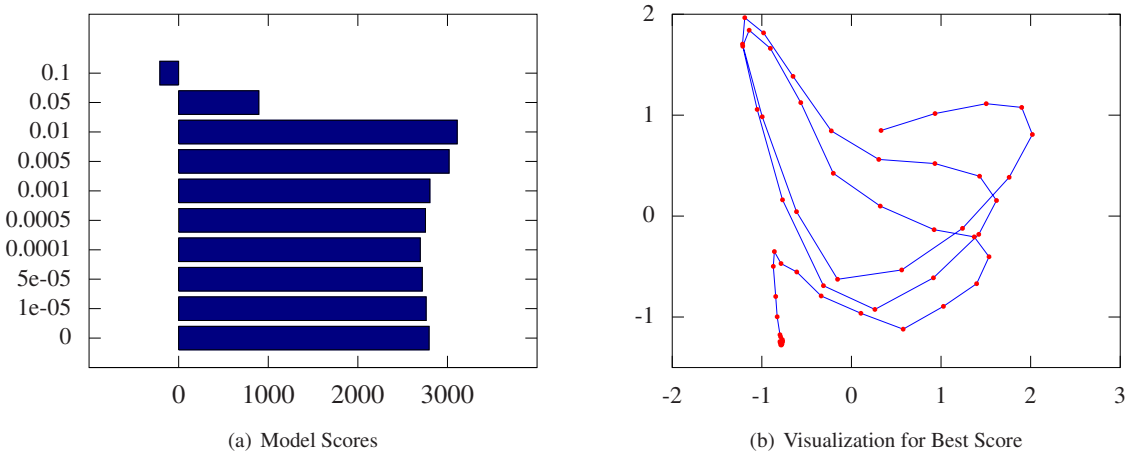


(a) Scores on Motion Capture Data



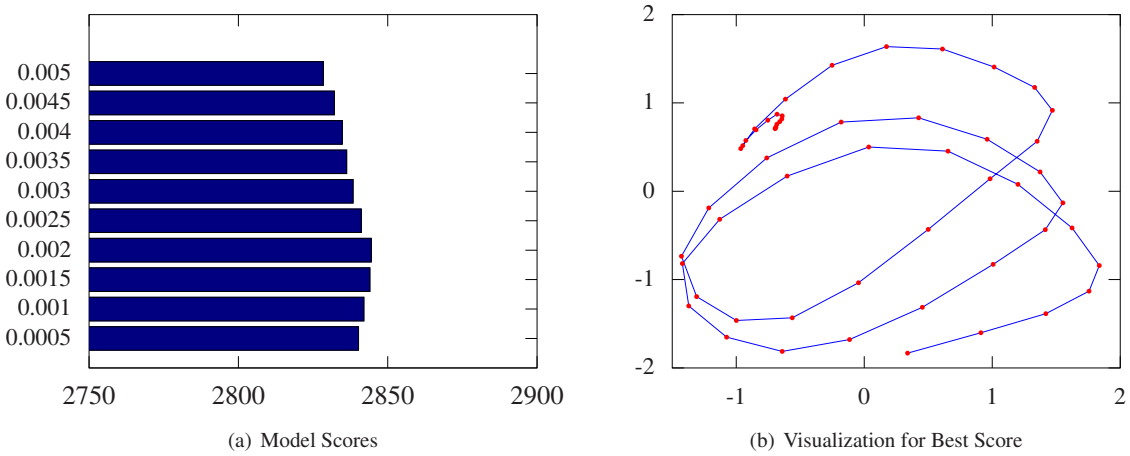
(b) Scores on Robot Navigation Data

Figure 7: Model scores for the different spectral approaches. (a) the motion capture data visualizations, (b) the robot navigation example visualizations.



(a) Model Scores (b) Visualization for Best Score

Figure 8: Structure learning for the DRILL algorithm on the motion capture data set. A model with 20 neighbors was fitted to the data. L1 regularization was used to reduce the number of neighbors associated with each data point. (a) shows the model score for the different L1 regularization parameters and (b) shows the visualization that corresponded to the best score (regularization parameter is 0.01).



(a) Model Scores (b) Visualization for Best Score

Figure 9: Full structure learning for the DRILL algorithm. Here all points are considered neighbors, the structure of the model is then recovered by L1 regularization. (a) shows the model score associated with the different L1 regularization parameters and (b) shows the visualization corresponding to the best score (regularization parameter 0.002).

random field, we can see this final step as visualizing that random field. There are many potential ways to visualize that field and the eigenvectors of the precision is just one of them. In fact, there is an entire field of graph visualization proposing different approaches to visualizing such graphs. However, we could even choose not to visualize the resulting graph. It may be that the structure of the graph is of interest in itself. Work in human cognition by Kemp and Tenenbaum (2008) has sought to fit Gaussian graphical models to data in natural structures such as trees, chains and rings. Visualization of such graphs through reduced

dimensional spaces is only likely to be appropriate in some cases, for example planar structures. For this model only the first two steps are necessary.

One advantage to conflating the three steps we have identified is the possibility to speed up the complete algorithm. For example, conflating the second and third step allows us to speed up algorithms through never explicitly computing the similarity matrix. Using the fact that the principal eigenvectors of the similarity are the minor eigenvalues of the Laplacian and exploiting fast eigensolvers that act on sparse matrices very large data sets can be addressed. However, we still can understand the algorithm from the unifying perspective while exploiting the computational advantages offered by this neat shortcut.

5.1 Gaussian Process Latent Variable Models

Finally, there are similarities between maximum entropy unfolding and the Gaussian process latent variable model (GP-LVM). Both specify a Gaussian density over the training data and in practise the GP-LVM normally makes an assumption of independence across the features. In the GP-LVM a Gaussian process is defined that maps from the latent space, $\mathbf{X}$, to the data space, $\mathbf{Y}$. The resulting likelihood is then optimized with respect to the latent points, $\mathbf{X}$. Maximum entropy unfolding leads to a Gauss Markov Random field, where the conditional dependencies are between neighbors. In one dimension, a Gauss Markov random field can easily be specified by a Gaussian process through appropriate covariance functions. The Ornstein-Uhlbeck covariance function is the unique covariance function for a stationary Gauss Markov process. If such a covariance was defined in a GP-LVM with a one dimensional latent space

$$k(x,x') = \exp(-\|x-x\prime\|_1)$$

then the inverse covariance will be sparse with only the nearest neighbors in the one dimensional latent space connected. The elements of the inverse covariance would be dependent on the distance between the two latent points, which in the GP-LVM is optimized as part of the training procedure. The resulting model is strikingly similar to the MEU model, but in the GP-LVM the neighborhood is learnt by the model (through optimization of $\mathbf{X}$), rather than being specified in advance. The visualization is given directly by the resulting $\mathbf{X}$. There is no secondary step of performing an eigendecomposition to recover the point positions. For larger latent dimensions and different neighborhood sizes, the exact correspondence is harder to establish: Gaussian processes are defined on a continuous space and the Markov property we exploit in MEU arises from discrete relations. But the models are still similar in that they proscribe a Gaussian covariance across the data points which is derived from the spatial relationships between the points.

5.2 Notes

The plots in this document were generated using MATweave (`http://staffwww.dcs.shef.ac.uk/people/N.Lawrence/matweave.html`). Code was run using Octave version 3.2.4 on `x86_64-pc-linux-gnu` architecture. Results were generated on 23/10/2010. All plots and results can be reproduced from the underlying LaTeX document which includes the MATLAB/Octave source code.

Acknowledgments

Conversations with John Kent, Chris Williams, Brenden Lake, Joshua Tenenbaum and John Lafferty have influenced the thinking in this paper.

References

Onureena Banerjee, Laurent El Ghaoui, and Alexandre d'Aspremont. Model selection through sparse maximum likelihood estimation for multivariate Gaussian or binary data. *Journal of Machine Learning Research*, 2007.

Mikhail Belkin and Partha Niyogi. Laplacian eigenmaps for dimensionality reduction and data representation. *Neural Computation*, 15(6):1373–1396, 2003. doi: 10.1162/089976603321780317.

Yoshua Bengio, Olivier Delalleau, Jean-Francois Palement, Nicolas Le Roux, Marie Ouimet, and Pascal Vincent. Learning eigenfunctions links spectral embedding and kernel PCA. *Neural Computation*, 16(10): 2197–2219, 2004a.

Yoshua Bengio, Jean-Francois Paiement, Pascal Vincent, Olivier Delalleau, Nicolas Le Roux, and Marie Ouimet. Out-of-sample extensions for LLE, isomap, MDS, eigenmaps, and spectral clustering. In Sebastian Thrun, Lawrence Saul, and Bernhard Schölkopf, editors, *Advances in Neural Information Processing Systems*, volume 16, pages 177–184, Cambridge, MA, 2004b. MIT Press.

Julian Besag. Statistical analysis of non-lattice data. *The Statistician*, 24(3):179–195, 1975.

George E. P. Box and Norman R. Draper. *Empirical Model-Building and Response Surfaces*. John Wiley and Sons, 1987. ISBN 0471810339.

Brian D. Ferris, Dieter Fox, and Neil D. Lawrence. WiFi-SLAM using Gaussian process latent variable models. In Manuela M. Veloso, editor, *Proceedings of the 20th International Joint Conference on Artificial Intelligence (IJCAI 2007)*, pages 2480–2485, 2007.

Brendan J. Frey, Geoffrey E. Hinton, and Peter Dayan. Does the wake-sleep algorithm learn good density estimators? In David Touretzky, Michael Mozer, and Mark Hasselmo, editors, *Advances in Neural Information Processing Systems*, volume 8, pages 661–670, Cambridge, MA, 1996. MIT Press.

Jerome Friedman, Trevor Hastie, and Robert Tibshirani. Sparse inverse covariance estimation with the graphical lasso. *Biostatistics*, 9(3):432–441, Jul 2008. doi: 10.1093/biostatistics/kxm045.

Jihun Ham, Daniel D. Lee, Sebastian Mika, and Bernhard Schölkopf. A kernel view of dimensionality reduction of manifolds. In Russell Greiner and Dale Schuurmans, editors, *Proceedings of the International Conference in Machine Learning*, volume 21. Omnipress, 2004.

John M. Hammersley and Peter Clifford. Markov fields on finite graphs and lattices. Technical report, 1971. URL `http://www.statslab.cam.ac.uk/~grg/books/hammfest/hamm-cliff.pdf`.

Stefan Harmeling. Exploring model selection techniques for nonlinear dimensionality reduction. Technical Report EDI-INF-RR-0960, University of Edinburgh, 2007.

Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The Elements of Statistical Learning*. Springer-Verlag, 2nd edition, 2009.

Edwin T. Jaynes. Bayesian methods: General background. In J. H. Justice, editor, *Maximum Entropy and Bayesian Methods in Applied Statistics*, pages 1–25. Cambridge University Press, 1986.

Charles Kemp and Joshua B. Tenenbaum. The discovery of structural form. *Proc. Natl. Acad. Sci. USA*, 105 (31), 2008.

Daphne Koller and Nir Friedman. *Probabilistic Graphical Models: Principles and Techniques*. MIT Press, 2009. ISBN 978-0-262-01319-2.

Solomon Kullback and Richard A. Leibler. On information and sufficiency. *Annals of Mathematical Statistics*, 22:79–86, 1951.

Hugo Larochelle and Iain Murray. The neural autoregressive distribution estimator. *JMLR: W&CP*, 15: 29–37, 2011.

Neil D. Lawrence. Gaussian process models for visualisation of high dimensional data. In Sebastian Thrun, Lawrence Saul, and Bernhard Schölkopf, editors, *Advances in Neural Information Processing Systems*, volume 16, pages 329–336, Cambridge, MA, 2004. MIT Press.

Neil D. Lawrence. Probabilistic non-linear principal component analysis with Gaussian process latent variable models. *Journal of Machine Learning Research*, 6:1783–1816, 11 2005.

Na Li and Matthew Stephens. Modeling linkage disequilibrium and identifying recombination hotspots using single-nucleotide polymorphism data. *Genetics*, 165:2213–2233, 2003. URL `http://www.genetics.org/cgi/content/abstract/165/4/2213`.

Kantilal V. Mardia, John T. Kent, and John M. Bibby. *Multivariate Analysis*. Academic Press, London, 1979. ISBN 0-12-471252-5.

Radford M. Neal. Connectionist learning of belief networks. *Artificial Intelligence*, 56:71–113, 1992.

Sam T. Roweis and Lawrence K. Saul. Nonlinear dimensionality reduction by locally linear embedding. *Science*, 290(5500):2323–2326, 2000. doi: 10.1126/science.290.5500.2323.

Bernhard Schölkopf, Alexander Smola, and Klaus-Robert Müller. Nonlinear component analysis as a kernel eigenvalue problem. *Neural Computation*, 10:1299–1319, 1998. doi: 10.1162/089976698300017467.

Joshua B. Tenenbaum, Virginia de Silva, and John C. Langford. A global geometric framework for nonlinear dimensionality reduction. *Science*, 290(5500):2319–2323, 2000. doi: 10.1126/science.290.5500.2319.

Michael E. Tipping and Christopher M. Bishop. Probabilistic principal component analysis. *Journal of the Royal Statistical Society, B*, 6(3):611–622, 1999. doi: doi:10.1111/1467-9868.00196.

Larry A. Wasserman. *All of Statistics*. Springer-Verlag, New York, 2003. ISBN 9780387402727.

Kilian Q. Weinberger, Fei Sha, and Lawrence K. Saul. Learning a kernel matrix for nonlinear dimensionality reduction. In Russell Greiner and Dale Schuurmans, editors, *Proceedings of the International Conference in Machine Learning*, volume 21, pages 839–846. Omnipress, 2004.

Christopher K. I. Williams. On a connection between kernel PCA and metric multidimensional scaling. In Todd K. Leen, Thomas G. Dietterich, and Volker Tresp, editors, *Advances in Neural Information Processing Systems*, volume 13, pages 675–681, Cambridge, MA, 2001. MIT Press.

Xiaojin Zhu, John Lafferty, and Zoubin Ghahramani. Semi-supervised learning: From Gaussian fields to Gaussian processes. Technical Report CMU-CS-03-175, Carnegie Mellon University, 2003.

Journal of Machine Learning Research 13 (2012) 1639-1663 Submitted 8/11; Revised 3/12; Published 5/12

Mixability is Bayes Risk Curvature Relative to Log Loss

Tim van Erven TIM@TIMVANERVEN.NL
Département de Mathématiques
Université Paris-Sud
91405 Orsay Cedex, France

Mark D. Reid* MARK.REID@ANU.EDU.AU
Robert C. Williamson* BOB.WILLIAMSON@ANU.EDU.AU
Research School of Computer Science
The Australian National University
Canberra ACT 0200, Australia

Editor: Gábor Lugosi

Abstract

Mixability of a loss characterizes fast rates in the online learning setting of prediction with expert advice. The determination of the mixability constant for binary losses is straightforward but opaque. In the binary case we make this transparent and simpler by characterising mixability in terms of the second derivative of the Bayes risk of proper losses. We then extend this result to multiclass proper losses where there are few existing results. We show that mixability is governed by the maximum eigenvalue of the Hessian of the Bayes risk, relative to the Hessian of the Bayes risk for log loss. We conclude by comparing our result to other work that bounds prediction performance in terms of the geometry of the Bayes risk. Although all calculations are for proper losses, we also show how to carry the results across to improper losses.

Keywords: mixability, multiclass, prediction with expert advice, proper loss, learning rates

1. Introduction

In *prediction with expert advice* (Vovk, 1990, 1995, 2001; Cesa-Bianchi and Lugosi, 2006) a learner has to predict a sequence of outcomes, which might be chosen adversarially. The setting is online, meaning that learning proceeds in rounds; and the learner is aided by a finite number of experts. At the start of each round, all experts first announce their predictions for that round, then the learner has to make a prediction, and finally the real outcome is revealed. The discrepancy between a prediction and an outcome is measured by a *loss function*, and losses add up between rounds. Finally, the goal for the learner is to minimize their *regret*, which is the difference between their cumulative loss and the cumulative loss of the best expert after T rounds.

Strategies for the learner usually come with guaranteed bounds on the regret in the worst case over all possible outcomes and expert predictions, which ensures good learning performance under all circumstances. How strong these guaranteed bounds can be depends on the loss function. Some losses are easy in the sense that the worst-case regret can be bounded by a constant, which is $O(1)$ in T. For other losses only a rate of $O(\sqrt{T})$ or worse can be guaranteed (Kalnishkan and Vyugin,

∗. Also affiliated with National ICT Australia (NICTA)

2008). Our results provide new insight and new technical tools for the class of losses for which fast, $O(1)$ rates are possible.

1.1 Fast Rates and Mixability

It is known that, under very general conditions, $O(1)$ rates are possible if and only if the loss is η-*mixable* (defined below) for some $\eta > 0$, which means that mixability characterizes fast rates. More specifically, if a loss is η-mixable and there are N experts, then using the so-called *aggregating algorithm* (Vovk, 2001) the learner is guaranteed to have regret bounded by

$$\frac{\ln N}{\eta}, \tag{1}$$

which does not grow with T. Conversely, if the loss is not η-mixable for any $\eta > 0$ and satisfies very mild regularity conditions, then it is not possible to bound the worst-case regret by an additive constant for any strategy (Kalnishkan and Vyugin, 2008; Vovk, 1995). Examples of mixable losses include the logarithmic loss, the relative entropy loss, the square loss on binary outcomes (Haussler et al., 1998) and the Brier score (Vovk and Zhdanov, 2009), which are all 1-mixable except for the square loss, which is 2-mixable.

A related condition requires the loss to be *exp-concave* (Cesa-Bianchi and Lugosi, 2006). Although exp-concavity implies mixability, the converse is not true, and therefore exp-concavity does not characterize fast rates.

Although mixability is associated with fast rates, it also appears in the analysis of losses with $O(\sqrt{T})$ rates. For example, the analysis of Kalnishkan and Vyugin (2008) may be interpreted as approximating non-mixable losses by a sequence of η-mixable losses with η going to zero (Kalnishkan and Vyugin, 2008, Remark 19). Thus mixability appears to be one of the most fundamental properties to study in the prediction with expert advice setting.

1.2 Main Results

The aggregating algorithm depends on η, and its regret bound (1) is optimized when η is as large as possible. For any loss of interest ℓ, it is thus desirable to know the largest η for which ℓ is η-mixable. We call this the *mixability constant* for ℓ.

For outcomes with two possible values, determining the mixability constant is straight-forward using a formula due to Haussler et al. (1998), but their expression has no clear interpretation. In Section 4.1 we show how, for the important class of *proper losses*, the result by Haussler et al. simplifies considerably, and may be expressed in terms of the curvature of the *Bayes risk* of the loss relative to the Bayes risk for the logarithmic loss. The relevant notions of properness and Bayes risk will first be reviewed in Section 3.

We refer to the case where outcomes have more than two possible values as the *multiclass* setting. Here no general result has previously been available, and the mixability constant has only been determined for a limited number of cases (mainly logarithmic loss and the Brier score). Our main contribution is a simple explicit formula for the mixability constant in the multiclass setting (Theorem 13 and Corollary 14), which generalises our result for binary-valued outcomes. Along the way we develop other useful characterizations of mixability in Theorem 10. We illustrate the usefulness of our results by giving a short proof for 1-mixability of the multiclass Brier score in Section 5, which is simpler than the previously known proof (Vovk and Zhdanov, 2009).

Although our results are stated for proper losses, in Section 6 we show how they carry across to losses that are not proper.

1.3 Outline

The paper is structured as follows. In the next section we introduce general notation. Then Section 3 reviews the class of proper losses and the definition of Bayes risk, along with some of their properties that are required later. It also states Condition A, which lists several continuity conditions on the loss that are required for our main results.

In Section 4 we come to the main part of the paper. There mixability is formally defined, and in Section 4.1 we state our results for binary-valued outcomes. The remainder of Section 4 is devoted to generalising this result to the multiclass setting (Theorem 13 and Corollary 14). An important intermediate result is stated in Theorem 10, and we discuss some of its direct consequences in Corollaries 11 and 12. These show that the sum of two η-mixable losses is η-mixable and that the logarithmic loss is the "most mixable" in a sense.

Section 5 contains a simplified proof for 1-mixability of the Brier score. And in Section 6 we show how our results carry across to losses that are not proper. In Section 7 we also relate our results to recent work by Abernethy et al. (2009) in a related online learning setting. Our proofs in Section 4 require some results from matrix calculus, which we review briefly in Appendix A.

2. Setting

We consider a game of *prediction with expert advice*, which goes on for rounds $t = 1,\dots,T$. At the start of each round t, N experts choose their predictions $v_t^1,\dots,v_t^N$ from a set $\mathcal{V}$; then the learner chooses their prediction $v_t \in \mathcal{V}$; and finally the true outcome $y_t \in \mathcal{Y} = \{1,\dots,n\}$ is revealed. When the outcomes are binary-valued, $n = 2$, but in the multiclass setting n can be any positive integer. Losses are measured by a function $\ell\colon \mathcal{Y} \times \mathcal{V} \to [0,\infty]$ and over the course of the game add up to $\mathrm{Loss}(T) := \sum_{t=1}^T \ell(y_t, v_t)$ for the learner and to $\mathrm{Loss}_j(T) = \sum_{t=1}^T \ell(y_t, v_t^j)$ for the j-th expert. The goal for the learner is to predict nearly as well as the best expert, as measured by the *regret*

$$R(T) = \mathrm{Loss}(T) - \min_j \mathrm{Loss}_j(T).$$

Typical strategies in the literature come with bounds on the regret that hold in the worst case, for any possible expert predictions and any possible sequence of outcomes. In particular, if the loss ℓ is η-mixable for some $\eta > 0$ and the learner predicts according to the aggregating algorithm, then the regret is bounded by

$$R(T) \le \frac{\ln N}{\eta}, \tag{2}$$

no matter what the expert predictions or the outcomes are.

2.1 Notation

We use the following notation throughout. Let $[n] := \{1,\dots,n\}$ and denote by $\mathbb{R}_+$ the non-negative reals. The transpose of a vector x is x'. If x is a n-vector, $A = \mathrm{diag}(x)$ is the $n \times n$ matrix with entries $A_{i,i} = x_i$, $i \in [n]$ and $A_{i,j} = 0$ for $i \neq j$. We also write $\mathrm{diag}(x_i)_{i=1}^n := \mathrm{diag}(x_1,\dots,x_n) := \mathrm{diag}((x_1,\dots,x_n)')$. The inner product of two n-vectors x and y is denoted by matrix product $x'y$. We

sometimes write $A \cdot B$ for the matrix product AB for clarity when required. If $A - B$ is positive definite (resp. semi-definite), then we write $A \succ B$ (resp. $A \succcurlyeq B$). The n-simplex $\Delta^n := \{(x_1, \ldots, x_n)' \in \mathbb{R}^n : x_i \geq 0,\ i \in [n],\ \sum_{i=1}^n x_i = 1\}$. Other notation (the Kronecker product $\otimes$, the derivative D, and the Hessian H) is defined in Appendix A, which also includes several matrix calculus results we use.

3. Proper Multiclass Losses

We consider multiclass losses for class probability estimation, in which predictions are probability distributions: $\mathcal{V} = \Delta^n$. As we will often consider how the loss changes as a function of the predicted distribution $q \in \Delta^n$, it is convenient to define a *partial loss function* $\ell_i(q) = \ell(i, q)$ for any outcome $i \in [n]$. Together these partial loss functions make up the full *loss function* $\ell : \Delta^n \to [0, \infty]^n$, which assigns a loss vector $\ell(q) = (\ell_1(q), \ldots, \ell_n(q))'$ to each distribution $q \in \Delta^n$. If the outcomes are distributed with probability $p \in \Delta^n$ then the *risk* for predicting q is just the expected loss

$$L(p, q) := p' \ell(q) = \sum_{i=1}^n p_i \ell_i(q).$$

The *Bayes risk* for p is the minimal achievable risk for that outcome distribution,

$$\underline{L}(p) := \inf_{q \in \Delta^n} L(p, q).$$

A loss is called *proper* whenever the minimal risk is always achieved by predicting the true outcome distribution, that is, $\underline{L}(p) = L(p, p)$ for all $p \in \Delta^n$. A proper loss is *strictly proper* if there exists no $q \neq p$ such that $L(p, q) = \underline{L}(p)$. For example, the *log loss* $\ell_{\log}(p) := (-\ln(p_1), \ldots, -\ln(p_n))'$ is strictly proper, and its corresponding Bayes risk is the entropy $\underline{L}_{\log}(p) = -\sum_{i=1}^n p_i \ln(p_i)$.

We call a proper loss ℓ *strongly invertible* if for all distributions $p \neq q \in \Delta^n$ there exists at least one outcome $i \in [n]$ such that $\ell_i(p) \neq \ell_i(q)$ and $p_i > 0$. Note that without the requirement that $p_i > 0$ this would be ordinary invertibility. One might also understand strong invertibility as saying that the loss should be invertible, and if we restrict the game to a face of the simplex (effectively removing one possible outcome), then the loss function for the resulting game should again be strongly invertible.

Since it is central to our results, we will assume all losses are strictly proper for the remainder of the paper (except Section 6 where we show how the assumption may be relaxed). Lemma 2 in the next section shows that strictness is not such a strong requirement.

3.1 Projecting Down to $n-1$ Dimensions

Because probabilities sum up to one, any $p \in \Delta^n$ is fully determined by its first $n-1$ components $\tilde{p} = (p_1, \ldots, p_{n-1})$. It follows that any function of p can also be expressed as a function of $\tilde{p}$, which is convenient in order to use the standard rules when taking derivatives on Δ^n. To go back and forth between p and $\tilde{p}$, we define $p_n(\tilde{p}) := 1 - \sum_{i=1}^{n-1} \tilde{p}_i$ and the projection

$$\Pi_\Delta(p) := (p_1, \ldots, p_{n-1})',$$

which is a continuous and invertible function from Δ^n to $\tilde{\Delta}^n := \{(p_1, \ldots, p_{n-1})' : p \in \Delta^n\}$, with continuous inverse $\Pi_\Delta^{-1}(\tilde{p}) = (\tilde{p}_1, \ldots, \tilde{p}_{n-1}, p_n(\tilde{p}))$. For similar reasons, we sometimes project loss

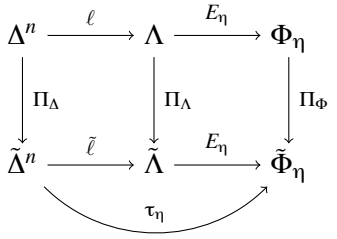


Figure 1: Mappings and spaces.

vectors $\ell(p)$ onto their first $n-1$ components $(\ell_1(p),\ldots,\ell_{n-1}(p))'$, using the projection

$$\Pi_\Lambda(\lambda) := (\lambda_1,\ldots,\lambda_{n-1})'.$$

We write $\Lambda := \ell(\Delta^n)$ for the domain of Π_Λ and $\tilde{\Lambda}$ for its range.

For loss functions $\ell(p)$, we will overload notation and abbreviate $\ell(\tilde{p}) := \ell(\Pi_\Delta^{-1}(\tilde{p}))$. In addition, we write

$$\tilde{\ell}(\tilde{p}) := \Pi_\Lambda(\ell(\tilde{p})) = (\ell_1(\tilde{p}),\ldots,\ell_{n-1}(\tilde{p}))'$$

for the first $n-1$ components of the loss (see Figure 1). By contrast, for $\underline{L}(p)$ we will be more careful about its domain, and use the separate notation $\tilde{\underline{L}}(\tilde{p}) := \underline{L}(\Pi_\Delta^{-1}(\tilde{p}))$ when we consider it as a function of $\tilde{p}$.

It may well be that one can avoid the explicit projection down to $n-1$ dimensions using the intrinsic methods of differential geometry (Thorpe, 1979), but we have been unable to prove our results using that machinery. In any case, in order to do calculations, one will need some coordinate system. Our projection simply defines the natural $(n-1)$-dimensional coordinate system on Δ^n.

3.2 First Properties

Our final result requires the following conditions on the loss:

Condition A *The loss $\ell(p)$ is strictly proper, continuous on Δ^n, and continuously differentiable on the relative interior* $\operatorname{rel\,int}(\Delta^n)$ *of its domain.*

As the projection Π_Δ is a linear function, differentiability of $\ell(p)$ is equivalent to differentiability of $\ell(\tilde{p})$, which will usually be easier to verify. Note that it follows from (15) below that existence of $\mathsf{D}\tilde{\ell}$ guarantees the existence of $\mathsf{H}\underline{L}$.

Lemma 1 *Let $\ell(p)$ be a strictly proper loss. Then the corresponding Bayes risk $\underline{L}(p)$ is strictly concave, and if $\ell(p)$ is differentiable on the relative interior* $\operatorname{rel\,int}(\Delta^n)$ *of Δ^n then it satisfies the* stationarity condition

$$p'\mathsf{D}\ell(\tilde{p}) = 0_{n-1} \qquad \text{for } p \in \operatorname{rel\,int}(\Delta^n). \tag{3}$$

If $\ell(p)$ is also continuous on the whole simplex Δ^n, then Π_Λ, $\ell(p)$ and $\tilde{\ell}(\tilde{p})$ are all continuous and invertible, with continuous inverses.

Proof Let $p_0, p_1 \in \Delta^n$ and let $p_\lambda = (1-\lambda)p_0 + \lambda p_1$. Then for any $\lambda \in (0,1)$

$$\underline{L}(p_\lambda) = p_\lambda'\ell(p_\lambda) = (1-\lambda)L(p_0,p_\lambda) + \lambda L(p_1,p_\lambda) > (1-\lambda)\underline{L}(p_0) + \lambda\underline{L}(p_1),$$

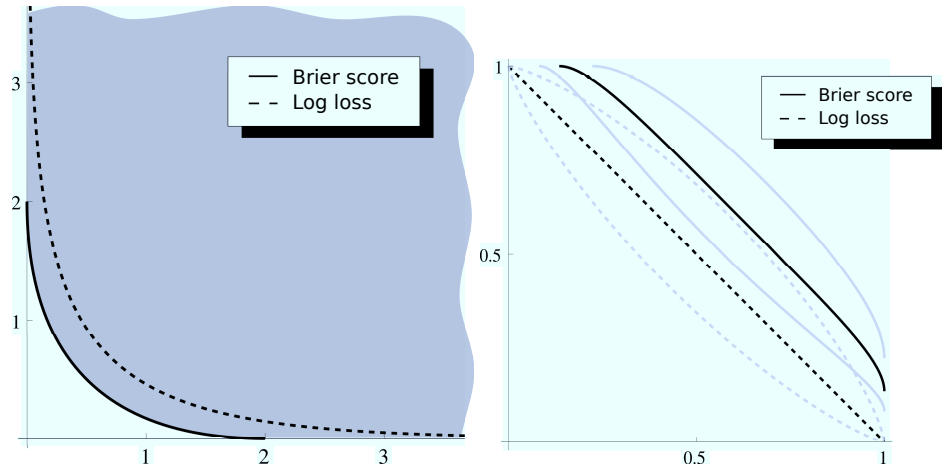


Figure 2: Left: the (boundary of the) superprediction set on two outcomes for the Brier score and the boundary of the superprediction set for log loss. Right: the same boundaries after applying the η-exponential operator for $\eta \in \{3/4, 1, 5/4\}$. The dark curves correspond to $\eta = 1$.

so $\underline{L}(p)$ is strictly concave. Properness guarantees that the function $L_p(\tilde{q}) := L(p, q(\tilde{q}))$ has a minimum at $\tilde{q} = \tilde{p}$. Hence $\mathsf{D}L_p(\tilde{q}) = p'\mathsf{D}\ell(\tilde{q}) = 0_{n-1}$ at $\tilde{q} = \tilde{p}$, giving the stationarity condition.

Now suppose ℓ is continuous on Δ^n, and observe that Π_Λ is also continuous. Then by tracing the relations in Figure 1, one sees that all remaining claims follow if we can establish invertibility of $\tilde{\ell}$ and continuity of its inverse. (Recall that Π_Δ is invertible with continuous inverse.)

To establish invertibility, suppose there exist $\tilde{p} \neq \tilde{q}$ in $\tilde{\Delta}^n$ such that $\tilde{\ell}(\tilde{p}) = \tilde{\ell}(\tilde{q})$ and assume without loss of generality that $\ell_n(p) \leq \ell_n(q)$ (otherwise, just swap them). Then $\underline{L}(q) = \tilde{q}'\tilde{\ell}(\tilde{q}) + q_n\ell_n(q) \geq \tilde{q}'\tilde{\ell}(\tilde{p}) + q_n\ell_n(p) = L(q,p)$, which contradicts strict properness. Hence $\tilde{\ell}$ must be invertible.

To establish continuity of $\tilde{\ell}^{-1}$, we need to show that $\tilde{\ell}(\tilde{p}_m) \to \tilde{\ell}(\tilde{p})$ implies $\tilde{p}_m \to \tilde{p}$ for any sequence $(\tilde{p}_m)_{m=1,2,\ldots}$ of elements from $\tilde{\Delta}^n$. To this end, let $\varepsilon > 0$ be arbitrary. Then it is sufficient to show that there exist only a finite number of elements in $(\tilde{p}_m)$ such that $\|\tilde{p}_m - \tilde{p}\| > \varepsilon$. Towards a contradiction, suppose that $(\tilde{q}_k)_{k=1,2,\ldots}$ is a subsequence of $(\tilde{p}_m)$ such that $\|\tilde{q}_k - \tilde{p}\| \geq \varepsilon$ for all $\tilde{q}_k$. Then the fact that $\tilde{\Delta}^n$ is a compact subset of $\mathbb{R}^{n-1}$ implies (by the Bolzano-Weierstrass theorem) that $(\tilde{q}_k)$ contains a converging subsequence $\tilde{r}_v \to \tilde{r}$. Since continuity of ℓ and Π_Δ^{-1} imply continuity of $\tilde{\ell}$, we have $\tilde{\ell}(\tilde{r}_v) \to \tilde{\ell}(\tilde{r})$. But since $\tilde{r}_v$ is a subsequence of $(\tilde{p}_m)$, we also have that $\tilde{\ell}(\tilde{r}_v) \to \tilde{\ell}(\tilde{p})$ and hence $\tilde{\ell}(\tilde{r}) = \tilde{\ell}(\tilde{p})$. But then strict properness implies that $\tilde{r} = \tilde{p}$, which contradicts the assumption that $\|\tilde{r}_v - \tilde{p}\| \geq \varepsilon$ for all v. ■

4. Mixability

We use the following characterisation of mixability (as discussed by Vovk and Zhdanov, 2009) and motivate our main result by looking at the binary case. To define mixability we need the notions of a superprediction set and a parametrised exponential operator. The *superprediction set* S_ℓ for a

loss $\ell : \Delta^n \to [0,\infty]^n$ is the set of points in $[0,\infty]^n$ that point-wise dominate some point on the loss surface. That is,

$$S_\ell := \{\lambda \in [0,\infty]^n \colon \exists q \in \Delta^n,\ \forall i \in [n],\ \ell_i(q) \le \lambda_i\}. \tag{4}$$

For any dimension m and $\eta \ge 0$, the η-*exponential operator* $E_\eta \colon [0,\infty]^m \to [0,1]^m$ is defined by

$$E_\eta(\lambda) := (e^{-\eta\lambda_1}, \ldots, e^{-\eta\lambda_m}).$$

For $\eta > 0$ it is clearly invertible, with inverse $E_\eta^{-1}(\phi) = -\eta^{-1}(\ln\phi_1, \ldots, \ln\phi_m)$. We will both apply it for $m = n$ and for $m = n-1$. The dimension will always be clear from the context.

A loss ℓ is η-*mixable* when the set $E_\eta(S_\ell)$ is convex. The largest η such that a loss is η-mixable is of special interest, because it determines the best possible bound in (2). We call this the *mixability constant* and denote it by η_ℓ:

$$\eta_\ell := \max\{\eta \ge 0 \colon\ \ell \text{ is } \eta\text{-mixable}\}.$$

A loss is always 0-mixable, so $\eta_\ell \ge 0$, but note that for $\eta_\ell = 0$ the bound in (2) is vacuous. A loss is therefore called *mixable* only if its mixability constant is positive, that is, $\eta_\ell > 0$.

One may rewrite the definition of $E_\eta(S_\ell)$ as follows:

$$\begin{aligned} E_\eta(S_\ell) &= \{E_\eta(\lambda) \colon \lambda \in [0,\infty]^n,\ \exists q \in \Delta^n,\ \forall i \in [n],\ \ell_i(q) \le \lambda_i\} \\ &= \{z \in [0,1]^n \colon \exists q \in \Delta^n,\ \forall i \in [n],\ e^{-\eta\ell_i(q)} \ge z_i\}, \end{aligned}$$

since $x \mapsto e^{-\eta x}$ is nonincreasing (in fact, decreasing for $\eta > 0$). Hence in order for $E_\eta(S_\ell)$ to be convex $\operatorname{graph}(f_\eta) = \Phi_\eta := \{(e^{-\eta\ell_1(q)}, \ldots, e^{-\eta\ell_n(q)}) \colon q \in \Delta^n\}$ needs to be *concave*. Here f_η is the function whose graph is given by the set above. An explicit definition of f_η is given in (11) after we have introduced some more notation. Observe that Φ_η is the (upper) boundary of $E_\eta(S_\ell)$; that is why concavity of f_η corresponds to *convexity* of $E_\eta(S_\ell)$.

Lemma 2 *If a proper, strongly invertible loss ℓ is mixable, then it is strictly proper.*

An example of a mixable proper loss that is not strictly proper, is when $\ell(p)$ does not depend on p. In this case the loss is not invertible.

Proof Suppose ℓ is not strictly proper. Then there exist $p \ne q$ such that $\underline{L}(p) = L(p,q)$. In addition, mixability implies that for any $\lambda \in (0,1)$ there exists a distribution r_λ such that for all $i \in [n]$

$$\ell_i(r_\lambda) \le -\tfrac{1}{\eta_\ell}\log\Big((1-\lambda)e^{-\eta_\ell\ell_i(p)} + \lambda e^{-\eta_\ell\ell_i(q)}\Big) \le (1-\lambda)\ell_i(p) + \lambda\ell_i(q),$$

where the second inequality follows from (strict) convexity of $x \mapsto e^{-x}$ and is strict when $\ell_i(p) \ne \ell_i(q)$. Since $\ell_i(p) \ne \ell_i(q)$ for at least one i with $p_i > 0$, it follows that

$$L(p, r_\lambda) = p'\ell(r_\lambda) < p'\big((1-\lambda)\ell(p) + \lambda\ell(q)\big) = \underline{L}(p),$$

which contradicts the definition of $\underline{L}(p)$. Thus mixability implies that ℓ must be strictly proper. ■

4.1 The Binary Case

A loss is called binary if there are only two outcomes: $n = 2$. For twice differentiable binary losses ℓ it is known (Haussler et al., 1998) that

$$\eta_\ell = \inf_{\tilde{p}\in(0,1)} \frac{\ell_1'(\tilde{p})\ell_2''(\tilde{p}) - \ell_1''(\tilde{p})\ell_2'(\tilde{p})}{\ell_1'(\tilde{p})\ell_2'(\tilde{p})(\ell_2'(\tilde{p}) - \ell_1'(\tilde{p}))}. \tag{5}$$

When a proper binary loss ℓ is differentiable, the stationarity condition (3) implies

$$\begin{aligned} & \tilde{p}\ell_1'(\tilde{p}) + (1-\tilde{p})\ell_2'(\tilde{p}) = 0 \\ \Rightarrow \quad & \tilde{p}\ell_1'(\tilde{p}) = (\tilde{p}-1)\ell_2'(\tilde{p}) && (6) \\ \Rightarrow \quad & \frac{\ell_1'(\tilde{p})}{\tilde{p}-1} = \frac{\ell_2'(\tilde{p})}{\tilde{p}} =: w(\tilde{p}) =: w_\ell(\tilde{p}). && (7) \end{aligned}$$

We have $\underline{\tilde{L}}(\tilde{p}) = \tilde{p}\ell_1(\tilde{p}) + (1-\tilde{p})\ell_2(\tilde{p})$. Thus by differentiating both sides of (6) and substituting into $\underline{\tilde{L}}''(\tilde{p})$ one obtains $\underline{\tilde{L}}''(\tilde{p}) = \frac{\ell_1'(\tilde{p})}{1-\tilde{p}} = -w(\tilde{p})$. (See Reid and Williamson, 2011). Equation 7 implies $\ell_1'(\tilde{p}) = (\tilde{p}-1)w(\tilde{p})$, $\ell_2'(\tilde{p}) = \tilde{p}w(\tilde{p})$ and hence $\ell_1''(\tilde{p}) = w(\tilde{p}) + (\tilde{p}-1)w'(\tilde{p})$ and $\ell_2''(\tilde{p}) = w(\tilde{p}) + \tilde{p}w'(\tilde{p})$. Substituting these expressions into (5) gives

$$\eta_\ell = \inf_{\tilde{p}\in(0,1)} \frac{(\tilde{p}-1)w(\tilde{p})[w(\tilde{p}) + \tilde{p}w'(\tilde{p})] - [w(\tilde{p}) + (\tilde{p}-1)w'(\tilde{p})]\tilde{p}w(\tilde{p})}{(\tilde{p}-1)w(\tilde{p})\tilde{p}w(\tilde{p})[\tilde{p}w(\tilde{p}) - (\tilde{p}-1)w(\tilde{p})]} = \inf_{\tilde{p}\in(0,1)} \frac{1}{\tilde{p}(1-\tilde{p})w(\tilde{p})}.$$

Observing that $\underline{L}_{\log}(p) = -p_1 \ln p_1 - p_2 \ln p_2$ we have $\underline{\tilde{L}}_{\log}(\tilde{p}) = -\tilde{p}\ln\tilde{p} - (1-\tilde{p})\ln(1-\tilde{p})$ and thus $\underline{\tilde{L}}_{\log}''(\tilde{p}) = \frac{-1}{\tilde{p}(1-\tilde{p})}$ and so $w_{\log}(\tilde{p}) = \frac{1}{\tilde{p}(1-\tilde{p})}$. Thus

$$\boxed{\eta_\ell = \inf_{\tilde{p}\in(0,1)} \frac{w_{\log}(\tilde{p})}{w_\ell(\tilde{p})} = \inf_{\tilde{p}\in(0,1)} \frac{\underline{\tilde{L}}_{\log}''(\tilde{p})}{\underline{\tilde{L}}''(\tilde{p})}.} \tag{8}$$

That is, the mixability constant of binary proper losses is the minimal ratio of the second derivatives of the Bayes risks for log loss and the loss in question. The rest of this paper is devoted to the generalisation of (8) to the multiclass case. That there is a relationship between Bayes risk and mixability was also pointed out (in a less explicit form) by Kalnishkan et al. (2004).

By substituting $w_\ell(\tilde{p}) = \frac{\ell_1'(\tilde{p})}{\tilde{p}-1}$ and $w_{\log}(\tilde{p}) = \frac{1}{\tilde{p}(1-\tilde{p})}$ into (8), one obtains an expression to compute η_ℓ that is simpler than (5):

$$\boxed{\frac{-1}{\eta_\ell} = \inf_{\tilde{p}\in(0,1)} \tilde{p}\ell_1'(\tilde{p}).} \tag{9}$$

This result also generalizes to the multiclass case; see Corollary 14.

4.2 Mixability and the Concavity of the Function f_η

Our aim is to relate mixability of a loss to the curvature of its Bayes risk surface. Since mixability is equivalent to concavity of the function f_η, which maps the first $n-1$ coordinates of Φ_η to the n-th coordinate, we will start by giving an explicit expression for f_η. We will assume throughout that the loss ℓ is strictly proper and continuous on Δ^n.

It is convenient to introduce an auxiliary function $\tau_\eta : \tilde{\Delta}^n \to [0,1]^{n-1}$ as

$$\tau_\eta(\tilde{p}) := E_\eta(\tilde{\ell}(\tilde{p})) = \left(e^{-\eta\ell_1(\tilde{p})}, \ldots, e^{-\eta\ell_{n-1}(\tilde{p})}\right), \tag{10}$$

which maps a distribution $\tilde{p}$ to the first $n-1$ coordinates of an element in Φ_η. The range of τ_η will be denoted $\tilde{\Phi}_\eta$ (see Figure 1). In addition, let the projection $\Pi_\Phi \colon \Phi_\eta \to \tilde{\Phi}_\eta$ map any element of $\phi \in \Phi_\eta$ to its first $n-1$ coordinates $(\phi_1, \ldots, \phi_{n-1})$. Then under our assumptions, all the maps we have defined are well-behaved:

Lemma 3 *Let ℓ be a continuous, strictly proper loss. Then for $\eta > 0$ all functions in Figure 1 are continuous and invertible with continuous inverse.*

Proof Lemma 1 already covers most of the functions. Given that E_η satisfies the required properties, they can be derived for the remaining functions by writing them as a composition of functions for which the properties are known. For example, $\tau_\eta = E_\eta \circ \tilde{\ell}$ is a composition of two continuous and invertible functions, which each have a continuous inverse. ■

It follows that, under the conditions of the lemma, the function $f_\eta \colon \tilde{\Phi}_\eta \to [0,1]$ may be defined as

$$f_\eta(\tilde{\phi}) = e^{-\eta\ell_n\left(\tau_\eta^{-1}(\tilde{\phi})\right)} \tag{11}$$

and is continuous. Moreover, as $\tilde{\Phi}_\eta$ (the domain of f_η) is the preimage under τ^{-1} of the closed set $\tilde{\Delta}^n$, continuity of τ^{-1} implies that $\tilde{\Phi}_\eta$ is closed as well. However, continuity implies that we may restrict attention to the interiors of $\tilde{\Phi}_\eta$ and of the probability simplex:

Lemma 4 *Let ℓ be a continuous, strictly proper loss. Then, for $\eta > 0$, f_η is concave if and only if it is concave on the interior* $\operatorname{int}(\tilde{\Phi}_\eta)$ *of its domain. Furthermore this set corresponds to a subset of the interior of the simplex:* $\tau_\eta^{-1}(\operatorname{int}(\tilde{\Phi}_\eta)) \subseteq \operatorname{int}(\tilde{\Delta}^n) = \Pi_\Delta(\operatorname{rel}\operatorname{int}(\Delta^n))$.

Proof The restriction to $\operatorname{int}(\tilde{\Phi}_\eta)$ follows trivially from continuity of f_η. The set $\tau_\eta^{-1}(\operatorname{int}(\tilde{\Phi}_\eta))$ is the preimage under τ_η of the open set $\operatorname{int}(\tilde{\Phi}_\eta)$. Since τ_η is continuous, it follows that this set must also be open and hence be a subset of the interior of $\tilde{\Delta}^n$. ■

4.3 Relating Concavity of f_η to the Hessian of $\underline{L}$

The aim of this subsection is to express the Hessian of f_η in terms of the Bayes risk of the loss function defining f_η. We first note that a twice differentiable function $f : X \to \mathbb{R}$ defined on $X \subseteq \mathbb{R}^{n-1}$ is concave if and only if its Hessian at x, $\mathsf{H}f(x)$, is negative semi-definite for all $x \in X$ (Hiriart-Urruty and Lemaréchal, 1993). The argument that follows consists of repeated applications of the chain and inverse rules for Hessians to compute $\mathsf{H}f_\eta$.

We start the analysis by considering the η-exponential operator, used in the definition of τ (10):

Lemma 5 *Suppose $\eta > 0$. Then the derivatives of E_η and E_η^{-1} are*

$$\mathsf{D}E_\eta(\lambda) = -\eta \operatorname{diag}(E_\eta(\lambda)) \quad \textit{and} \quad \mathsf{D}E_\eta^{-1}(\phi) = -\eta^{-1}\left[\operatorname{diag}(\phi)\right]^{-1}.$$

And the Hessian of E_η^{-1} *is*

$$\mathsf{H}E_\eta^{-1}(\phi) = \frac{1}{\eta}\begin{bmatrix} \operatorname{diag}(\phi_1^{-2}, 0, \ldots, 0) \\ \vdots \\ \operatorname{diag}(0, \ldots, 0, \phi_n^{-2}) \end{bmatrix}. \tag{12}$$

If $\eta = 1$ *and* $\ell = \ell_{\log} = p \mapsto -(\ln p_1, \ldots, \ln p_n)'$ *is the log loss, then the map* τ_1 *is the identity map (i.e.,* $\tilde{\phi} = \tau_1(\tilde{p}) = \tilde{p}$*) and* $E_1^{-1}(\tilde{p}) = \tilde{\ell}_{\log}(\tilde{p})$ *is the (projected) log loss.*

Proof The derivatives follow immediately from the definitions. By (24) the Hessian $\mathsf{H}E_\eta^{-1}(\phi) = \mathsf{D}\left(\mathsf{D}E_\eta^{-1}(\phi)\right)$ and so

$$\mathsf{H}E_\eta^{-1}(\phi) = \mathsf{D}\left(\left(-\frac{1}{\eta}\left[\operatorname{diag}(\phi)\right]^{-1}\right)'\right) = -\frac{1}{\eta}\mathsf{D}\operatorname{diag}(\phi_i^{-1})_{i=1}^n.$$

Let $h(\phi) = \operatorname{diag}(\phi_i^{-1})_{i=1}^n$. We have

$$\mathsf{D}h(\phi) = \mathsf{D}\operatorname{vec} h(\phi) = \begin{bmatrix} \operatorname{diag}(-\phi_1^{-2}, 0, \ldots, 0) \\ \vdots \\ \operatorname{diag}(0, \ldots, 0, -\phi_n^{-2}) \end{bmatrix}.$$

The result for $\eta = 1$ and $\ell_{\log}$ follows from $\tau_1(\tilde{p}) = E_1(\tilde{\ell}(\tilde{p})) = (e^{-1\cdot - \ln \tilde{p}_1}, \ldots, e^{-1\cdot - \ln \tilde{p}_{n-1}})'$. ■

Next we turn our attention to other components of f_η. Using the stationarity condition and invertibility of ℓ from Lemma 1, simple expressions can be derived for the Jacobian and Hessian of the projected Bayes risk $\underline{\tilde{L}}(\tilde{p}) := \underline{L}(\Pi_\Delta^{-1}(\tilde{p}))$:

Lemma 6 *Suppose the loss* ℓ *satisfied Condition A. Take* $\tilde{p} \in \operatorname{int}(\tilde{\Delta}^n)$*, and let* $y(\tilde{p}) := -\tilde{p}/p_n(\tilde{p})$*. Then*

$$Y(\tilde{p}) := -p_n(\tilde{p})\mathsf{D}y(\tilde{p}) = \left(I_{n-1} + \frac{1}{p_n}\tilde{p}\mathbb{1}_{n-1}'\right)$$

is invertible for all $\tilde{p}$*, and*

$$\mathsf{D}\ell_n(\tilde{p}) = y(\tilde{p})' \cdot \mathsf{D}\tilde{\ell}(\tilde{p}). \tag{13}$$

The projected Bayes risk function $\underline{\tilde{L}}(\tilde{p})$ *satisfies*

$$\mathsf{D}\underline{\tilde{L}}(\tilde{p}) = \tilde{\ell}(\tilde{p})' - \ell_n(\tilde{p})\mathbb{1}_{n-1}' \tag{14}$$

$$\text{and} \quad \mathsf{H}\underline{\tilde{L}}(\tilde{p}) = Y(\tilde{p})' \cdot \mathsf{D}\tilde{\ell}(\tilde{p}). \tag{15}$$

Furthermore, the matrix $\mathsf{H}\underline{\tilde{L}}(\tilde{p})$ *is negative definite and invertible for all* $\tilde{p}$*, and when* $\ell = \ell_{\log}$ *is the log loss*

$$\mathsf{H}\underline{\tilde{L}}_{\log}(\tilde{p}) = -Y(\tilde{p})' \cdot \left[\operatorname{diag}(\tilde{p})\right]^{-1}. \tag{16}$$

Proof The stationarity condition (Lemma 1) guarantees that $p'\mathsf{D}\ell(\tilde{p}) = 0_{n-1}$ for all $p \in \operatorname{rel\,int}(\Delta^n)$. This is equivalent to $\tilde{p}'\mathsf{D}\tilde{\ell}(\tilde{p}) + p_n(\tilde{p})\mathsf{D}\ell_n(\tilde{p}) = 0_{n-1}$, which can be rearranged to obtain (13).

By the product rule $\mathsf{D}a'b = (\mathsf{D}a')b + a'(\mathsf{D}b)$, we obtain

$$\begin{aligned}\mathsf{D}y(\tilde{p}) &= -\tilde{p}\mathsf{D}[p_n(\tilde{p})^{-1}] - [p_n(\tilde{p})^{-1}]\mathsf{D}\tilde{p} \\ &= \tilde{p}[p_n(\tilde{p})^{-2}]\mathsf{D}p_n(\tilde{p}) - [p_n(\tilde{p})^{-1}]I_{n-1} \\ &= -\tilde{p}[p_n(\tilde{p})^{-2}]\mathbb{1}'_{n-1} - [p_n(\tilde{p})^{-1}]I_{n-1} \\ &= -\frac{1}{p_n(\tilde{p})}\left[I_{n-1} + \frac{1}{p_n(\tilde{p})}\tilde{p}\mathbb{1}'_{n-1}\right],\end{aligned}$$

since $p_n(\tilde{p}) = 1 - \sum_{i\in[n-1]}\tilde{p}_i$ implies $\mathsf{D}p_n(\tilde{p}) = -\mathbb{1}'_{n-1}$. This establishes that $Y(\tilde{p}) = I_{n-1} + \frac{1}{p_n(\tilde{p})}\tilde{p}\mathbb{1}'_{n-1}$. That this matrix is invertible can be easily checked since

$$(I_{n-1} - \tilde{p}\mathbb{1}'_{n-1})(I_{n-1} + \frac{1}{p_n(\tilde{p})}\tilde{p}\mathbb{1}'_{n-1}) = I_{n-1}$$

by expanding and noting $\tilde{p}\mathbb{1}'_{n-1}\tilde{p}\mathbb{1}'_{n-1} = (1-p_n)\tilde{p}\mathbb{1}'_{n-1}$.

The Bayes risk is $\tilde{\underline{L}}(\tilde{p}) = \tilde{p}'\tilde{\ell}(\tilde{p}) + p_n(\tilde{p})\ell_n(\tilde{p})$. Taking the derivative and using the product rule gives

$$\begin{aligned}\mathsf{D}\tilde{\underline{L}}(\tilde{p}) &= \mathsf{D}\left[\tilde{p}'\tilde{\ell}(\tilde{p})\right] + \mathsf{D}\left[p_n(\tilde{p})\ell_n(\tilde{p})\right] \\ &= \tilde{\ell}(\tilde{p}) + \tilde{p}'\mathsf{D}\tilde{\ell}(\tilde{p}) + \left[\mathsf{D}p_n(\tilde{p})\right]\ell_n(\tilde{p}) + p_n(\tilde{p})\mathsf{D}\ell_n(\tilde{p}) \\ &= \tilde{\ell}(\tilde{p}) - p_n(\tilde{p})\mathsf{D}\ell_n(\tilde{p}) - \ell_n(\tilde{p})\mathbb{1}'_{n-1} + p_n(\tilde{p})\mathsf{D}\ell_n(\tilde{p})\end{aligned}$$

by (13). Thus, $\mathsf{D}\tilde{\underline{L}}(\tilde{p}) = \tilde{\ell}(\tilde{p})' - \ell_n(\tilde{p})\mathbb{1}'_{n-1}$, establishing (14).

Equation 15 is obtained by taking derivatives once more and using (13) again, yielding

$$\mathsf{H}\tilde{\underline{L}}(\tilde{p}) = \mathsf{D}\left(\left(\mathsf{D}\tilde{\underline{L}}(\tilde{p})\right)'\right) = \mathsf{D}\tilde{\ell}(\tilde{p}) - \mathbb{1}_{n-1}\cdot\mathsf{D}\ell_n(\tilde{p}) = \left(I_{n-1} + \frac{1}{p_n}\mathbb{1}_{n-1}\tilde{p}'\right)\mathsf{D}\tilde{\ell}(\tilde{p})$$

as required. Now $\tilde{\underline{L}}(\tilde{p}) = \underline{L}(p_1,\ldots,p_{n-1},p_n(\tilde{p})) = \underline{L}(p_1,\ldots,p_{n-1},1-\sum_{i=1}^{n-1}p_i) = \underline{L}(C(\tilde{p}))$ where C is affine. Since $p \mapsto \underline{L}(p)$ is strictly concave (Lemma 1) it follows (Hiriart-Urruty and Lemaréchal, 1993) that $\tilde{p} \mapsto \tilde{\underline{L}}(\tilde{p})$ is also strictly concave and thus $\mathsf{H}\tilde{\underline{L}}(\tilde{p})$ is negative definite. It is invertible since we have shown $Y(\tilde{p})$ is invertible and $\mathsf{D}\tilde{\ell}$ is invertible by the inverse function theorem and the invertibility of $\tilde{\ell}$ (Lemma 1).

Finally, Equation 16 holds since Lemma 5 gives us $E_1^{-1} = \tilde{\ell}_{\log}$ so (15) specialises to $\mathsf{H}\tilde{\underline{L}}_{\log}(\tilde{p}) = Y(\tilde{p})'\cdot\mathsf{D}\tilde{\ell}_{\log}(\tilde{p}) = Y(\tilde{p})'\cdot\mathsf{D}E_1^{-1}(\tilde{p}) = -Y(\tilde{p})'\cdot[\mathrm{diag}(\tilde{p})]^{-1}$, also by Lemma 5. ■

4.4 Completion of the Argument

Recall that our aim is to compute the Hessian of the function describing the boundary of the η-exponentiated superprediction set and determine when it is negative semi-definite. The boundary is described by the function f_η which can be written as the composition $h_\eta \circ g_\eta$ where $h_\eta(z) := e^{-\eta z}$ and $g_\eta(\tilde{\phi}) := \ell_n\left(\tau_\eta^{-1}(\tilde{\phi})\right)$. The Hessian of f_η can be expanded in terms of g_η using the chain rule for the Hessian (Theorem 21) as follows.

Lemma 7 *Suppose the loss ℓ satisfies Condition A and $\eta > 0$. Then for all $\tilde{\phi} \in \operatorname{int}(\tilde{\Phi})$, the Hessian of f_η at $\tilde{\phi}$ is*

$$\mathsf{H} f_\eta(\tilde{\phi}) = \eta e^{-\eta g_\eta(\tilde{\phi})} \Gamma_\eta(\tilde{\phi}),$$

where $\Gamma_\eta(\tilde{\phi}) := \eta \mathsf{D} g_\eta(\tilde{\phi})' \cdot \mathsf{D} g_\eta(\tilde{\phi}) - \mathsf{H} g_\eta(\tilde{\phi})$. Furthermore, for $\eta > 0$ the negative semi-definiteness of $\mathsf{H} f_\eta(\tilde{\phi})$ (and thus the concavity of f_η) is equivalent to the negative semi-definiteness of $\Gamma_\eta(\tilde{\phi})$.

Proof Using $f := f_\eta$ and $g := g_\eta$ temporarily and letting $z = g(\tilde{\phi})$, the chain rule for H gives

$$\begin{aligned}
\mathsf{H} f(\tilde{\phi}) &= \left(I_1 \otimes \mathsf{D} g(\tilde{\phi})'\right) \cdot (\mathsf{H} h_\eta(z)) \cdot \mathsf{D} g(\tilde{\phi}) + (\mathsf{D} h_\eta(z) \otimes I_{n-1}) \cdot \mathsf{H} g(\tilde{\phi}) \\
&= \eta^2 e^{-\eta z} \mathsf{D} g(\tilde{\phi})' \cdot \mathsf{D} g(\tilde{\phi}) - \eta e^{-\eta z} \mathsf{H} g(\tilde{\phi}) \\
&= \eta e^{-\eta g(\tilde{\phi})} \left[\eta \mathsf{D} g(\tilde{\phi})' \cdot \mathsf{D} g(\tilde{\phi}) - \mathsf{H} g(\tilde{\phi})\right],
\end{aligned}$$

since $\alpha \otimes A = \alpha A$ for scalar α and matrix A and $\mathsf{D} h_\eta(z) = \mathsf{D}[\exp(-\eta z)] = -\eta e^{-\eta z}$ so $\mathsf{H} h(z) = \eta^2 e^{-\eta z}$. Whether $\mathsf{H} f \preccurlyeq 0$ depends only on Γ_η since $\eta e^{-\eta g(\tilde{\phi})}$ is positive for all $\eta > 0$ and $\tilde{\phi}$. ■

We proceed to compute the derivative and Hessian of g_η:

Lemma 8 *Suppose ℓ satisfies Condition A. For $\eta > 0$ and $\tilde{\phi} \in \operatorname{int}(\tilde{\Phi}_\eta)$, let $\lambda := E_\eta^{-1}(\tilde{\phi})$ and $\tilde{p} := \tilde{\ell}^{-1}(\lambda)$. Then*

$$\begin{aligned}
&\mathsf{D} g_\eta(\tilde{\phi}) = y(\tilde{p})' A_\eta(\tilde{\phi}) \qquad (17) \\
\textit{and} \quad &\mathsf{H} g_\eta(\tilde{\phi}) = -\frac{1}{p_n(\tilde{p})} A_\eta(\tilde{\phi})' \cdot \left[\eta \operatorname{diag}(\tilde{p}) + Y(\tilde{p}) \cdot \left[\mathsf{H} \underline{\tilde{L}}(\tilde{p})\right]^{-1} \cdot Y(\tilde{p})'\right] \cdot A_\eta(\tilde{\phi}),
\end{aligned}$$

where $A_\eta(\tilde{\phi}) := \mathsf{D} E_\eta^{-1}(\tilde{\phi})$.

Proof By definition, $g_\eta(\tilde{\phi}) := \ell_n(\tau_\eta^{-1}(\tilde{\phi}))$. Since $\tau_\eta^{-1} = \tilde{\ell}^{-1} \circ E_\eta^{-1}$ we have $g_\eta = \ell_n \circ \tilde{\ell}^{-1} \circ E_\eta^{-1}$. Thus, by the chain rule, Equation 13 from Lemma 6, and the inverse function theorem, we obtain

$$\mathsf{D} g_\eta(\tilde{\phi}) = \mathsf{D} \ell_n(\tilde{p}) \cdot \mathsf{D} \tilde{\ell}^{-1}(\lambda) \cdot \mathsf{D} E_\eta^{-1}(\tilde{\phi}) = y(\tilde{p})' \mathsf{D} \tilde{\ell}(\tilde{p}) \cdot \left[\mathsf{D} \tilde{\ell}(\tilde{p})\right]^{-1} \cdot \left[\mathsf{D} E_\eta^{-1}(\tilde{\phi})\right] = y(\tilde{p})' A_\eta(\tilde{\phi})$$

yielding (17). Since $\tilde{p} = \tau_\eta^{-1}(\tilde{\phi})$ and $\mathsf{H} g_\eta = \mathsf{D}((\mathsf{D} g_\eta)')$ (see (24)), the chain and product rules give

$$\begin{aligned}
\mathsf{H} g_\eta(\tilde{\phi}) &= \mathsf{D}\left[\left(\mathsf{D} E_\eta^{-1}(\tilde{\phi})\right)' \cdot y\left(\tau_\eta^{-1}(\tilde{\phi})\right)\right] \\
&= \left(y(\tau_\eta^{-1}(\tilde{\phi}))' \otimes I_{n-1}\right) \cdot \mathsf{D}\left(\mathsf{D} E_\eta^{-1}(\tilde{\phi})'\right) + \left(I_1 \otimes (\mathsf{D} E_\eta^{-1}(\tilde{\phi}))'\right) \cdot \mathsf{D}\left(y\left(\tau_\eta^{-1}(\tilde{\phi})\right)\right) \\
&= \left(y(\tilde{p})' \otimes I_{n-1}\right) \cdot \mathsf{H} E_\eta^{-1}(\tilde{\phi}) + \left(\mathsf{D} E_\eta^{-1}(\tilde{\phi})\right)' \cdot \mathsf{D} y(\tilde{p}) \cdot \mathsf{D} \tau_\eta^{-1}(\tilde{\phi}) \\
&= -\frac{\eta}{p_n(\tilde{p})} A_\eta(\tilde{\phi}) \cdot \operatorname{diag}(\tilde{p}) \cdot A_\eta(\tilde{\phi}) + A_\eta(\tilde{\phi})' \cdot \mathsf{D} y(\tilde{p}) \cdot \mathsf{D} \tau_\eta^{-1}(\tilde{\phi}). \qquad (18)
\end{aligned}$$

The first summand in (18) is due to (12) and the fact that

$$\begin{aligned}(y\otimes I_{n-1})\cdot \mathsf{H}E_\eta^{-1}(\tilde\phi) &= \frac{1}{\eta}[y_1 I_{n-1},\dots,y_{n-1}I_{n-1}]\cdot\begin{bmatrix}\operatorname{diag}(\phi_1^{-2},0,\dots,0)\\ \vdots \\ \operatorname{diag}(0,\dots,0,\phi_{n-1}^{-2})\end{bmatrix}\\ &= \frac{1}{\eta}\sum_{i=1}^{n-1} y_i\cdot I_{n-1}\cdot\operatorname{diag}(0,\dots,0,\phi_i^{-2},0,\dots,0)\\ &= \frac{1}{\eta}\operatorname{diag}(y_i\phi_i^{-2})_{i=1}^{n-1}\\ &= \frac{-\eta}{p_n(\tilde p)}A_\eta(\tilde\phi)'\cdot\operatorname{diag}(\tilde p)\cdot A_\eta(\tilde\phi).\end{aligned}$$

The last equality holds because $A_\eta(\tilde\phi)'\cdot A_\eta(\tilde\phi)=\eta^{-2}\operatorname{diag}(\tilde\phi_i^{-2})_{i=1}^{n-1}$ by Lemma 5, the definition of $y(\tilde p)=-[p_n(\tilde p)]^{-1}\tilde p$, and because all the matrices are diagonal and thus commute.

The second summand in (18) reduces by $\mathsf{D}y(\tilde p)=-\frac{1}{p_n(\tilde p)}Y(\tilde p)$ from Lemma 6 and $\tau_\eta=E_\eta\circ\tilde\ell$:

$$\begin{aligned}\mathsf{D}\tau_\eta^{-1}(\tilde\phi) &= \left[\mathsf{D}E_\eta(\lambda)\cdot\mathsf{D}\tilde\ell(\tilde p)\right]^{-1}\\ &= \left[\mathsf{D}E_\eta(\lambda)\cdot(Y(\tilde p)')^{-1}\cdot\mathsf{H}\underline{\tilde L}(\tilde p)\right]^{-1}\\ &= \left[\mathsf{H}\underline{\tilde L}(\tilde p)\right]^{-1}\cdot Y(\tilde p)'\cdot\mathsf{D}E_\eta^{-1}(\lambda).\end{aligned}$$

Substituting these into (18) gives

$$\mathsf{H}g_\eta(\tilde\phi)=-\frac{\eta}{p_n(\tilde p)}A_\eta(\tilde\phi)\cdot\operatorname{diag}(\tilde p)\cdot A_\eta(\tilde\phi)-\frac{1}{p_n(\tilde p)}A_\eta(\tilde\phi)'\cdot Y(\tilde p)\cdot\left[\mathsf{H}\underline{\tilde L}(\tilde p)\right]^{-1}\cdot Y(\tilde p)'\cdot A_\eta(\tilde\phi),$$

which can be factored into the required result. ■

We can now use the last two lemmata to express the function Γ_η in terms of the Hessian of the Bayes risk functions for the specified loss ℓ and the log loss.

Lemma 9 *Suppose a loss ℓ satisfies Condition A. Then for $\eta>0$ the matrix-valued function Γ_η satisfies the following: for all $\tilde\phi\in\operatorname{int}(\tilde\Phi_\eta)$ and $\tilde p=\tau_\eta^{-1}(\tilde\phi)$,*

$$\Gamma_\eta(\tilde\phi)=\frac{1}{p_n}A_\eta(\tilde\phi)'\cdot Y(\tilde p)\cdot\left[\left[\mathsf{H}\underline{\tilde L}(\tilde p)\right]^{-1}-\eta\left[\mathsf{H}\underline{\tilde L}_{\log}(\tilde p)\right]^{-1}\right]\cdot Y(\tilde p)'\cdot A_\eta(\tilde\phi),\tag{19}$$

and is negative semi-definite if and only if $R(\eta,\ell,\tilde p):=\left[\mathsf{H}\underline{\tilde L}(\tilde p)\right]^{-1}-\eta\left[\mathsf{H}\underline{\tilde L}_{\log}(\tilde p)\right]^{-1}$ is negative semi-definite.

Proof Substituting the values of $\mathsf{D}g_\eta$ and $\mathsf{H}g_\eta$ from Lemma 8 into the definition of Γ_η from Lemma 7 and then using Lemma 5 and the definition of $y(\tilde p)$, we obtain

$$\begin{aligned}\Gamma_\eta(\tilde\phi) &= \eta A_\eta(\tilde\phi)'\cdot y(\tilde p)\cdot y(\tilde p)'\cdot A_\eta(\tilde\phi)\\ &\quad+\frac{1}{p_n(\tilde p)}A_\eta(\tilde\phi)'\cdot\left[\eta\operatorname{diag}(\tilde p)+Y(\tilde p)\cdot\left[\mathsf{H}\underline{\tilde L}(\tilde p)\right]^{-1}\cdot Y(\tilde p)'\right]\cdot A_\eta(\tilde\phi)\\ &= \frac{1}{p_n}A_\eta(\tilde\phi)'\cdot\left[\eta\frac{1}{p_n}\tilde p\cdot\tilde p'+\eta\operatorname{diag}(\tilde p)+Y(\tilde p)\cdot\left[\mathsf{H}\underline{\tilde L}(\tilde p)\right]^{-1}\cdot Y(\tilde p)'\right]\cdot A_\eta(\tilde\phi).\end{aligned}\tag{20}$$

Using Lemma 6 we then see that

$$\begin{aligned}
-Y(\tilde{p}) \cdot \left[\mathsf{H}\underline{\tilde{L}}_{\log}(\tilde{p})\right]^{-1} \cdot Y(\tilde{p})' &= -Y(\tilde{p}) \cdot \left[-Y(\tilde{p})' \operatorname{diag}(\tilde{p})^{-1}\right]^{-1} \cdot Y(\tilde{p})' \\
&= Y(\tilde{p}) \cdot \operatorname{diag}(\tilde{p}) \cdot (Y(\tilde{p})')^{-1} \cdot Y(\tilde{p})' \\
&= (I_{n-1} + \frac{1}{p_n} \mathbb{1}_{n-1} \tilde{p}') \cdot \operatorname{diag}(\tilde{p}) \\
&= \operatorname{diag}(\tilde{p}) + \frac{1}{p_n} \tilde{p} \cdot \tilde{p}'.
\end{aligned}$$

Substituting this for the appropriate terms in (20) gives

$$\Gamma_\eta(\tilde{\phi}) = \frac{1}{p_n} A_\eta(\tilde{\phi})' \cdot \left[Y(\tilde{p}) \cdot \left[\mathsf{H}\underline{\tilde{L}}(\tilde{p})\right]^{-1} \cdot Y(\tilde{p})' - \eta Y(\tilde{p}) \cdot \left[\mathsf{H}\underline{\tilde{L}}_{\log}(\tilde{p})\right]^{-1} \cdot Y(\tilde{p})' \right] \cdot A_\eta(\tilde{\phi}),$$

which equals (19).

Since $\Gamma_\eta = [p_n]^{-1} BRB'$ where $B = A_\eta(\tilde{\phi})' Y(\tilde{p})$ and $R = R(\eta, \ell, \tilde{p})$ the definition of negative semi-definiteness and the positivity of p_n means we need to show that $\forall x : x'\Gamma_\eta x \leq 0 \iff \forall y : y'Ry \leq 0$. It suffices to show that B is invertible, since we can let $y = Bx$ to establish the equivalence. The matrix $A_\eta(\tilde{\phi})$ is invertible since, by definition, $A_\eta(\tilde{\phi}) = \mathsf{D}E_\eta^{-1}(\tilde{\phi}) = -\eta^{-1}[\operatorname{diag}(\tilde{\phi})]^{-1}$ by Lemma 5 and so has matrix inverse $-\eta \operatorname{diag}(\tilde{\phi})$. The matrix $Y(\tilde{p})$ is invertible by Lemma 8. Thus, B is invertible because it is the product of two invertible matrices. ■

The above arguments result in a characterisation of the concavity of the function f_η (via its Hessian)—and hence the convexity of the η-exponentiated superprediction set—in terms of the Hessian of the Bayes risk function of the loss ℓ and the log loss $\ell_{\log}$. As in the binary case (cf. (8)), this means we are now able to specify the mixability constant η_ℓ in terms of the curvature $\mathsf{H}\underline{\tilde{L}}$ of the Bayes risk for ℓ relative to the curvature $\mathsf{H}\underline{\tilde{L}}_{\log}$ of the Bayes risk for log loss.

Theorem 10 *Suppose a loss ℓ satisfies Condition A. Let $\underline{\tilde{L}}(\tilde{p})$ be the Bayes risk for ℓ and $\underline{\tilde{L}}_{\log}(\tilde{p})$ be the Bayes risk for the log loss. Then the following statements are equivalent:*

(i.) *ℓ is η-mixable;*

(ii.) *$\eta\mathsf{H}\underline{\tilde{L}}(\tilde{p}) \succcurlyeq \mathsf{H}\underline{\tilde{L}}_{\log}(\tilde{p})$ for all $\tilde{p} \in \operatorname{int}(\tilde{\Delta}^n)$;*

(iii.) *$\eta\underline{L}(p) - \underline{L}_{\log}(p)$ is convex on $\operatorname{rel\,int}(\Delta^n)$;*

(iv.) *$\eta\underline{\tilde{L}}(\tilde{p}) - \underline{\tilde{L}}_{\log}(\tilde{p})$ is convex on $\operatorname{int}(\tilde{\Delta}^n)$.*

Note that the largest η that satisfies any one of (i)–(iv) is the mixability constant for the loss. For example,

$$\eta_\ell = \max\{\eta \geq 0 \colon \forall \tilde{p} \in \operatorname{int}(\tilde{\Delta}^n)\ ,\ \eta\mathsf{H}\underline{\tilde{L}}(\tilde{p}) \succcurlyeq \mathsf{H}\underline{\tilde{L}}_{\log}(\tilde{p})\}.$$

Proof The case $\eta = 0$ is trivial, so suppose $\eta > 0$. Then by Lemmas 7 and 9 we know $\mathsf{H}f_\eta(\tilde{p}) \preccurlyeq 0 \iff R(\eta, \ell, \tilde{p}) \preccurlyeq 0$. By Lemma 6, $\mathsf{H}\underline{\tilde{L}}(\tilde{p}) \prec 0$ and $\mathsf{H}\underline{\tilde{L}}_{\log}(\tilde{p}) \prec 0$ for all $\tilde{p}$ and so we can use the fact that for positive definite matrices A and B we have $A \succcurlyeq B \iff B^{-1} \succcurlyeq A^{-1}$ (Horn and Johnson, 1985, Corollary 7.7.4). This means $R(\eta, \ell, \tilde{p}) \preccurlyeq 0 \iff \mathsf{H}\underline{\tilde{L}}(\tilde{p})^{-1} \preccurlyeq \eta\mathsf{H}\underline{\tilde{L}}_{\log}(\tilde{p})^{-1} \iff \eta^{-1}\mathsf{H}\underline{\tilde{L}}_{\log}(\tilde{p}) \preccurlyeq$

$\mathsf{H}\tilde{\underline{L}}(\tilde{p}) \iff \eta\mathsf{H}\tilde{\underline{L}}(\tilde{p}) \succcurlyeq \mathsf{H}\tilde{\underline{L}}_{\log}(\tilde{p})$. Therefore f_η is concave at $\tilde{p}$ if and only if $\eta\mathsf{H}\tilde{\underline{L}}(\tilde{p}) \succcurlyeq \mathsf{H}\tilde{\underline{L}}_{\log}(\tilde{p})$. Since concavity of f_η was equivalent to η-mixability, this establishes equivalence of (i) and (ii).

Since $\eta\mathsf{H}\tilde{\underline{L}}(\tilde{p}) \succcurlyeq \mathsf{H}\tilde{\underline{L}}_{\log}(\tilde{p}) \iff \mathsf{H}\big(\eta\tilde{\underline{L}}(\tilde{p}) - \tilde{\underline{L}}_{\log}(\tilde{p})\big) \succcurlyeq 0$, equivalence of (ii) and (iv) follows from the fact that positive semi-definiteness of the Hessian of a function on an open set is equivalent to convexity of the function (Hiriart-Urruty and Lemaréchal, 1993). Finally, equivalence of (iv) and (iii) follows by linearity of the map $p_n(\tilde{p}) = 1 - \sum_{i=1}^{n-1} \tilde{p}_i$. ■

The lemma allows one to derive η-mixability of an average of two η-mixable proper losses that satisfy its conditions:

Corollary 11 *Suppose ℓ_A and ℓ_B are two η-mixable losses that satisfy Condition A. Then, for any $\lambda \in (0,1)$, the loss $\ell = (1-\lambda)\ell_A + \lambda\ell_B$ is also η-mixable.*

Proof Clearly ℓ is continuous and continuously differentiable. And because properness of ℓ_A and ℓ_B implies that $\underline{L}_\ell(p) = (1-\lambda)\underline{L}_{\ell_A}(p) + \lambda\underline{L}_{\ell_B}(p)$, it is also strictly proper. Thus Theorem 10 applies to ℓ, and we just need to verify that $\eta\underline{L}_\ell(p) - \underline{L}_{\log}(p)$ is convex. Noting that

$$\eta\underline{L}_\ell(p) - \underline{L}_{\log}(p) = (1-\lambda)\Big(\eta\underline{L}_{\ell_A}(p) - \underline{L}_{\log}(p)\Big) + \lambda\Big(\eta\underline{L}_{\ell_B}(p) - \underline{L}_{\log}(p)\Big)$$

is a convex combination of two convex functions, the result follows. ■

One may wonder which loss is the most mixable. In the following we derive a straight-forward result that shows the (perhaps unsurprising) answer is log loss. Let $e_i \in \Delta^n$ denote the point-mass on the i-th outcome. Then we call a proper loss *fair* if $L(e_i, e_i) = \underline{L}(e_i) = 0$ for all i (Reid and Williamson, 2011). That is, if one is certain that outcome i will occur and this is correct, then it is only fair if one incurs no loss. Any loss can be made fair by subtracting the unique affine function that interpolates $\{\underline{L}(e_i) \colon i \in [n]\}$ from its Bayes risk. This does not change the curvature of $\underline{L}$ and thus by Theorem 10 it has the same mixability constant (provided the conditions of the theorem are satisfied). We will call a proper loss *normalised* if it is fair and $\max_{p\in\Delta^n} \underline{L}(p) = 1$. If a fair proper loss is not normalised, one may normalise it by dividing the loss on all outcomes by $\max_{p\in\Delta^n} \underline{L}(p)$. This scales up the mixability constant by $\max_{p\in\Delta^n} \underline{L}(p)$. For example, log loss is fair, but in order to normalise it, one needs to divide by $\max_{p\in\Delta^n} \underline{L}_{\log}(p) = \log(n)$, and the mixability constant η_ℓ for the resulting loss is $\log(n)$.

Corollary 12 *Suppose a loss ℓ satisfies Condition A. Then, if ℓ is normalised and $\underline{L}(p)$ is continuous, it can only be η-mixable for $\eta \le \log(n)$. This bound is achieved if ℓ is the normalised log loss.*

Proof Since $\underline{L}(p)$ is continuous and has a compact domain, there exists a $p^* = \arg\max_{p\in\Delta^n} \underline{L}(p)$ that achieves its maximum, which is 1 by assumption. Now by Theorem 10, η-mixability implies convexity of $\eta\underline{L}(p) - \underline{L}_{\log}(p)$ on $\operatorname{int}(\Delta^n)$, which extends to convexity on Δ^n by continuity of $\underline{L}(p)$ and $\underline{L}_{\log}(p)$, and hence

$$\begin{aligned} 0 = \mathbb{E}_{i\sim p^*}\Big[\eta\underline{L}(e_i) - \underline{L}_{\log}(e_i)\Big] &\ge \eta\underline{L}(p^*) - \underline{L}_{\log}(p^*) = \eta - \underline{L}_{\log}(p^*) \\ \Rightarrow \eta \le \underline{L}_{\log}(p^*) &\le \underline{L}_{\log}\big(\tfrac{1}{n}, \ldots, \tfrac{1}{n}\big) = \log(n), \end{aligned}$$

where the first equality follows from fairness of ℓ and log loss, and the first inequality follows from Jensen's inequality. ■

The mixability constant can also be expressed in terms of the maximal eigenvalue of the "ratio" of the Hessian matrices for the Bayes risk for log loss and the loss in question. In the following, $\lambda_i(A)$ will denote the ith largest (possibly repeated) eigenvalue of the $n \times n$ symmetric matrix A. That is, $\lambda_{\min}(A) := \lambda_1(A) \leq \lambda_2(A) \leq \cdots \leq \lambda_n =: \lambda_{\max}(A)$ where each $\lambda_i(A)$ satisfies $|A - \lambda_i(A)I| = 0$.

Theorem 13 *Suppose a loss ℓ satisfies Condition A. Then its mixability constant is*

$$\boxed{\eta_\ell = \inf_{\tilde{p} \in \operatorname{int}(\tilde{\Delta}^n)} \lambda_{\max}\left((\mathsf{H}\underline{\tilde{L}}(\tilde{p}))^{-1} \cdot \mathsf{H}\underline{\tilde{L}}_{\log}(\tilde{p}) \right).} \tag{21}$$

Equation 21 reduces to (8) when $n = 2$ since the maximum eigenvalue of a 1×1 matrix is simply its single entry. Since the maximum eigenvalue of the Hessian of a function can be thought of as the "curvature", the above result justifies the title of the paper.

Proof For $\tilde{p} \in \operatorname{int}(\tilde{\Delta}^n)$, we define $C_\eta(\tilde{p}) := \eta \mathsf{H}\underline{\tilde{L}}(\tilde{p}) - \mathsf{H}\underline{\tilde{L}}_{\log}(\tilde{p})$ and $\rho(\tilde{p}) := \mathsf{H}\underline{\tilde{L}}(\tilde{p})^{-1} \cdot \mathsf{H}\underline{\tilde{L}}_{\log}(\tilde{p})$ and first show that zero is an eigenvalue of $C_\eta(\tilde{p})$ if and only if η is an eigenvalue of $\rho(\tilde{p})$. This can be seen since $\mathsf{H}\underline{\tilde{L}}(\tilde{p})$ is invertible (Lemma 6) so

$$\begin{aligned} &|C_\eta(\tilde{p}) - 0I| = 0 \iff |\eta \mathsf{H}\underline{\tilde{L}}(\tilde{p}) - \mathsf{H}\underline{\tilde{L}}_{\log}(\tilde{p})| = 0 \iff |\mathsf{H}\underline{\tilde{L}}(\tilde{p})^{-1}||\eta \mathsf{H}\underline{\tilde{L}}(\tilde{p}) - \mathsf{H}\underline{\tilde{L}}_{\log}(\tilde{p})| = 0 \\ &\iff \left|\mathsf{H}\underline{\tilde{L}}(\tilde{p})^{-1} \cdot \left[\eta \mathsf{H}\underline{\tilde{L}}(\tilde{p}) - \mathsf{H}\underline{\tilde{L}}_{\log}(\tilde{p})\right]\right| = 0 \iff |\eta I - \mathsf{H}\underline{\tilde{L}}(\tilde{p})^{-1} \cdot \mathsf{H}\underline{\tilde{L}}_{\log}(\tilde{p})| = 0. \end{aligned}$$

Since a symmetric matrix is positive semidefinite if and only if all its eigenvalues are non-negative it must be the case that if $\lambda_{\min}(C_\eta(\tilde{p})) \geq 0$ then $C_\eta(\tilde{p}) \succcurlyeq 0$ since every other eigenvalue is bigger than the minimum one. Conversely, if $C_\eta(\tilde{p}) \not\succcurlyeq 0$ then at least one eigenvalue must be negative, thus the smallest eigenvalue must be negative. Thus, $\lambda_{\min}(C_\eta(\tilde{p})) \geq 0 \iff C_\eta(\tilde{p}) \succcurlyeq 0$. Now define $\eta(\tilde{p}) := \sup\{\eta > 0 : C_\eta(\tilde{p}) \succcurlyeq 0\} = \sup\{\eta > 0 : \lambda_{\min}(C_\eta(\tilde{p})) \geq 0\}$. We show that for each $\tilde{p}$ the function $\eta \mapsto \lambda_{\min}(C_\eta(\tilde{p}))$ is continuous and only has a single root. First, continuity follows because the entries of $C_\eta(\tilde{p})$ are continuous in η for each $\tilde{p}$ and eigenvalues are continuous functions of their matrix's entries (Horn and Johnson, 1985, Appendix D). Second, as a function of its matrix arguments, the minimum eigenvalue $\lambda_{\min}$ is known to be concave (Magnus and Neudecker, 1999, §11.6). Thus, for any fixed $\tilde{p}$, its restriction to the convex set of matrices $\{C_\eta(\tilde{p}) : \eta > 0\}$ is also concave in its entries and so in η. Since $C_0(\tilde{p}) = -\mathsf{H}\underline{\tilde{L}}_{\log}(\tilde{p})$ is positive definite for every $\tilde{p}$ (Lemma 6) we have $\lambda_{\min}(C_0(\tilde{p})) > 0$ and so, by the concavity of the map $\eta \mapsto \lambda_{\min}(C_\eta(\tilde{p}))$, there can be only one $\eta > 0$ for which $\lambda_{\min}(C_\eta(\tilde{p})) = 0$ and by continuity it must be largest non-negative one, that is, $\eta(\tilde{p})$.

Thus

$$\eta(\tilde{p}) = \sup\{\eta > 0 : \lambda_{\min}(C_\eta(\tilde{p})) = 0\} = \sup\{\eta > 0 : \eta \text{ is an eigenvalue of } \rho(\tilde{p})\} = \lambda_{\max}(\rho(\tilde{p})).$$

Now let $\eta^* := \inf_{\tilde{p} \in \operatorname{int}(\tilde{\Delta}^n)} \eta(\tilde{p}) = \inf_{\tilde{p} \in \operatorname{int}(\tilde{\Delta}^n)} \lambda_{\max}(\rho(\tilde{p}))$. We now claim that $C_{\eta^*}(\tilde{p}) \succcurlyeq 0$ for all $\tilde{p}$ since if there was some $\tilde{q} \in \tilde{\Delta}^n$ such that $C_{\eta^*}(\tilde{q}) \not\succcurlyeq 0$ we would have $\eta(\tilde{q}) < \eta^*$ since $\eta \mapsto \lambda_{\min}(C_\eta(\tilde{q}))$ only has a single root—a contradiction. Thus, since we have shown η^* is the largest η such that $C_{\eta^*}(\tilde{p}) \succcurlyeq 0$ it must be η_ℓ, by Theorem 10, as required. ■

The following Corollary gives an expression for η_ℓ that is simpler than (21), generalising (9) from the binary case.

Corollary 14 *Suppose ℓ satisfies Condition A. Then its mixability constant satisfies*

$$\boxed{\frac{-1}{\eta_\ell} = \inf_{\tilde{p}\in\mathrm{int}(\tilde{\Delta}^n)} \lambda_{\max}\left(\mathrm{diag}(\tilde{p})\cdot \mathsf{D}\tilde{\ell}(\tilde{p})\right).} \tag{22}$$

Proof Theorem 13 combined with Lemma 6 allows us to write

$$\begin{aligned}
\eta_\ell &= \inf_{\tilde{p}\in\mathrm{int}\,\tilde{\Delta}^n} \lambda_{\max}\left(\left(Y(\tilde{p})'\cdot \mathsf{D}\tilde{\ell}(\tilde{p})\right)^{-1}\cdot\left(Y(\tilde{p})'\cdot \mathsf{D}\tilde{\ell}_{\log}(\tilde{p})\right)\right)\\
&= \inf_{\tilde{p}\in\mathrm{int}\,\tilde{\Delta}^n} \lambda_{\max}\left(\left(\mathsf{D}\tilde{\ell}(\tilde{p})\right)^{-1}\cdot \mathsf{D}\tilde{\ell}_{\log}(\tilde{p})\right)\\
&= \inf_{\tilde{p}\in\mathrm{int}\,\tilde{\Delta}^n} \lambda_{\max}\left(\left(\mathsf{D}\tilde{\ell}(\tilde{p})\right)^{-1}\cdot \mathrm{diag}(-1/p_i)_{i=1}^{n-1}\right).\\
&= -\sup_{\tilde{p}\in\mathrm{int}\,\tilde{\Delta}^n} \lambda_{\min}\left(\left(\mathsf{D}\tilde{\ell}(\tilde{p})\right)^{-1}\cdot \mathrm{diag}(1/p_i)_{i=1}^{n-1}\right)
\end{aligned}$$

and thus (22) follows since $\lambda_{\max}(A) = 1/\lambda_{\min}(A^{-1})$. ■

5. Mixability of the Brier Score

We will now apply the results from the previous section to show that the multiclass Brier score is mixable with mixability constant 1, as first proved by Vovk and Zhdanov (2009). The n-class Brier score is[1]

$$\ell_{\mathrm{Brier}}(y,\hat{p}) = \sum_{i=1}^{n}(\llbracket y_i = 1\rrbracket - \hat{p}_i)^2,$$

where $y\in\{0,1\}^n$ and $\hat{p}\in\Delta^n$. Thus

$$L_{\mathrm{Brier}}(p,\hat{p}) = \sum_{i=1}^{n}\mathbb{E}_{\mathsf{Y}\sim p}(\llbracket \mathsf{Y}_i = 1\rrbracket - \hat{p}_i)^2 = \sum_{i=1}^{n}(p_i - 2p_i\hat{p}_i + \hat{p}_i^2).$$

Hence $\underline{L}_{\mathrm{Brier}}(p) = L_{\mathrm{Brier}}(p,p) = \sum_{i=1}^n(p_i - 2p_ip_i + p_i^2) = 1 - \sum_{i=1}^n p_i^2$ since $\sum_{i=1}^n p_i = 1$, and $\tilde{\underline{L}}_{\mathrm{Brier}}(\tilde{p}) = 1 - \sum_{i=1}^{n-1} p_i^2 - \left(1 - \sum_{i=1}^{n-1} p_i\right)^2$.

Theorem 15 *The Brier score is mixable, with mixability constant $\eta_{\mathrm{Brier}} = 1$.*

Proof It can be verified by basic calculus that ℓ_{Brier} is continuous and continuously differentiable on $\mathrm{int}(\tilde{\Delta}^n)$. To see that it is strictly proper, note that for $\hat{p}\neq p$ the inequality $L_{\mathrm{Brier}}(p,\hat{p}) > \underline{L}_{\mathrm{Brier}}(p)$ is equivalent to

$$\sum_{i=1}^{n}(p_i^2 - 2p_i\hat{p}_i + \hat{p}_i^2) > 0 \qquad \text{or} \qquad \sum_{i=1}^{n}(p_i - \hat{p}_i)^2 > 0,$$

1. This is the definition used by Vovk and Zhdanov (2009). Cesa-Bianchi and Lugosi (2006) use a different definition (for the binary case) which differs by a constant. Their definition results in $\tilde{\underline{L}}(\tilde{p}) = \tilde{p}(1-\tilde{p})$ and thus $\tilde{\underline{L}}''(\tilde{p}) = -2$. If $n=2$, then $\tilde{\underline{L}}_{\mathrm{Brier}}$ as defined above leads to $\tilde{\underline{L}}''_{\mathrm{Brier}}(\tilde{p}) = \mathsf{H}\tilde{\underline{L}}_{\mathrm{Brier}}(\tilde{p}) = -2(1+1) = -4$.

and the latter inequality is true because $p_i \neq \hat{p}_i$ for at least one i by assumption. Hence the conditions of Theorem 10 are satisfied.

We will first prove that $\eta_{\text{Brier}} \leq 1$ by showing that convexity of $\eta\tilde{\underline{L}}_{\text{Brier}}(\tilde{p}) - \tilde{\underline{L}}_{\log}(\tilde{p})$ on $\text{int}(\tilde{\Delta}^n)$ implies $\eta \leq 1$. If $\eta\tilde{\underline{L}}_{\text{Brier}}(\tilde{p}) - \tilde{\underline{L}}_{\log}(\tilde{p})$ is convex, then it is convex as a function of p_1 when all other elements of $\tilde{p}$ are kept fixed. Consequently, the second derivative with respect to p_1 must be nonnegative:

$$0 \leq \frac{\partial^2}{\partial p_1^2}\left(\eta\tilde{\underline{L}}_{\text{Brier}}(\tilde{p}) - \tilde{\underline{L}}_{\log}(\tilde{p})\right) = \frac{1}{p_1} + \frac{1}{p_n} - 4\eta.$$

By letting p_1 and p_n both tend to $1/2$, it follows that $\eta \leq 1$.

It remains to show that $\eta_{\text{Brier}} \geq 1$. By Theorem 10 it is sufficient to show that, for $\eta \leq 1$, $\eta\underline{L}_{\text{Brier}}(p) - \underline{L}_{\log}(p)$ is convex on $\text{rel}\,\text{int}(\Delta^n)$. We proceed by induction. For $n = 1$, the required convexity holds trivially. Suppose the lemma holds for $n-1$, and let $f_n(p_1,\ldots,p_n) = \eta\underline{L}_{\text{Brier}}(p) - \underline{L}_{\log}(p)$ for all n. Then for $n \geq 2$

$$f_n(p_1,\ldots,p_n) = f_{n-1}(p_1+p_2,p_3,\ldots,p_n) + g(p_1,p_2),$$

where $g(p_1,p_2) = -\eta p_1^2 - \eta p_2^2 + \eta(p_1+p_2)^2 + p_1\ln p_1 + p_2 \ln p_2 - (p_1+p_2)\ln(p_1+p_2)$. Since f_{n-1} is convex by inductive assumption and the sum of two convex functions is convex, it is therefore sufficient to show that $g(p_1,p_2)$ is convex or, equivalently, that its Hessian is positive semi-definite. Abbreviating $q = p_1 + p_2$, we have that

$$\mathsf{H}g(p_1,p_2) = \begin{pmatrix} 1/p_1 - 1/q & 2\eta - 1/q \\ 2\eta - 1/q & 1/p_2 - 1/q \end{pmatrix}.$$

A 2×2 matrix is positive semi-definite if its trace and determinant are both non-negative, which is easily verified in the present case: $\mathsf{Tr}(\mathsf{H}g(p_1,p_2)) = 1/p_1 + 1/p_2 - 2/q \geq 0$ and $|\mathsf{H}g(p_1,p_2)| = (1/p_1 - 1/q)(1/p_2 - 1/q) - (2\eta - 1/q)^2$, which is non-negative if

$$\begin{aligned} \frac{1}{p_1 p_2} - \frac{1}{p_1 q} - \frac{1}{p_2 q} &\geq 4\eta^2 - \frac{4\eta}{q} \\ 0 &\geq 4\eta^2 q - 4\eta \\ \eta q &\leq 1. \end{aligned}$$

Since $q = p_1 + p_2 \leq 1$, this inequality holds for $\eta \leq 1$, which shows that $g(p_1,p_2)$ is convex and thereby completes the proof. ■

6. Extension to Improper Losses

Our results are stated for proper losses. However, they also extend to a large class of *improper* (i.e., not proper) loss functions $\ell_{\text{imp}}\colon \mathcal{V} \to [0,\infty]$, which may be related to a proper loss ℓ with the same mixability constant using the following construction.

For any distribution $p \in \Delta^n$ and action $v \in \mathcal{V}$, let $L_{\text{imp}}(p,v) = p'\ell_{\text{imp}}(v)$ denote the risk and let $\underline{L}_{\text{imp}}(p) = \inf_{v\in\mathcal{V}} L_{\text{imp}}(p,v)$ denote the Bayes risk for ℓ_{imp}. If the infimum in the definition of the Bayes risk is achieved for all p, there exists a (possibly non-unique) *reference link* $\psi_{\text{imp}}\colon \Delta^n \to \mathcal{V}$ (Reid and Williamson, 2010), which is a function satisfying

$$L_{\text{imp}}(p,\psi_{\text{imp}}(p)) = \underline{L}_{\text{imp}}(p).$$

This function can be seen as one which "calibrates" ℓ_{imp} by returning $\psi_{\text{imp}}(p)$, the best possible prediction under outcomes distributed by p. The loss function defined by

$$\ell(q) := \ell_{\text{imp}}(\psi_{\text{imp}}(q)) \qquad (q \in \Delta^n)$$

is proper by definition of the reference link.

If for every action $v \in \mathcal{V}$ there exists a distribution $p \in \Delta^n$ such that $\psi_{\text{imp}}(p) = v$ (i.e., the reference link is surjective), then ℓ is just a reparametrization of ℓ_{imp} and their superprediction sets S_ℓ and $S_{\ell_{\text{imp}}}$, as defined in (4), are the same. It then follows that $E_\eta(S_\ell) = E_\eta(S_{\ell_{\text{imp}}})$ for all η, such that ℓ and ℓ_{imp} must have the same mixability constants.

It turns out that the superprediction sets of ℓ and ℓ_{imp} are often the same even if ψ_{imp} is not surjective. This follows from Theorem 20 of Chernov et al. (2010) and its proof,[2] which may be reformulated as follows.

Theorem 16 (Chernov et al., 2010) *Let $\Lambda_{\text{imp}} = \ell_{\text{imp}}(\mathcal{V})$ be the set of achievable loss vectors. Suppose ℓ_{imp} is mixable and satisfies the following conditions:*

(i.) Λ_{imp} is a compact subset of $[0,\infty]^n$ (in the extended topology);

(ii.) There exists an action $v \in \mathcal{V}$ such that all components of $\ell_{\text{imp}}(v)$ are finite;

(iii.) For every distribution $p \in \Delta^n$ such that $p_i = p_j = 0$ for some $i \neq j$, the minimum of $L_{\text{imp}}(p,\cdot)$ is unique.

Then a unique reference link ψ_{imp} exists and $S_\ell = S_{\ell_{\text{imp}}}$, so ℓ and ℓ_{imp} have the same mixability constants. Moreover, ℓ is continuous and strictly proper.

Remark 17 *To see the equivalence between our version and Theorem 20 of Chernov et al. (2010), note that mixability of ℓ_{imp} implies that $\Sigma_\Lambda^\eta = \Sigma_\Lambda$ in their notation, for any $\eta > 0$ such that ℓ_{imp} is η-mixable.*

It seems likely that the mixability constants for ℓ_{imp} and ℓ will be the same even under weaker conditions than those of Theorem 16. In particular, we suspect that mixability of ℓ_{imp} is not always necessary, and Chernov and Vovk (2010) suggest that Condition iii may be removed. See also the discussion on mixability of composite losses by Vernet et al. (2012).

In the absence of such strengthenings of Theorem 16, it may be useful to recall that exp-concavity of ℓ_{imp} implies mixability (Cesa-Bianchi and Lugosi, 2006). An easy test to determine the mixability constant for ℓ_{imp} in some cases where it is 0, is given by the following observation (Kalnishkan and Vyugin, 2008):

Lemma 18 *If $S_{\ell_{\text{imp}}}$ is not convex, then ℓ_{imp} is not mixable.*

Proof Suppose ℓ_{imp} is η-mixable for some $\eta > 0$. Then, for any $x, y \in S_{\ell_{\text{imp}}}$ and any $\lambda \in [0,1]$, the set $E_\eta(S_{\ell_{\text{imp}}})$ contains the point $z = (1-\lambda)E_\eta(x) + \lambda E_\eta(y)$. Consequently, $S_{\ell_{\text{imp}}}$ itself contains $z' = E_\eta^{-1}(z)$, and by construction each component of z' satisfies

$$z'_i = -\frac{1}{\eta}\ln\left((1-\lambda)e^{-\eta x_i} + \lambda e^{-\eta y_i}\right) \leq (1-\lambda)x_i + \lambda y_i \qquad (i = 1,\ldots,n)$$

2. We thank a COLT2011 referee for referring us to this result.

by convexity of the exponential function. It follows that the point $(1-\lambda)x+\lambda y$ dominates z' and hence is also contained in $S_{\ell_{\text{imp}}}$. Thus $S_{\ell_{\text{imp}}}$ is convex, and we have shown that mixability implies convexity of $S_{\ell_{\text{imp}}}$, from which the result follows. ■

7. Connection to α-Flatness and Strong Convexity

We now briefly relate our result to recent work by Abernethy et al. (2009). They formulate the learning problem slightly differently. They do not restrict themselves to proper losses and so the predictions are not restricted to the simplex. This means it is not necessary to go to the submanifold $\tilde{\Delta}^n$ in order for derivatives to be well defined.

Abernethy et al. (2009) have developed their own bounds on cumulative loss in terms of the α-flatness (defined below) of $\underline{L}(p)$. They show that α-flatness is implied by strong convexity of the loss ℓ. The duality between the loss surface and Bayes risk that they established through the use of support functions can also be seen in Lemma 6 in the relationship between the Hessian of $\underline{\tilde{L}}$ and the derivative of $\tilde{\ell}$. Although it is obscured somewhat due to our use of functions of $\tilde{p}$, this relationship is due to the properness of ℓ guaranteeing that ℓ^{-1} is the (homogeneously extended) Gauss map for the surface $\underline{\tilde{L}}$. Below we point out the relationship between α-flatness and the positive definiteness of $\mathsf{H}\underline{L}(p)$ (we stress that in our work we used $\mathsf{H}\underline{\tilde{L}}(\tilde{p})$). Whilst the two results are not precisely comparable, the comparison below seems to suggest that the condition of Abernethy et al. (2009) is stronger than necessary.

Suppose $\mathcal{X}$ is a Banach space with norm $\|\cdot\|$. Given a real number $\alpha>0$ and a function $\sigma:\mathbb{R}_+\to[0,\infty]$ such that $\sigma(0)=0$, a convex function $f\colon\mathcal{X}\to\mathbb{R}$ is said to be $(\alpha,\sigma,\|\cdot\|)$-*flat* (or $(\alpha,\sigma,\|\cdot\|)$-smooth)[3] if for all $x,x_0\in\mathcal{X}$,

$$f(x)-f(x_0)\leq \mathsf{D}f(x_0)\cdot(x-x_0)+\alpha\sigma(\|x-x_0\|).$$

A concave function g is flat if the convex function $-g$ is flat. When $\|\cdot\|=\|\cdot\|_2$, and $\sigma(x)=x^2$, it is known (Hiriart-Urruty and Lemaréchal, 1993) that for $\alpha>0$, f is $(\alpha,x\mapsto x^2,\|\cdot\|_2)$-flat if and only if $f-\alpha\|\cdot\|^2$ is concave. Thus f is α-flat if and only if $\mathsf{H}(f-\alpha\|\cdot\|^2)$ is negative semi-definite, which is equivalent to $\mathsf{H}f-2\alpha I\preccurlyeq 0 \iff \mathsf{H}f\preccurlyeq 2\alpha I$.

Abernethy et al. (2009) show that if $\underline{L}$ is $(\alpha,x\mapsto x^2,\|\cdot\|_1)$-flat, then the minimax regret for a prediction game with T rounds is bounded above by $4\alpha\log T$. It is thus of interest to relate their assumption on $\underline{L}$ to the mixability condition (which guarantees constant regret, in the prediction with experts setting).

In contrast to the above quoted result for $\|\cdot\|_2$, we only get a one-way implication for $\|\cdot\|_1$.

Lemma 19 *If $f-\alpha\|\cdot\|_1^2$ is concave on $\mathbb{R}^n_+$ then f is $(\alpha,x\mapsto x^2,\|\cdot\|_1)$-flat.*

Proof It is known (Hiriart-Urruty and Lemaréchal, 1993, page 183) that a function h is concave if and only if $h(x)\leq h(x_0)+\mathsf{D}h(x_0)\cdot(x-x_0)$ for all x,x_0. Hence $f-\alpha\|\cdot\|_1^2$ is concave on $\mathbb{R}^n_+$ if and

3. This definition is redundantly parametrised: $(\alpha,\sigma,\|\cdot\|)$-flatness is equivalent to $(1,\alpha\sigma,\|\cdot\|)$-flatness. We have defined the notion as above in order to relate to existing definitions and because in fact one sometimes fixes σ and then is interested in the effect of varying α. When $\sigma(x)=x^2$, Abernethy et al. (2009) and Kakade et al. (2010) call this α-flat with respect to $\|\cdot\|$. Azé and Penot (1995) and Zălinescu (1983) would say f is σ-flat with respect to an implicitly given norm if f is (in our definition) $(\alpha,\sigma,\|\cdot\|)$-flat for some $\alpha>0$ (which in their setup is effectively bundled into σ). These differences do not matter (unless one wishes to use results from the earlier literature, which we do not).

only if for all $x, x_0 \in \mathbb{R}^n_+$,

$$\begin{aligned} & f(x) - \alpha\|x\|_1^2 \;\leq\; f(x_0) - \alpha\|x_0\|_1^2 + \mathsf{D}(f(x_0) - \alpha\|x_0\|_1^2) \cdot (x - x_0) \\ \Leftrightarrow\; & f(x) - f(x_0) \;\leq\; \alpha\|x\|_1^2 - \alpha\|x_0\|_1^2 + \mathsf{D}f(x_0) \cdot (x - x_0) - \alpha\mathsf{D}(\|x_0\|_1^2) \cdot (x - x_0). \end{aligned} \tag{23}$$

Since $\mathsf{D}(\|x_0\|_1^2) = 2\|x_0\|_1 \mathbb{1}$ and $2\|x_0\|_1(\mathbb{1} \cdot (x - x_0)) = 2\|x_0\|_1(\|x\|_1 - \|x_0\|_1) = 2\|x_0\|_1\|x\|_1 - 2\|x_0\|_1^2$,

$$\begin{aligned} (23) \Leftrightarrow\; & f(x) - f(x_0) \;\leq\; \alpha\left(\|x\|_1^2 + \|x_0\|^2 - 2\|x_0\|_1\|x\|_1\right) + \mathsf{D}f(x_0) \cdot (x - x_0) \\ \Leftrightarrow\; & f(x) - f(x_0) \;\leq\; Df(x_0) \cdot (x - x_0) + \alpha\left(\|x\|_1 - \|x_0\|_1\right)^2. \end{aligned}$$

By the reverse triangle inequality $\|x - x_0\|_1 \geq |\|x\|_1 - \|x_0\|_1| \geq \|x\|_1 - \|x_0\|_1$ and thus $\|x - x_0\|_1^2 \geq (\|x\|_1 - \|x_0\|_1)^2$, which gives

$$\Rightarrow\; f(x) - f(x_0) \;\leq\; \mathsf{D}f(x_0) \cdot (x - x_0) + \alpha\|x - x_0\|_1^2.$$

■

Now $f - \alpha\|\cdot\|_1^2$ is concave if and only if $\mathsf{H}(f - \alpha\|\cdot\|_1^2) \preccurlyeq 0$. We have (again for $x \in \mathbb{R}^n_+$) $\mathsf{H}(f - \alpha\|\cdot\|_1^2) = \mathsf{H}f - \alpha\mathsf{H}(\|\cdot\|_1^2)$. Let $\phi(x) = \|x\|_1^2$. Then $\mathsf{D}\phi(x) = 2\|x\|_1\mathsf{D}(\|x\|_1) = 2\|x\|_1\mathbb{1}$. Hence $\mathsf{H}\phi(x) = \mathsf{D}\left(\mathsf{D}\phi(x)\right)') = \mathsf{D}(2\|x\|_1\mathbb{1}') = 2\mathbb{1} \cdot \mathbb{1}'$. Thus $(\alpha, x \mapsto x^2, \|\cdot\|_1)$-flatness of $\underline{L}$ is implied by negative semi-definiteness of the Hessian of $\underline{L}$ relative to $2\alpha\mathbb{1} \cdot \mathbb{1}'$, instead of $\underline{L}_{\log}$ (see Theorem 10, part ii). The comparison with log loss is not that surprising in light of the observations regarding mixability by Grünwald (2007, §17.9).

The above analysis is not entirely satisfactory for three reasons: 1) Lemma 19 does not characterise the flatness condition (it is only a sufficient condition); 2) we have glossed over the fact that in order to compute derivatives one needs to work in $\tilde{\Delta}^n$; and 3) the learning protocols for the two situations are not identical. These last two points can be potentially addressed in future work. However the first seems impossible since there can not exist a characterisation of $(\alpha, x \mapsto x^2, \|\cdot\|_1)$-flatness in terms of concavity of some function. To see this, consider the one dimensional case and suppose there was some function g such that f was flat if g was concave. Then we would require $\mathsf{D}g(x) \cdot (x - x_0) = \alpha\|x - x_0\|_1^2 \Rightarrow \mathsf{D}g(x)(x - x_0) = \alpha|x - x_0|^2 = \alpha(x - x_0)^2 \Rightarrow \mathsf{D}g(x) = \alpha(x - x_0)$ which is impossible because the left hand side $\mathsf{D}g(x)$ does not depend upon x_0. On the other hand, perhaps it is not worth further investigation since the result due to Abernethy et al. (2009) is only a *sufficient* condition for logarithmic regret.

8. Conclusion

Mixability characterizes fast rates in the prediction with expert advice setting in terms of the mixability constant. An explicit formula to determine the mixability constant was previously available only for binary-valued outcomes, and the formula did not have a clear interpretation.

For strictly proper losses, Theorem 13 simplifies this formula and generalises it to outcomes with any finite number of possible values. The new formula has a clear interpretation as the minimal curvature of the Bayes risk for the loss relative to log loss. This shows in a precise and intuitive way the effect of the choice of loss function on the worst-case regret of the learner, and the special

role played by log loss in such settings. Closely related characterizations of mixability are given in Theorem 10 and Corollary 14.

Although our main results are stated only for proper losses, Section 6 shows that many losses that are not proper can be related to a proper loss with the same mixability constant, which implies that our results cover these improper losses as well.

Acknowledgments

We thank the JMLR referees for helpful comments which improved the presentation, and Elodie Vernet for useful technical discussions. This work was supported by the Australian Research Council and NICTA which is funded by the Australian Government through the ICT Centre of Excellence program. It was done while Tim van Erven was affiliated with the Centrum Wiskunde & Informatica, Amsterdam, the Netherlands. Some of the work was done while all the authors were visiting Microsoft Research, Cambridge and some was done while Tim van Erven was visiting ANU and NICTA. It was also supported in part by the IST Programme of the European Community, under the PASCAL2 Network of Excellence, IST-2007-216886. This publication only reflects the authors' views. An earlier and shorter version of this paper appeared in the proceedings of COLT2011, and the present version has benefited from comments from the COLT referees.

Appendix A. Matrix Calculus

We adopt the notation of Magnus and Neudecker (1999): I_n is the $n \times n$ identity matrix, A' is the transpose of A, the n-vector $\mathbb{1}_n := (1,\ldots,1)'$, and $0_{n\times m}$ denotes the zero matrix with n rows and m columns. The unit n-vector $e_i^n := (0,\ldots,0,1,0,\ldots,0)'$ has a 1 in the ith coordinate and zeroes elsewhere. If $A = [a_{ij}]$ is an $n \times m$ matrix, $\operatorname{vec} A$ is the vector of columns of A stacked on top of each other. The *Kronecker product* of an $m \times n$ matrix A with a $p \times q$ matrix B is the $mp \times nq$ matrix

$$A \otimes B := \begin{pmatrix} A_{1,1}B & \cdots & A_{1,n}B \\ \vdots & \ddots & \vdots \\ A_{m,1}B & \cdots & A_{m,n}B \end{pmatrix}.$$

We use the following properties of Kronecker products (See Magnus and Neudecker, 1999, Chapter 2): $(A \otimes B)(C \otimes D) = (AC \otimes BD)$ for all appropriately sized A, B, C, D and $(A \otimes B)^{-1} = (A^{-1} \otimes B^{-1})$ for invertible A and B.

If $f : \mathbb{R}^n \to \mathbb{R}^m$ is differentiable at c then the *partial derivative* of f_i with respect to the jth coordinate at c is denoted $\mathsf{D}_j f_i(c)$ and is often[4] also written as $[\partial f_i / \partial x_j]_{x=c}$. The $m \times n$ matrix of partial derivatives of f is the *Jacobian* of f and denoted

$$(\mathsf{D}f(c))_{i,j} := \mathsf{D}_j f_i(c) \quad \text{for } i \in [m], j \in [n].$$

The *inverse function theorem* relates the Jacobians of a function and its inverse (cf. Fleming, 1977, §4.5):

4. See Chapter 9 of Magnus and Neudecker (1999) for why the $\partial/\partial x$ notation is a poor one for multivariate differential calculus despite its popularity.

Theorem 20 *Let $S \subset \mathbb{R}^n$ be an open set and $g : S \to \mathbb{R}^n$ be a C^q function with $q \geq 1$ (i.e., continuous with at least one continuous derivative). If $\mathsf{D}g(s) \neq 0$ then: there exists an open set S_0 such that $s \in S_0$ and the restriction of g to S_0 is invertible; $g(S_0)$ is open; f, the inverse of the restriction of g to S_0, is C^q; and $\mathsf{D}f(t) = [\mathsf{D}g(s)]^{-1}$ for $t = g(s)$ and $s \in S_0$.*

If F is a matrix valued function $\mathsf{D}F(X) := \mathsf{D}f(\operatorname{vec} X)$ where $f(X) = \operatorname{vec} F(X)$.

We will require the product rule for matrix valued functions (Fackler, 2005): Suppose $f\colon \mathbb{R}^n \to \mathbb{R}^{m\times p}$, $g\colon \mathbb{R}^n \to \mathbb{R}^{p\times q}$ so that $(f \times g)\colon \mathbb{R}^n \to \mathbb{R}^{m\times q}$. Then

$$\mathsf{D}(f \times g)(x) = (g(x)' \otimes I_m) \cdot \mathsf{D}f(x) + (I_q \otimes f(x)) \cdot \mathsf{D}g(x).$$

The *Hessian* at $x \in X \subseteq \mathbb{R}^n$ of a real-valued function $f : \mathbb{R}^n \to \mathbb{R}$ is the $n \times n$ real, symmetric matrix of second derivatives at x

$$(\mathsf{H}f(x))_{j,k} := \mathsf{D}_{k,j}f(x) = \frac{\partial^2 f}{\partial x_k \partial x_j}.$$

Note that the derivative $D_{k,j}$ is in row j, column k. It is easy to establish that the Jacobian of the transpose of the Jacobian of f is the Hessian of f. That is,

$$\mathsf{H}f(x) = \mathsf{D}\left((\mathsf{D}f(x))'\right) \tag{24}$$

(Magnus and Neudecker, 1999, Chapter 10). If $f : X \to \mathbb{R}^m$ for $X \subseteq \mathbb{R}^n$ is a vector valued function then the Hessian of f at $x \in X$ is the $mn \times n$ matrix that consists of the Hessians of the functions f_i stacked vertically:

$$\mathsf{H}f(x) := \begin{pmatrix} \mathsf{H}f_1(x) \\ \vdots \\ \mathsf{H}f_m(x) \end{pmatrix}.$$

The following theorem regarding the chain rule for Hessian matrices can be found in the book of Magnus and Neudecker (1999, pg. 110).

Theorem 21 *Let S be a subset of $\mathbb{R}^n$, and $f : S \to \mathbb{R}^m$ be twice differentiable at a point c in the interior of S. Let T be a subset of $\mathbb{R}^m$ containing $f(S)$, and $g : T \to \mathbb{R}^p$ be twice differentiable at the interior point $b = f(c)$. Then the function $h(x) := g(f(x))$ is twice differentiable at c and*

$$\mathsf{H}h(c) = (I_p \otimes \mathsf{D}f(c))' \cdot (\mathsf{H}g(b)) \cdot \mathsf{D}f(c) + (\mathsf{D}g(b) \otimes I_n) \cdot \mathsf{H}f(c).$$

Applying the chain rule to functions that are inverses of each other gives the following corollary.

Corollary 22 *Suppose $f : \mathbb{R}^n \to \mathbb{R}^n$ is invertible with inverse $g := f^{-1}$. If $b = f(c)$ then*

$$\mathsf{H}f^{-1}(b) = -\left(G \otimes G'\right) \mathsf{H}f(c) G,$$

where $G := [\mathsf{D}f(c)]^{-1} = \mathsf{D}g(b)$.

Proof Since $f \circ g = \mathrm{id}$ and $\mathsf{H}[\mathrm{id}] = 0_{n^2\times n}$ Theorem 21 implies that for c in the interior of the domain of f and $b = f(c)$

$$\mathsf{H}(g \circ f)(c) = (I_n \otimes \mathsf{D}f(c))' \cdot \mathsf{H}g(b) \cdot \mathsf{D}f(c) + (\mathsf{D}g(b) \otimes I_n) \cdot \mathsf{H}f(c) = 0_{n^2\times n}.$$

Solving this for $\mathsf{H}g(b)$ gives

$$\mathsf{H}g(b) = -\left[(I_n \otimes \mathsf{D}f(c))'\right]^{-1} \cdot (\mathsf{D}g(b)) \otimes I_n) \cdot \mathsf{H}f(c) \cdot [\mathsf{D}f(c)]^{-1}.$$

Since $(A \otimes B)^{-1} = (A^{-1} \otimes B^{-1})$ and $(A')^{-1} = (A^{-1})'$ we have $[(I \otimes B)']^{-1} = [(I \otimes B)^{-1}]' = (I^{-1} \otimes B^{-1})' = (I \otimes B^{-1})'$ so the first term in the above product simplifies to $-\left[(I_n \otimes \mathsf{D}f(c)^{-1})\right]'$. The inverse function theorem implies $\mathsf{D}g(b) = [\mathsf{D}f(c)]^{-1} =: G$ and so

$$\begin{aligned}\mathsf{H}g(b) &= -(I_n \otimes G)' \cdot (G \otimes I_n) \cdot \mathsf{H}f(c) \cdot G \\ &= -(G \otimes G') \cdot \mathsf{H}f(c) \cdot G\end{aligned}$$

as required, since $(A \otimes B)(C \otimes D) = (AC \otimes BD)$. ■

References

Jacob Abernethy, Alekh Agarwal, Peter L. Bartlett, and Alexander Rakhlin. A stochastic view of optimal regret through minimax duality. In *Proceedings of the 22nd Annual Conference on Learning Theory*, 2009.

Dominique Azé and Jean-Paul Penot. Uniformly convex and uniformly smooth convex functions. *Annales de la faculté des sciences de Toulouse*, 4(4):705–730, 1995.

Nicolò Cesa-Bianchi and Gabor Lugosi. *Prediction, Learning, and Games*. Cambridge University Press, 2006.

Alexey Chernov and Vladimir Vovk. Prediction with advice of unknown number of experts. In *Proceedings of the 26th Conference on Uncertainty in Artificial Intelligence (UAI 2010)*, 2010.

Alexey Chernov, Yuri Kalnishkan, Fedor Zhdanov, and Vladimir Vovk. Supermartingales in prediction with expert advice. *Theoretical Computer Science*, 411:2647–2669, 2010.

Paul K. Fackler. Notes on matrix calculus. North Carolina State University, 2005.

Wendell H. Fleming. *Functions of Several Variables*. Springer, 1977.

Peter D. Grünwald. *The Minimum Description Length Principle*. MIT Press, Cambridge, MA, 2007.

David Haussler, Jyrki Kivinen, and Manfred K. Warmuth. Sequential prediction of individual sequences under general loss functions. *IEEE Transactions on Information Theory*, 44(5):1906–1925, 1998.

Jean-Baptiste Hiriart-Urruty and Claude Lemaréchal. *Convex Analysis and Minimization Algorithms: Part I: Fundamentals*. Springer, Berlin, 1993.

Roger A. Horn and Charles R. Johnson. *Matrix Analysis*. Cambridge University Press, 1985.

Sham M. Kakade, Shai Shalev-Shwartz, and Ambuj Tewari. Regularization techniques for learning with matrices. arXiv:0910.0610v2, October 2010.

Yuri Kalnishkan and Michael V. Vyugin. The weak aggregating algorithm and weak mixability. *Journal of Computer and System Sciences*, 74:1228–1244, 2008.

Yuri Kalnishkan, Volodya Vovk, and Michael V. Vyugin. Loss functions, complexities, and the Legendre transformation. *Theoretical Computer Science*, 313:195–207, 2004.

Jan R. Magnus and Heinz Neudecker. *Matrix Differential Calculus with Applications in Statistics and Econometrics (revised edition)*. John Wiley & Sons, Ltd., 1999.

Mark D. Reid and Robert C. Williamson. Composite binary losses. *Journal of Machine Learning Research*, 11:2387–2422, 2010.

Mark D. Reid and Robert C. Williamson. Information, divergence and risk for binary experiments. *Journal of Machine Learning Research*, 12:731–817, March 2011.

John A. Thorpe. *Elementary Topics in Differential Geometry*. Springer, 1979.

Elodie Vernet, Robert C. Williamson, and Mark D. Reid. Composite multiclass losses. *Journal of Machine Learning Research*, March 2012. To be submitted.

Volodya Vovk. Aggregating strategies. In *Proceedings of the Third Annual Workshop on Computational Learning Theory (COLT)*, pages 371–383, 1990.

Volodya Vovk. A game of prediction with expert advice. In *Proceedings of the Eighth Annual Conference on Computational Learning Theory*, pages 51–60. ACM, 1995.

Volodya Vovk. Competitive on-line statistics. *International Statistical Review*, 69(2):213–248, 2001.

Volodya Vovk and Fedor Zhdanov. Prediction with expert advice for the Brier game. *Journal of Machine Learning Research*, 10:2445–2471, 2009.

Constantin Zălinescu. On uniformly convex functions. *Journal of Mathematical Analysis and Applications*, 95:344–374, 1983.

Journal of Machine Learning Research 13 (2012) 1665-1697 Submitted 9/10; Revised 12/11; Published 5/12

Restricted Strong Convexity and Weighted Matrix Completion: Optimal Bounds with Noise

Sahand Negahban SAHAND_N@EECS.BERKELEY.EDU
Martin J. Wainwright* WAINWRIG@STAT.BERKELEY.EDU
Department of Electrical Engineering and Computer Sciences
University of California
Berkeley, CA 94720-1776, USA

Editor: Inderjit Dhillon

Abstract

We consider the matrix completion problem under a form of row/column weighted entrywise sampling, including the case of uniform entrywise sampling as a special case. We analyze the associated random observation operator, and prove that with high probability, it satisfies a form of restricted strong convexity with respect to weighted Frobenius norm. Using this property, we obtain as corollaries a number of error bounds on matrix completion in the weighted Frobenius norm under noisy sampling and for both exact and near low-rank matrices. Our results are based on measures of the "spikiness" and "low-rankness" of matrices that are less restrictive than the incoherence conditions imposed in previous work. Our technique involves an M-estimator that includes controls on both the rank and spikiness of the solution, and we establish non-asymptotic error bounds in weighted Frobenius norm for recovering matrices lying with ℓ_q-"balls" of bounded spikiness. Using information-theoretic methods, we show that no algorithm can achieve better estimates (up to a logarithmic factor) over these same sets, showing that our conditions on matrices and associated rates are essentially optimal.

Keywords: matrix completion, collaborative filtering, convex optimization

1. Introduction

Matrix completion problems correspond to reconstructing matrices, either exactly or approximately, based on observing a subset of their entries (Laurent, 2001; Deza and Laurent, 1997). In the simplest formulation of matrix completion, the observations are assumed to be uncorrupted, whereas a more general formulation (as considered in this paper) allows for noisiness in these observations. Matrix recovery based on only partial information is an ill-posed problem, and accurate estimates are possible only if the matrix satisfies additional structural constraints, with examples including bandedness, positive semidefiniteness, Euclidean distance measurements, Toeplitz, and low-rank structure (see the survey paper by Laurent (2001) and references therein for more background).

The focus of this paper is low-rank matrix completion based on noisy observations. This problem is motivated by a variety of applications where an underlying matrix is likely to have low-rank, or near low-rank structure. The archetypal example is the Netflix challenge, a version of the collaborative filtering problem, in which the unknown matrix is indexed by individuals and movies, and each observed entry of the matrix corresponds to the rating assigned to the associated movie

*. Also in the department of Statistics.

by the given individual. Since the typical person only watches a tiny number of movies (compared to the total Netflix database), it is only a sparse subset of matrix entries that are observed. In this context, one goal of collaborative filtering is to use the observed entries to make recommendations to a person regarding movies that they have *not* yet seen. We refer the reader to Srebro (2004) (and references therein) for further discussion and motivation for collaborative filtering and related problems.

In this paper, we analyze a method for approximate low-rank matrix recovery using an M-estimator that is a combination of a data term, and a weighted nuclear norm as a regularizer. The nuclear norm is the sum of the singular values of a matrix (Horn and Johnson, 1985), and has been studied in a body of past work, both on matrix completion and more general problems of low-rank matrix estimation (e.g., Fazel, 2002; Srebro, 2004; Srebro et al., 2005, 2004; Recht et al., 2010; Bach, 2008; Candes and Tao, 2010; Recht, 2011; Keshavan et al., 2010a,b; Negahban and Wainwright, 2011; Rohde and Tsybakov, 2011). A parallel line of work has studied computationally efficient algorithms for solving problems with nuclear norm constraints (e.g, Mazumber et al., 2010; Nesterov, 2007; Lin et al., 2009). Here we limit our detailed discussion to those papers that study various aspects of the matrix completion problem. Motivated by various problems in collaborative filtering, Srebro (2004) and Srebro et al. (2005) studied various aspects nuclear norm regularization, and established generalization error bounds under certain conditions. Candès and Recht (2009) studied the exact reconstruction of a low-rank matrix given perfect (noiseless) observations of a subset of entries, and provided sufficient conditions for exact recovery via nuclear norm relaxation, with later refinements provided by various authors (Candes and Tao, 2010; Recht, 2011; Gross, 2011). In particular, Gross (2011) recognized the utility of the Ahlswede-Winter matrix concentration bounds, and the simplest argument to date is provided by Recht (2011). In a parallel line of work, Keshavan et al. (2010a,b) have studied a method based on thresholding and singular value decomposition, and established various results on its behavior, both for noiseless and noisy matrix completion. Among other results, Rohde and Tsybakov (2011) establish prediction error bounds for matrix completion, a different metric than the matrix recovery problem of interest here. In recent work, Salakhutdinov and Srebro (2010) provided various motivations for the use of weighted nuclear norms, in particular showing that the standard nuclear norm relaxation can behave very poorly when the sampling is non-uniform. The analysis of this paper applies to both uniform and non-uniform sampling, as well as a form of reweighted nuclear norm as suggested by these authors, one which includes the ordinary nuclear norm as a special case. We provide a more detailed comparison between our results and some aspects of past work in Section 3.4.

As has been noted before (Candès and Plan, 2010), a significant theoretical challenge is that conditions that have proven very useful for sparse linear regression—among them the restricted isometry property—are *not* satisfied for the matrix completion problem. For this reason, it is natural to seek an alternative and less restrictive property that might be satisfied in the matrix completion setting. In recent work, Negahban et al. (2009) have isolated a weaker and more general condition known as *restricted strong convexity* (RSC), and proven that certain statistical models satisfy RSC with high probability when the associated regularizer satisfies a *decomposability* condition. When an M-estimator satisfies the RSC condition, it is relatively straightforward to derive non-asymptotic error bounds on parameter estimates (Negahban et al., 2009). The class of decomposable regularizers includes the nuclear norm as particular case, and the RSC/decomposability approach has been exploited to derive bounds for various matrix estimation problems, among them multi-task learning, autoregressive system identification, and compressed sensing (Negahban and Wainwright, 2011).

To date, however, an open question is whether or not an appropriate form of RSC holds for the matrix completion problem. If it did hold, then it would be possible to derive non-asymptotic error bounds (in Frobenius norm) for matrix completion based on noisy observations. Within this context, the main contribution of this paper is to prove that with high probability, a form of the RSC condition holds for the matrix completion problem, in particular over an interesting set of matrices $\mathfrak{C}$, as defined in Equation (4) to follow, that have both low nuclear/Frobenius norm ratio and low "spikiness". Exploiting this RSC condition then allows us to derive non-asymptotic error bounds on matrix recovery in weighted Frobenius norms, both for exactly and approximately low-rank matrices. The theoretical core of this paper consists of three main results. Our first result (Theorem 1) proves that the matrix completion loss function satisfies restricted strong convexity with high probability over the set $\mathfrak{C}$. Our second result (Theorem 2) exploits this fact to derive a non-asymptotic error bound for matrix recovery in the weighted Frobenius norm, one applicable to general matrices. We then specialize this result to the problem of estimating exactly low-rank matrices (with a small number of non-zero singular values), as well as near low-rank matrices characterized by relatively swift decay of their singular values. To the best of our knowledge, our results on near low-rank matrices are the first for approximate matrix recovery in the noisy setting, and as we discuss at more length in Section 3.4, our results on the exactly low-rank case are sharper than past work on the problem. Indeed, our final result (Theorem 3) uses information-theoretic techniques to establish that up to logarithmic factors, no algorithm can obtain faster rates than our method over the ℓ_q-balls of matrices with bounded spikiness treated in this paper.

The remainder of this paper is organized as follows. We begin in Section 2 with background and a precise formulation of the problem. Section 3 is devoted to a statement of our main results, and discussion of some of their consequences. In Sections 4 and Section 5, we prove our main results, with more technical aspects of the arguments deferred to appendices. We conclude with a discussion in Section 6.

2. Background and Problem Formulation

In this section, we introduce background on low-rank matrix completion problem, and also provide a precise statement of the problem studied in this paper.

2.1 Uniform and Weighted Sampling Models

Let $\Theta^* \in \mathbb{R}^{d_r \times d_c}$ be an unknown matrix, and consider an observation model in which we make n i.i.d. observations of the form

$$\widetilde{y}_i = \Theta^*_{j(i)k(i)} + \frac{\nu}{\sqrt{d_r d_c}} \widetilde{\xi}_i, \tag{1}$$

Here the quantities $\frac{\nu}{\sqrt{d_r d_c}} \widetilde{\xi}_i$ correspond to additive observation noises with variance appropriately scaled according to the matrix dimensions. In defining the observation model, one can either allow the Frobenius norm of Θ^* to grow with the dimension, as in done in other work (Candès and Plan, 2010; Keshavan et al., 2010b), or rescale the noise as we have done here. This choice is consistent with our assumption that Θ^* has constant Frobenius norm regardless of its rank or dimensions. With this scaling, each observation in the model (1) has a constant signal-to-noise ratio regardless of matrix dimensions.

In the simplest model, the row $j(i)$ and column $k(i)$ indices are chosen uniformly at random from the sets $\{1,2,\ldots,d_r\}$ and $\{1,2,\ldots,d_c\}$ respectively. In this paper, we consider a somewhat more general weighted sampling model. In particular, let $R \in \mathbb{R}^{d_r \times d_r}$ and $C \in \mathbb{R}^{d_c \times d_c}$ be diagonal matrices, with rescaled diagonals $\{R_j/d_r, j = 1,2,\ldots,d_r\}$ and $\{C_k/d_c, k = 1,2,\ldots,d_c\}$ representing probability distributions over the rows and columns of an $d_r \times d_c$ matrix. We consider the weighted sampling model in which we make a noisy observation of entry (j,k) with probability $R_j C_k/(d_r d_c)$, meaning that the row index $j(i)$ (respectively column index $k(i)$) is chosen according to the probability distribution R/d_r (respectively C/d_c). Note that in the special case that $R = \mathbf{1}_{d_r}$ and $C = \mathbf{1}_{d_c}$, the observation model (1) reduces to the usual model of uniform sampling.

We assume that each row and column is sampled with positive probability, in particular that there is some constant $1 \leq L < \infty$ such that $R_a \geq 1/L$ and $C_b \geq 1/L$ for all rows and columns. However, apart from the constraints $\sum_{a=1}^{d_r} R_{aa} = d_r$ and $\sum_{b=1}^{d_c} C_{bb} = d_c$, we do not require that the row and column weights remain bounded as d_r and d_c tend to infinity.

2.2 The Observation Operator and Restricted Strong Convexity

We now describe an alternative formulation of the observation model (1) that, while statistically equivalent to the original, turns out to be more natural for analysis. For each $i = 1,2,\ldots,n$, define the matrix

$$X^{(i)} = \sqrt{d_r d_c}\, \varepsilon_i\, e_{a(i)} e_{b(i)}^T,$$

where $\varepsilon_i \in \{-1,+1\}$ is a random sign, and consider the observation model

$$y_i = \langle\langle X^{(i)},\, \Theta^* \rangle\rangle + \nu \xi_i, \qquad \text{for } i = 1,\ldots,n, \tag{2}$$

where $\langle\langle A,\, B \rangle\rangle := \sum_{j,k} A_{jk} B_{jk}$ is the trace inner product, and ξ_i is an additive noise from the same distribution as the original model. The model (2) is can be obtained from the original model (1) by rescaling all terms by the factor $\sqrt{d_r d_c}$, and introducing the random signs ε_i. The rescaling has no statistical effect, and nor do the random signs, since the noise is symmetric (so that $\xi_i = \varepsilon_i \widetilde{\xi}_i$ has the same distribution as $\widetilde{\xi}_i$). Thus, the observation model (2) is statistically equivalent to the original one (1).

In order to specify a vector form of the observation model, let us define a linear operator $\mathfrak{X}_n : \mathbb{R}^{d_r \times d_c} \to \mathbb{R}^n$ via

$$[\mathfrak{X}_n(\Theta)]_i := \langle\langle X^{(i)},\, \Theta \rangle\rangle, \quad \text{for } i = 1,2,\ldots n.$$

We refer to $\mathfrak{X}_n$ as the *observation operator*, since it maps any matrix $\Theta \in \mathbb{R}^{d_r \times d_c}$ to an n-vector of samples. With this notation, we can write the observations (2) in a vectorized form as $y = \mathfrak{X}_n(\Theta^*) + \nu\xi$.

The reformulation (2) is convenient for various reasons. For any matrix $\Theta \in \mathbb{R}^{d_r \times d_c}$, we have $\mathbb{E}[\langle\langle X^{(i)},\, \Theta \rangle\rangle] = 0$ and

$$\mathbb{E}\big[\langle\langle X^{(i)},\, \Theta \rangle\rangle^2\big] = \sum_{j=1}^{d_r} \sum_{k=1}^{d_c} R_j \Theta_{jk}^2 C_k \;=\; \underbrace{|||\sqrt{R}\Theta\sqrt{C}|||_F^2}_{|||\Theta|||_{\omega(F)}^2},$$

where we have defined the *weighted Frobenius norm* $|||\cdot|||_{\omega(F)}$ in terms of the row R and column C weights. As a consequence, the signal-to-noise ratio in the observation model (2) is given by the ratio $\text{SNR} = \frac{|||\Theta^*|||^2_{\omega(F)}}{\nu^2}$.

As shown by Negahban et al. (2009), a key ingredient in establishing error bounds for the observation model (2) is obtaining lower bounds on the restricted curvature of the sampling operator—in particular, to establish the existence of a constant $c > 0$, which may be arbitrarily small as long as it is positive, such that

$$\frac{\|\mathfrak{X}_n(\Theta)\|_2}{\sqrt{n}} \geq c\,|||\Theta|||_{\omega(F)}. \tag{3}$$

For sample sizes of interest for matrix completion ($n \ll d_r d_c$) , one cannot expect such a bound to hold uniformly over all matrices $\Theta \in \mathbb{R}^{d_r \times d_c}$, even when rank constraints are imposed. Indeed, as noted by Candès and Plan (2010), the condition (3) is violated with high probability by the rank one matrix Θ^* such that $\Theta^*_{11} = 1$ with all other entries zero. Indeed, for a sample size $n \ll d_r d_c$, we have a vanishing probability of observing the entry Θ^*_{11}, so that $\mathfrak{X}_n(\Theta^*) = 0$ with high probability.

2.3 Controlling the Spikiness and Rank

Intuitively, one must exclude matrices that are overly "spiky" in order to avoid the phenomenon just described. Past work has relied on fairly restrictive matrix incoherence conditions (see Section 3.4 for more discussion), based on specific conditions on singular vectors of the unknown matrix Θ^*. In this paper, we formalize the notion of "spikiness" in a natural and less restrictive way—namely by comparing a weighted form of ℓ_∞-norm to the weighted Frobenius norm. In particular, for any non-zero matrix Θ, let us define (for any non-zero matrix) the *weighted spikiness ratio*

$$\alpha_{\text{sp}}(\Theta) := \sqrt{d_r d_c}\ \frac{|||\Theta|||_{\omega(\infty)}}{|||\Theta|||_{\omega(F)}},$$

where $|||\Theta|||_{\omega(\infty)} := \|\sqrt{R}\Theta\sqrt{C}\|_\infty$ is the weighted elementwise ℓ_∞-norm. Note that this ratio is invariant to the scaling of Θ, and satisfies the inequalities $1 \leq \alpha_{\text{sp}}(\Theta) \leq \sqrt{d_r d_c}$. We have $\alpha_{\text{sp}}(\Theta) = 1$ for any non-zero matrix whose entries are all equal, whereas the opposite extreme $\alpha_{\text{sp}}(\Theta) = \sqrt{d_r d_c}$ is achieved by the "maximally spiky" matrix that is zero everywhere except for a single position.

In order to provide a tractable measure of how close Θ is to a low-rank matrix, we define (for any non-zero matrix) the ratio

$$\beta_{\text{ra}}(\Theta) := \frac{|||\Theta|||_{\omega(1)}}{|||\Theta|||_{\omega(F)}}$$

which satisfies the inequalities $1 \leq \beta_{\text{ra}}(\Theta) \leq \sqrt{\min\{d_r, d_c\}}$. By definition of the (weighted) nuclear and Frobenius norms, note that $\beta_{\text{ra}}(\Theta)$ is simply the ratio of the ℓ_1 to ℓ_2 norms of the singular values of the weighted matrix $\sqrt{R}\Theta\sqrt{C}$. This measure can also be upper bounded by the rank of Θ: indeed, since R and C are full-rank, we always have

$$\beta^2_{\text{ra}}(\Theta) \leq \text{rank}(\sqrt{R}\Theta\sqrt{C}) \;=\; \text{rank}(\Theta),$$

with equality holding if all the non-zero singular values of $\sqrt{R}\Theta\sqrt{C}$ are identical.

3. Main Results and Their Consequences

We now turn to the statement of our main results, and discussion of their consequences. Section 3.1 is devoted to a result showing that a suitable form of restricted strong convexity holds for the random sampling operator $\mathfrak{X}_n$, as long as we restrict it to matrices Δ for which $\beta_{\rm ra}(\Delta)$ and $\alpha_{\rm sp}(\Delta)$ are not "overly large". In Section 3.2, we develop the consequences of the RSC condition for noisy matrix completion, and in Section 3.3, we prove that our error bounds are minimax-optimal up to logarithmic factors. In Section 3.4, we provide a detailed comparison of our results with past work.

3.1 Restricted Strong Convexity for Matrix Sampling

Introducing the convenient shorthand $d = \frac{1}{2}(d_r + d_c)$, let us define the constraint set

$$\mathfrak{C}(n; c_0) := \left\{ \Delta \in \mathbb{R}^{d_r \times d_c}, \Delta \neq 0 \mid \alpha_{\rm sp}(\Delta)\, \beta_{\rm ra}(\Delta) \leq \frac{1}{c_0 L} \sqrt{\frac{n}{d \log d}} \right\}, \tag{4}$$

where c_0 is a universal constant. Note that as the sample size n increases, this set allows for matrices with larger values of the spikiness and/or rank measures, $\alpha_{\rm sp}(\Delta)$ and $\beta_{\rm ra}(\Delta)$ respectively.

Theorem 1 *There are universal constants (c_0, c_1, c_2, c_3) such that as long as $n > c_3\, d \log d$, we have*

$$\frac{\|\mathfrak{X}_n(\Delta)\|_2}{\sqrt{n}} \geq \frac{1}{8} |||\Delta|||_{\omega(F)} \left\{ 1 - \frac{128\, \alpha_{\rm sp}(\Delta) L}{\sqrt{n}} \right\} \qquad \text{for all } \Delta \in \mathfrak{C}(n; c_0) \tag{5}$$

with probability greater than $1 - c_1 \exp(-c_2 d \log d)$.

Roughly speaking, this bound guarantees that the observation operator captures a substantial component of any matrix $\Delta \in \mathfrak{C}(n; c_0)$ that is not overly spiky. More precisely, as long as $\frac{128 L \alpha_{\rm sp}(\Delta)}{\sqrt{n}} \leq \frac{1}{2}$, the bound (5) implies that

$$\frac{\|\mathfrak{X}_n(\Delta)\|_2^2}{n} \geq \frac{1}{256} |||\Delta|||^2_{\omega(F)} \qquad \text{for any } \Delta \in \mathfrak{C}(n; c_0). \tag{6}$$

This bound can be interpreted in terms of *restricted strong convexity* (Negahban et al., 2009). In particular, given a vector $y \in \mathbb{R}^n$ of noisy observations, consider the quadratic loss function

$$\mathcal{L}(\Theta; y) = \frac{1}{2n} \|y - \mathfrak{X}_n(\Theta)\|_2^2.$$

Since the Hessian matrix of this function is given by $\mathfrak{X}_n{}^* \mathfrak{X}_n / n$, the bound (6) implies that the quadratic loss is strongly convex in a restricted set of directions Δ.

As discussed previously, the worst-case value of the "spikiness" measure is $\alpha_{\rm sp}(\Delta) = \sqrt{d_r d_c}$, achieved for a matrix that is zero everywhere except a single position. In this most degenerate of cases, the combination of the constraints $\frac{\alpha_{\rm sp}(\Delta)}{\sqrt{n}} < 1$ and the membership condition $\Delta \in \mathfrak{C}(n; c_0)$ imply that even for a rank one matrix (so that $\beta_{\rm ra}(\Delta) = 1$), we need sample size $n \gg d^2$ for Theorem 1 to provide a non-trivial result, as is to be expected.

3.2 Consequences for Noisy Matrix Completion

We now turn to some consequences of Theorem 1 for matrix completion in the noisy setting. In particular, assume that we are given n i.i.d. samples from the model (2), and let $\widehat{\Theta}$ be some estimate of the unknown matrix Θ^*. Our strategy is to exploit the lower bound (5) in application to the error matrix $\widehat{\Theta} - \Theta^*$, and accordingly, we need to ensure that it has relatively low-rank and spikiness. Based on this intuition, it is natural to consider the estimator

$$\widehat{\Theta} \in \arg \min_{|||\Theta|||_{\omega(\infty)} \leq \frac{\alpha^*}{\sqrt{d_r d_c}}} \big\{ \frac{1}{2n} \|y - \mathfrak{X}_n(\Theta)\|_2^2 + \lambda_n |||\Theta|||_{\omega(1)} \big\}, \tag{7}$$

where $\alpha^* \geq 1$ is a measure of spikiness, and the regularization parameter $\lambda_n > 0$ serves to control the nuclear norm of the solution. In the special case when both R and C are identity matrices (of the appropriate dimensions), this estimator is closely related to the standard one considered in past work on the problem, with the only difference between the additional ℓ_∞-norm constraint. In the more general weighted case, an M-estimator of the form (7) using the weighted nuclear norm (but without the elementwise constraint) was recently suggested by Salakhutdinov and Srebro (2010), who provided empirical results to show superiority of the weighted nuclear norm over the standard choice for the Netflix problem.

Past work on matrix completion has focused on the case of exactly low-rank matrices. Here we consider the more general setting of approximately low-rank matrices, including the exact setting as a particular case. We begin by stating a general upper bound that applies to any matrix Θ^*, and involves a natural decomposition into estimation and approximation error terms. The only relevant quantity is the signal-to-noise ratio, as measured by the ratio of the Frobenius norm of Θ^* to the noise variance, so that we allow the noise variance to be free, while assuming that $|||\widetilde{\Delta}|||_{\omega(F)}$ remains bounded.

Theorem 2 *Suppose that $n \geq Ld \log d$, and consider any solution $\widehat{\Theta}$ to the weighted SDP* (7) *using regularization parameter*

$$\lambda_n \geq 2\nu \, |||\frac{1}{n} \sum_{i=1}^{n} \xi_i R^{-\frac{1}{2}} X^{(i)} C^{-\frac{1}{2}} |||_{\mathrm{op}}, \tag{8}$$

and define $\lambda_n^* = \max\{\lambda_n, L\sqrt{\frac{d \log d}{n}}\}$. Then with probability greater than $1 - c_2 \exp(-c_2 \log d)$, for each $r = 1, \ldots, d_r$, the error $\widetilde{\Delta} = \widehat{\Theta} - \Theta^*$ satisfies

$$|||\widetilde{\Delta}|||_{\omega(F)}^2 \leq c_1 \, \alpha^* \, \lambda_n^* \left[\sqrt{r} |||\widetilde{\Delta}|||_{\omega(F)} + \sum_{j=r+1}^{d_r} \sigma_j(\sqrt{R}\Theta^*\sqrt{C}) \right] + \frac{c_1 (\alpha^* L)^2}{n}. \tag{9}$$

Apart from the trailing $O(n^{-1})$ the term, the bound (9) shows a natural splitting into two terms. The first can be interpreted as the *estimation error* associated with a rank r matrix, whereas the second term corresponds to *approximation error*, measuring how far $\sqrt{R}\Theta^*\sqrt{C}$ is from a rank r matrix. Of course, the bound holds for any choice of r, and in the corollaries to follow, we choose r optimally so as to balance the estimation and approximation error terms.

In order to provide concrete rates using Theorem 2, it remains to address two issues. First, we need to specify an explicit choice of λ_n by bounding the operator norm of the noise matrix

$\frac{1}{n}\sum_{i=1}^{n}\xi_i\sqrt{R}X^{(i)}\sqrt{C}$, and secondly, we need to understand how to choose the parameter r so as to achieve the tightest possible bound. When Θ^* is exactly low-rank, then it is obvious that we should choose $r = \operatorname{rank}(\Theta^*)$, so that the approximation error vanishes—more specifically, so that $\sum_{j=r+1}^{d_r}\sigma_j(\sqrt{R}\Theta^*\sqrt{C})_j = 0$. Doing so yields the following result:

Corollary 1 (Exactly low-rank matrices) *Suppose that the noise sequence $\{\xi_i\}$ is i.i.d., zero-mean and sub-exponential, and Θ^* has rank at most r, Frobenius norm at most 1, and spikiness at most $\alpha_{\mathrm{sp}}(\Theta^*) \leq \alpha^*$. If we solve the SDP (7) with $\lambda_n = 4\nu\sqrt{\frac{d\log d}{n}}$ then there is a numerical constant c_1' such that*

$$|||\widehat{\Theta} - \Theta^*|||^2_{\omega(F)} \leq c_1' \, (\nu^2 \vee L^2) \, (\alpha^*)^2 \, \frac{rd\log d}{n} + \frac{c_1(\alpha^* L)^2}{n} \tag{10}$$

with probability greater than $1 - c_2\exp(-c_3\log d)$.

Note that this rate has a natural interpretation: since a rank r matrix of dimension $d_r \times d_c$ has roughly $r(d_r + d_c)$ free parameters, we require a sample size of this order (up to logarithmic factors) so as to obtain a controlled error bound. An interesting feature of the bound (10) is the term $\nu^2 \vee 1 = \max\{\nu^2, 1\}$, which implies that we do not obtain exact recovery as $\nu \to 0$. As we discuss at more length in Section 3.4, under the mild spikiness condition that we have imposed, this behavior is unavoidable due to lack of identifiability within a certain radius, as specified in the set $\mathfrak{C}$. For instance, consider the matrix Θ^* and the perturbed version $\widetilde{\Theta} = \Theta^* + \frac{1}{\sqrt{d_r d_c}} e_1 e_1^T$. With high probability, we have $\mathfrak{X}_n(\Theta^*) = \mathfrak{X}_n(\widetilde{\Theta})$, so that the observations—even if they were noiseless—fail to distinguish between these two models. These types of examples, leading to non-identifiability, cannot be overcome without imposing fairly restrictive matrix incoherence conditions, as we discuss at more length in Section 3.4.

As with past work (Candès and Plan, 2010; Keshavan et al., 2010b), Corollary 1 applies to the case of matrices that have exactly rank r. In practical settings, it is more realistic to assume that the unknown matrix is not exactly low-rank, but rather can be well approximated by a matrix with low rank. One way in which to formalize this notion is via the ℓ_q-"ball" of matrices

$$\mathbb{B}_q(\rho_q) := \left\{ \Theta \in \mathbb{R}^{d_r \times d_c} \mid \sum_{j=1}^{\min\{d_r, d_c\}} |\sigma_j(\sqrt{R}\Theta\sqrt{C})|^q \leq \rho_q \right\}. \tag{11}$$

For $q = 0$, this set corresponds to the set of matrices with rank at most $r = \rho_0$, whereas for values $q \in (0, 1]$, it consists of matrices whose (weighted) singular values decay at a relatively fast rate. By applying Theorem 2 to this matrix family, we obtain the following corollary:

Corollary 2 (Estimation of near low-rank matrices) *Suppose that the noise $\{\xi_i\}$ is zero-mean and sub-exponential, Consider a matrix $\Theta^* \in \mathbb{B}_q(\rho_q)$ with spikiness at most $\alpha_{\mathrm{sp}}(\Theta^*) \leq \alpha^*$, and Frobenius norm at most one. With the same choice of λ_n as Corollary 1, there is a universal constant c_1' such that*

$$|||\widehat{\Theta} - \Theta^*|||^2_{\omega(F)} \leq c_1 \rho_q \left((\nu^2 \vee L^2)(\alpha^*)^2 \frac{d\log d}{n} \right)^{1 - \frac{q}{2}} + \frac{c_1(\alpha^* L)^2}{n} \tag{12}$$

with probability greater than $1 - c_2\exp(-c_3\log d)$.

Note that this result is a strict generalization of Corollary 1, to which it reduces in the case $q = 0$. (When $q = 0$, we have $\rho_0 = r$ so that the bound has the same form.) Note that the price that we pay for approximately low rank is a smaller exponent—namely, $1 - q/2$ as opposed to 1 in the case $q = 0$. The proof of Corollary 2 is based on a more subtle application of Theorem 2, one which chooses the effective rank r in the bound (9) so as to trade off between the estimation and approximation errors. In particular, the choice $r \asymp \rho_q \left(\frac{n}{d \log d}\right)^{q/2}$ turns out to yield the optimal trade-off, and hence the given error bound (12).

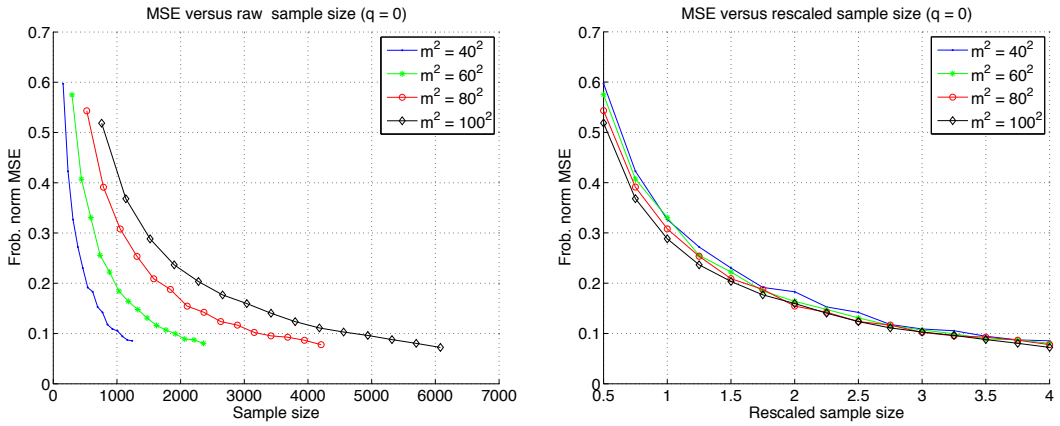
Although we have stated our results in terms of bounds on the weighted squared Frobenius norm $|||\Theta|||^2_{\omega(F)} = |||\sqrt{R}\Theta\sqrt{C}|||^2_F$, our assumed lower bound on the entries R and C implies that $|||\Theta|||^2_{\omega(F)} \geq \frac{|||\Theta|||^2_F}{L^2}$. Consequently, as long as each row and column is sampled a constant fraction of the time, our results also yield bounds on the Frobenius norm. In some applications, certain rows and columns might be heavily sampled, meaning that some entries of R and/or C could be relatively large. Since we require *only a lower bound* on the row/column sampling frequencies, our Frobenius norm bounds would not degrade if some rows and/or columns were heavily sampled. In contrast, a RIP-type analysis would not be valid in this setting, since heavy sampling means that the Frobenius norm could not be uniformly bounded from above.

In order to illustrate the sharpness of our theory, let us compare the predictions of our two corollaries to the empirical behavior of the M-estimator. In particular, we applied the nuclear norm SDP to simulated data, using Gaussian observation noise with variance $\nu^2 = 0.25$ and the uniform sampling model. In all cases, we solved the nuclear norm SDP using a non-smooth optimization procedure due to Nesterov (2007), via our own implementation in MATLAB. For a given problem size d, we ran $T = 25$ trials and computed the squared Frobenius norm error $|||\widehat{\Theta} - \Theta^*|||^2_F$ averaged over the trials.

Figure 1 shows the results in the case of exactly low-rank matrices ($q = 0$), with the matrix rank given by $r = \lceil \log^2(d) \rceil$. Panel (a) shows plots of the mean-squared Frobenius error versus the raw sample size, for three different problem sizes with the number of matrix elements sizes $d^2 \in \{40^2, 60^2, 80^2, 100^2\}$. These plots show that the M-estimator is consistent, since each of the curves decreases to zero as the sample size n increases. Note that the curves shift to the right as the matrix dimension d increases, reflecting the natural intuition that larger matrices require more samples. Based on the scaling predicted by Corollary 1, we expect that the mean-squared Frobenius error should exhibit the scaling $|||\widehat{\Theta} - \Theta^*|||^2_F \asymp \frac{rd \log d}{n}$. Equivalently, if we plot the MSE versus the *rescaled sample size* $N := \frac{n}{rd \log d}$, then all the curves should be relatively well aligned, and decay at the rate $1/N$. Panel (b) of Figure 1 shows the same simulation results re-plotted versus this rescaled sample size. Consistent with the prediction of Corollary 1, all four plots are now relatively well-aligned. Figure 2 shows the same plots for the case of approximately low-rank matrices ($q = 0.5$). Again, consistent with the prediction of Corollary 2, we see qualitatively similar behavior in the plots of the MSE versus sample size (panel (a)), and the rescaled sample size (panel (b)).

3.3 Information-theoretic Lower Bounds

The results of the previous section are achievable results, based on a particular polynomial-time estimator. It is natural to ask how these bounds compare to the fundamental limits of the problem, meaning the best performance achievable by any algorithm. As various authors have noted (Candès and Plan, 2010; Keshavan et al., 2010b), a parameter counting argument indicates that roughly $n \approx r(d_r + d_c)$ samples are required to estimate an $d_r \times d_c$ matrix with rank r. This calculation can



(a) (b)

Figure 1: Plots of the mean-squared error in Frobenius norm for $q = 0$. Each curve corresponds to a different problem size $d^2 \in \{40^2, 60^2, 80^2, 100^2\}$. (a) MSE versus the raw sample size n. As expected, the curves shift to the right as d increases, since more samples should be required to achieve a given MSE for larger problems. (b) The same MSE plotted versus the rescaled sample size $n/(rd\log d)$. Consistent with Corollary 1, all the plots are now fairly well-aligned.

be made more formal by metric entropy calculations for the Grassman manifold (e.g., Szarek, 1983); see also Rohde and Tsybakov (2011) for results on approximation numbers for the more general ℓ_q-balls of matrices. Such calculations, while accounting for the low-rank conditions, do *not* address the additional "spikiness" constraints that are essential to the setting of matrix completion. It is conceivable that these additional constraints could lead to a substantial volume reduction in the allowable class of matrices, so that the scalings suggested by parameter counting or metric entropy calculation for Grassman manifolds would be overly conservative.

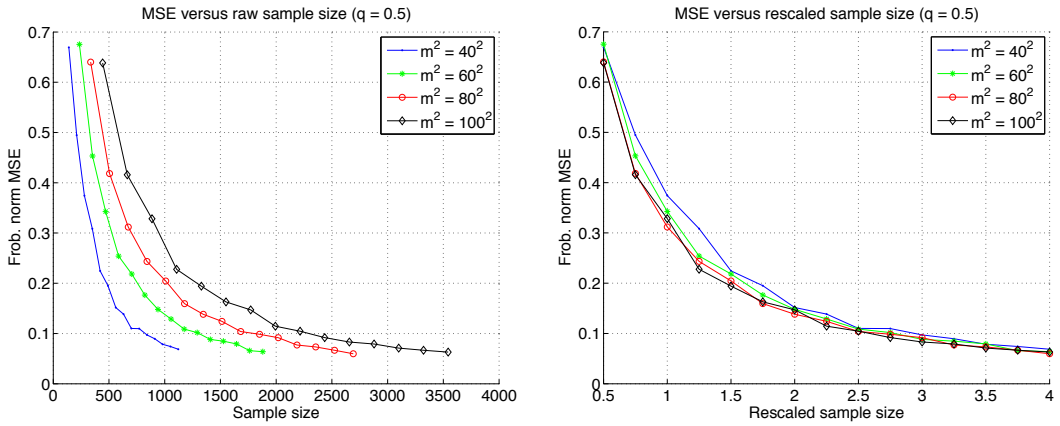
Accordingly, in this section, we provide a direct and constructive argument to lower bound the minimax rates of Frobenius norm over classes of matrices that are near low-rank and not overly spiky. This argument establishes that the bounds established in Corollaries 1 and 2 are sharp up to logarithmic factors, meaning that no estimator performs substantially better than the one considered here. More precisely, consider the matrix classes

$$\widetilde{\mathbb{B}}(\rho_q) = \left\{ \Theta \in \mathbb{R}^{d\times d} \mid \sum_{j=1}^{d} \sigma_j(\Theta)^q \leq \rho_q, \, \alpha_{\rm sp}(\Theta) \leq \sqrt{32\log d} \right\},$$

corresponding to square $d \times d$ matrices that are near low-rank (belonging to the ℓ_q-balls previously defined (11)), and have a logarithmic spikiness ratio. The following result applies to the *minimax risk* in Frobenius norm, namely the quantity

$$\mathfrak{M}_n(\widetilde{\mathbb{B}}(\rho_q)) := \inf_{\widetilde{\Theta}} \sup_{\Theta^* \in \widetilde{\mathbb{B}}(\rho_q)} \mathbb{E}\left[|||\widetilde{\Theta} - \Theta^*|||_F^2 \right],$$

where the infimum is taken over all estimators $\widetilde{\Theta}$ that are measurable functions of n samples.



(a) (b)

Figure 2: Plots of the mean-squared error in Frobenius norm for $q = 0.5$. Each curve corresponds to a different problem size $d^2 \in \{40^2, 60^2, 80^2, 100^2\}$. (a) MSE versus the raw sample size n. As expected, the curves shift to the right as d increases, since more samples should be required to achieve a given MSE for larger problems. (b) The same MSE plotted versus the rescaled sample size $n/(\rho_q^{\frac{1}{1-q/2}} d \log d)$. Consistent with Corollary 2, all the plots are now fairly well-aligned.

Theorem 3 *There is a universal numerical constant $c_5 > 0$ such that*

$$\mathfrak{M}_n(\widetilde{\mathbb{B}}(\rho_q)) \geq c_5 \min\left\{\rho_q \left(\frac{\nu^2 d}{n}\right)^{1-\frac{q}{2}}, \frac{\nu^2 d^2}{n}\right\}.$$

The term of primary interest in this bound is the first one—namely, $\rho_q \left(\frac{\nu^2 d}{n}\right)^{1-\frac{q}{2}}$. It is the dominant term in the bound whenever the ℓ_q-radius satisfies the bound

$$\rho_q \leq \left(\frac{\nu^2 d}{n}\right)^{\frac{q}{2}} d. \tag{13}$$

In the special case $q = 0$, corresponding the exactly low-rank case, the bound (13) always holds, since it reduces to requiring that the rank $r = \rho_0$ is less than or equal to d. In these regimes, Theorem 3 establishes that the upper bounds obtained in Corollaries 1 and 2 are minimax-optimal up to factors logarithmic in matrix dimension d.

3.4 Comparison to Other Work

We now turn to a detailed comparison of our bounds to those obtained in past work on noisy matrix completion, in particular the papers by Candès and Plan (2010) (hereafter CP) and Keshavan et al. (2010b) (hereafter KMO). Both papers considered only the case of exactly low-rank matrices, corresponding to the special case of $q = 0$ in our notation. Since neither paper provided results for the general case of near-low rank matrices, nor the general result (with estimation and approximation

errors) stated in Theorem 2, our discussion is mainly limited to comparing Corollary 1 to their results. So as to simplify discussion, we restate all results under the scalings used in this paper[1] (i.e., with $|||\Theta^*|||_F = 1$).

3.4.1 COMPARISON OF RATES

Under the strong incoherence conditions required for exact matrix recovery (see below for discussion), Theorem 7 in CP give an bound on $|||\widehat{\Theta} - \Theta^*|||_F$ that depends on the Frobenius norm of the potentially adversarial error matrix $\Xi \in \mathbb{R}^{d_1 \times d_2}$, as defined by the noise variables $[\Xi]_{j(i)\,k(i)} = \widetilde{\xi}_i$ in our case. In the special case of stochastic noise, under the observation model (1) and the scalings of our paper, as long as $n > d$, where $d = d_1 + d_2$—a condition certainly required for Frobenius norm consistency—we have $|||\Xi|||_F = \Theta(\nu\sqrt{n/d})$ with high probability. Given this scaling, the CP upper bound takes the form

$$|||\widehat{\Theta} - \Theta^*|||_F \lesssim \nu \left\{ \sqrt{d} + \frac{\sqrt{n}}{d} \right\}.$$

Note that if the noise standard deviation ν tends to zero while the sample size n, matrix size p and rank r all remain fixed, then this bound guarantees that the Frobenius error tends to zero. This behavior as $\nu \to 0$ is intuitively reasonable, given that their proof technique is an extrapolation from the case of exact recovery for noiseless observations ($\nu = 0$). However, note that for any fixed noise deviation $\nu > 0$, the first term increases to infinity as the matrix dimension d increases, whereas the second term actually grows as the sample size n increases. Consequently, the CP results do not guarantee statistical consistency, unlike the bounds proved here.

Turning to a setting with adversarial noise, suppose that the error vector has Frobenius norm at most δ. A modification of our analysis yields error bounds of the form $|||\widehat{\Theta} - \Theta^*|||_F \lesssim \{ \frac{d^2}{\sqrt{n}}\delta + \sqrt{\frac{rd\log d}{n}} \}$. In the setting of square matrices with $\delta \geq \sqrt{\frac{r\log d}{d}}$, our result yields an upper bound tighter by a factor of order $\sqrt{d}$ better than those presented in CP. Last, as pointed out by a reviewer, the CP analysis does yield bounds for approximately low-rank matrices, in particular by writing $\Theta^* = \Pi_r(\Theta^*) + \Delta$, where Π_r is the Frobenius norm projection onto the space of rank r matrices, and $\Delta = \Theta^* - \Pi_r(\Theta^*)$ is the approximation error. With this notation, their analysis guarantees error bounds of the form $\sqrt{d}|||\Delta|||_F$ with high probability, which is a weaker guarantee than our bound whenever $|||\Delta|||_F \geq c\sqrt{\frac{r\log d}{n}}$ and $n = \Omega(d\log d)$.

Keshavan et al. (2010b) analyzed alternative methods based on trimming and applying the SVD. For Gaussian noise, their methods guarantee bounds (with high probability) of the form

$$|||\widehat{\Theta} - \Theta^*|||_F \lesssim \nu \min\left\{ \alpha\sqrt{\frac{d_2}{d_1}}, \kappa^2(\Theta^*) \right\} \sqrt{\frac{rd_2}{n}}, \tag{14}$$

where d_2/d_1 is the aspect ratio of Θ^*, and $\kappa(\Theta^*) = \frac{\sigma_{\max}(\Theta^*)}{\sigma_{\min}(\Theta^*)}$ is the condition number of Θ^*. This result is more directly comparable to our Corollary 1; apart from the additional factor involving either the aspect ratio or the condition number, it is sharper since it does not involve the factor $\log d$ present in our bound. For a fixed noise standard deviation ν, the bound (14) guarantees statistical

1. The paper CP and KMO use two different sets of scaling, one with $|||\Theta^*|||_F = \Theta(d)$ and the other with $|||\Theta^*|||_F = \sqrt{r}$, so that some care is required in converting between results.

consistency as long as $\frac{rd_2}{n}$ tends to zero. The most significant differences are the presence of the aspect ratio d_2/d_1 or the condition number $\kappa(\Theta^*)$ in the upper bound (14). The aspect ratio is a quantity that can be as small as one, or as large as d_2, so that the pre-factor in the bound (14) can scale in a dimension-dependent way. Similarly, for any matrix with rank larger than one, the condition number can be made arbitrarily large. For instance, in the rank two case, define a matrix with $\sigma_{\max}(\Theta^*) = \sqrt{1-\delta^2}$ and $\sigma_{\min}(\Theta^*) = \delta$, and consider the behavior as $\delta \to 0$. In contrast, our bounds are invariant to both the aspect ratio and the condition number of Θ^*.

3.4.2 COMPARISON OF MATRIX CONDITIONS

We now turn to a comparison of the various *matrix incoherence assumptions* invoked in the analysis of CP and KMO, and comparison to our spikiness condition. As before, for clarity, we specialize our discussion to the square case ($d_r = d_c = d$), since the rectangular case is not essentially different. The matrix incoherence conditions are stated in terms of the singular value decomposition $\Theta^* = U\Sigma V^T$ of the target matrix. Here $U \in \mathbb{R}^{d\times r}$ and $V \in \mathbb{R}^{d\times r}$ are matrices of the left and right singular vectors respectively, satisfying $U^TU = V^TV = I_{r\times r}$, whereas $\Sigma \in \mathbb{R}^{r\times r}$ is a diagonal matrix of the singular values. The purpose of matrix incoherence is to enforce that the left and right singular vectors should not be aligned with the standard basis. Among other assumptions, the CP analysis imposes the incoherence conditions

$$\|UU^T - \frac{r}{d}I_{d\times d}\|_\infty \leq \mu\frac{\sqrt{r}}{d}, \qquad \|VV^T - \frac{r}{d}I_{d\times d}\|_\infty \leq \mu\frac{\sqrt{r}}{d}, \quad \text{and} \quad \|UV^T\|_\infty \leq \mu\frac{\sqrt{r}}{d}, \tag{15}$$

for some constant $\mu > 0$. Parts of the KMO analysis impose the related incoherence condition

$$\max_{j=1,\ldots,d} |UU^T|_{jj} \leq \mu_0\frac{r}{d}, \quad \text{and} \max_{j=1,\ldots,d} |VV^T|_{jj} \leq \mu_0\frac{r}{d}. \tag{16}$$

Both of these conditions ensure that the singular vectors are sufficiently "spread-out", so as not to be aligned with the standard basis.

A remarkable property of conditions (15) and (16) is that they exhibit *no dependence* on the singular values of Θ^*. If one is interested only in exact recovery in the noiseless setting, then this lack of dependence is reasonable. However, if approximate recovery is the goal—as is necessarily the case in the more realistic setting of noisy observations—then it is clear that a minimal set of sufficient conditions should also involve the singular values, as is the case for our spikiness measure $\alpha_{\text{sp}}(\Theta^*)$. The following example gives a concrete demonstration of an instance where our conditions are satisfied, so that approximate recovery is possible, whereas the incoherence conditions are violated.

Example. Let $\Gamma \in \mathbb{R}^{d\times d}$ be a positive semidefinite symmetric matrix with rank $r-1$, Frobenius norm $|||\Gamma|||_F = 1$ and $\|\Gamma\|_\infty \leq c_0/d$. For a scalar parameter $t > 0$, consider the matrix

$$\Theta^* := \Gamma + te_1e_1^T$$

where $e_1 \in \mathbb{R}^d$ is the canonical basis vector with one in its first entry, and zero elsewhere. By construction, the matrix Θ^* has rank at most r. Moreover, as long as $t = O(1/d)$, we are guaranteed that our spikiness measure satisfies the bound $\alpha_{\text{sp}}(\Theta^*) = O(1)$. Indeed, we have $|||\Theta^*|||_F \geq |||\Gamma|||_F - t = 1-t$, and hence

$$\alpha_{\text{sp}}(\Theta^*) = \frac{d\|\Theta^*\|_\infty}{|||\Theta^*|||_F} \leq \frac{d\big(\|\Gamma\|_\infty + t\big)}{1-t} \leq \frac{c_0 + dt}{1-t} = O(1).$$

Consequently, for any choice of Γ as specified above, Corollary 1 implies that the SDP will recover the matrix Θ^* up to a tolerance $O(\sqrt{\frac{rd\log d}{n}})$. This captures the natural intuition that "poisoning" the matrix Γ with the term $te_1^T e_1$ should have essentially no effect, as long as t is not too large.

On the other hand, suppose that we choose the matrix Γ such that its $r-1$ eigenvectors are orthogonal to e_1. In this case, we have $\Theta^* e_1 = te_1$, so that e_1 is also an eigenvector of Θ^*. Letting $U \in \mathbb{R}^{d\times r}$ be the matrix of eigenvectors, we have $e_1^T UU^T e_1 = 1$. Consequently, for any fixed μ (or μ_0) and rank $r \ll d$, conditions (15) and (16) are violated.

$\diamondsuit$

4. Proofs for Noisy Matrix Completion

We now turn to the proofs of our results. This section is devoted to the results that apply directly to noisy matrix completion, in particular the achievable result given in Theorem 2, its associated Corollaries 1 and 2, and the information-theoretic lower bound given in Theorem 3. The proof of Theorem 1 is provided in Section 5 to follow.

4.1 A Useful Transformation

We begin by describing a transformation that is useful both in these proofs, and the later proof of Theorem 1. In particular, we consider the mapping $\Theta \mapsto \Gamma := \sqrt{R}\Theta\sqrt{C}$, as well as the modified observation operator $\mathfrak{X}_n' : \mathbb{R}^{d\times d} \to \mathbb{R}^n$ with elements

$$[\mathfrak{X}_n'(\Gamma)]_i = \langle\langle \widetilde{X}^{(i)}, \Gamma\rangle\rangle, \quad \text{for } i = 1,2,\ldots,n,$$

where $\widetilde{X}^{(i)} := R^{-1/2} X^{(i)} C^{-1/2}$. Note that $\mathfrak{X}_n'(\Gamma) = \mathfrak{X}_n(\Theta)$ by construction, and moreover

$$|||\Gamma|||_F \;=\; |||\Theta|||_{\omega(F)}, \quad |||\Gamma|||_1 \;=\; |||\Theta|||_{\omega(1)}, \quad \text{and} \quad |||\Gamma|||_\infty \;=\; |||\Theta|||_{\omega(\infty)},$$

which implies that

$$\beta_{\mathrm{ra}}(\Theta) = \underbrace{\frac{|||\Gamma|||_1}{|||\Gamma|||_F}}_{\beta'_{\mathrm{ra}}(\Gamma)}, \quad \text{and} \quad \alpha_{\mathrm{sp}}(\Theta) = \underbrace{\frac{d\,\|\Gamma\|_\infty}{|||\Gamma|||_F}}_{\alpha'_{\mathrm{sp}}(\Gamma)}.$$

Based on this change of variables, let us define a modified version of the constraint set (4) as follows

$$\mathfrak{C}'(n;c_0) = \Big\{ 0 \neq \Gamma \in \mathbb{R}^{d\times d} \;\mid\; \alpha'_{\mathrm{sp}}(\Gamma)\,\beta'_{\mathrm{ra}}(\Gamma) \leq \frac{1}{c_0}\sqrt{\frac{n}{d\log d}} \Big\}. \tag{17}$$

In this new notation, the lower bound (5) from Theorem 1 can be re-stated as

$$\frac{\|\mathfrak{X}_n'(\Gamma)\|_2}{\sqrt{n}} \geq \frac{1}{8}|||\Gamma|||_F \big\{1 - \frac{128L\alpha'_{\mathrm{sp}}(\Gamma)}{\sqrt{n}}\big\} \quad \text{for all } \Gamma \in \mathfrak{C}'(n;c_0). \tag{18}$$

4.2 Proof of Theorem 2

We now turn to the proof of Theorem 2. Defining the estimate $\widehat{\Gamma} := \sqrt{R}\widehat{\Theta}\sqrt{C}$, we have

$$\widehat{\Gamma} \in \arg\min_{\|\Gamma\|_\infty \leq \frac{\alpha^*}{\sqrt{d_r d_c}}} \big\{ \frac{1}{2n}\|y - \mathfrak{X}_n'(\Gamma)\|_2^2 + \lambda_n |||\Gamma|||_1 \big\}, \tag{19}$$

and our goal is to upper bound the ordinary Frobenius norm $|||\widehat{\Gamma} - \Gamma^*|||_F$.

We now state a useful technical result. Parts (a) and (b) of the following lemma were proven by Recht et al. (2010), and Negahban and Wainwright (2011), respectively. We recall that we adopt the shorthand $\widehat{\Delta} = \widehat{\Gamma} - \Gamma^*$ throughout the analysis.

Lemma 1 *Let $(\widetilde{U}, \widetilde{V})$ represent a pair of r-dimensional subspaces of left and right singular vectors of Γ^*. Then there exists a matrix decomposition $\widehat{\Delta} = \widehat{\Delta}' + \widehat{\Delta}''$ of the error $\widehat{\Delta}$ such that*

(a) The matrix $\widehat{\Delta}'$ satisfies the constraint $\mathrm{rank}(\widehat{\Delta}') \leq 2r$, *and*

(b) Given the choice (8)*, the nuclear norm of $\widehat{\Delta}''$ is bounded as*

$$|||\widehat{\Delta}''|||_1 \;\leq\; 3|||\widehat{\Delta}'|||_1 + 4\sum_{j=r+1}^{d_r} \sigma_j(\Gamma^*). \tag{20}$$

Note that the bound (20), combined with triangle inequality, implies that

$$\begin{aligned} |||\widehat{\Delta}|||_1 \leq |||\widehat{\Delta}'|||_1 + |||\widehat{\Delta}''|||_1 \;&\leq\; 4|||\widehat{\Delta}'|||_1 + 4\sum_{j=r+1}^{d_r} \sigma_j(\Gamma^*) \\ &\leq 8\sqrt{r}|||\widehat{\Delta}|||_F + 4\sum_{j=r+1}^{d_r} \sigma_j(\Gamma^*) \end{aligned} \tag{21}$$

where the second inequality uses the fact that $\mathrm{rank}(\widehat{\Delta}') \leq 2r$.

We now split into two cases, depending on whether or not the error $\widehat{\Delta}$ belongs to the set $\mathfrak{C}'(n; c_0)$.

4.2.1 CASE 1

First suppose that $\widehat{\Delta} \notin \mathfrak{C}'(n; c_0)$. In this case, by the definition (17), we have

$$\begin{aligned} |||\widehat{\Delta}|||_F^2 &\leq c_0 L\big(\sqrt{d_r d_c}\|\widehat{\Delta}\|_\infty\big)|||\widehat{\Delta}|||_1\sqrt{\frac{d\log d}{n}} \\ &\leq 2c_0 L\alpha^*|||\widehat{\Delta}|||_1\sqrt{\frac{d\log d}{n}}, \end{aligned}$$

since $\|\widehat{\Delta}\|_\infty \leq \|\Gamma^*\|_\infty + \|\widehat{\Gamma}\|_\infty \leq \frac{2\alpha^*}{\sqrt{d_r d_c}}$. Now applying the bound (21), we obtain

$$|||\widehat{\Delta}|||_F^2 \leq 2c_0 L\alpha^*\sqrt{\frac{d\log d}{n}}\big\{8\sqrt{r}|||\widehat{\Delta}|||_F + 4\sum_{j=r+1}^{d_r}\sigma_j(\Gamma^*)\big\}. \tag{22}$$

4.2.2 CASE 2

Otherwise, we must have $\widehat{\Delta} \in \mathfrak{C}'(n; c_0)$. Recall the reformulated lower bound (18). On one hand, if $\frac{128L\alpha'_{\mathrm{sp}}(\widehat{\Delta})}{\sqrt{n}} > 1/2$, then we have

$$|||\widehat{\Delta}|||_F \leq \frac{256L\sqrt{d_r d_c}\|\widehat{\Delta}\|_\infty}{\sqrt{n}} \;\leq\; \frac{512L\alpha^*}{\sqrt{n}}. \tag{23}$$

On the other hand, if $\frac{128L\alpha'_{\rm sp}(\widehat{\Delta})}{\sqrt{n}} \leq 1/2$, then from the bound (18), we have

$$\frac{\|\mathfrak{X}_n{}'(\widehat{\Delta})\|_2}{\sqrt{n}} \geq \frac{|||\widehat{\Delta}|||_F}{16} \tag{24}$$

with high probability. Note that $\widehat{\Gamma}$ is optimal and Γ^* is feasible for the convex program (19), so that we have the basic inequality

$$\frac{1}{2n}\|y-\mathfrak{X}_n{}'(\widehat{\Gamma})\|_2^2+\lambda_n|||\widehat{\Gamma}|||_1 \leq \frac{1}{2n}\|y-\mathfrak{X}_n{}'(\Gamma^*)\|_2^2+\lambda_n|||\Gamma^*|||_1.$$

Some algebra then yields the inequality

$$\frac{1}{2n}\|\mathfrak{X}_n{}'(\widehat{\Delta})\|_2^2 \leq \nu\,\langle\langle \widehat{\Delta},\, \frac{1}{n}\sum_{i=1}^n \xi_i \widetilde{X}^{(i)}\rangle\rangle + \lambda_n|||\Gamma^*|||_1 - \lambda_n|||\Gamma^*+\widehat{\Delta}|||,$$

Substituting the lower bound (24) into this inequality yields

$$\frac{\|\widehat{\Delta}\|_F^2}{512} \leq \nu\,\langle\langle \widehat{\Delta},\, \frac{1}{n}\sum_{i=1}^n \xi_i \widetilde{X}^{(i)}\rangle\rangle + \lambda_n|||\Gamma^*|||_1 - \lambda_n|||\Gamma^*+\widehat{\Delta}|||.$$

From this point onwards, the proof is identical (apart from constants) to Theorem 1 in Negahban and Wainwright (2011), and we obtain that there is a numerical constant c_1 such that

$$|||\widehat{\Delta}|||_F^2 \leq c_1\,\alpha^*\,\lambda_n\Big\{\sqrt{r}|||\widehat{\Delta}|||_F + \sum_{j=r+1}^{d_r}\sigma_j(\Gamma^*)\Big\}. \tag{25}$$

4.2.3 PUTTING TOGETHER THE PIECES

Summarizing our results, we have shown that with high probability, one of the three bounds (22), (23) or (25) must hold. Since $\alpha^* \geq 1$, these claims can be summarized in the form

$$|||\widehat{\Delta}|||_F^2 \leq c_1\ \max\Big\{\lambda_n, \sqrt{\frac{d\log d}{n}}\Big\}\ \big[\sqrt{r}|||\widehat{\Delta}|||_F + \sum_{j=r+1}^{d_r}\sigma_j(\Gamma^*)\big].$$

for a universal positive constant c_1. Translating this result back to the original co-ordinate system ($\Gamma^* = \sqrt{R}\Theta^*\sqrt{C}$) yields the claim (9).

4.3 Proof of Corollary 1

When Θ^* (and hence $\sqrt{R}\Theta^*\sqrt{C}$) has rank $r < d_r$, then we have $\sum_{j=r+1}^{d_r}\sigma_j(\sqrt{R}\Theta^*\sqrt{C}) = 0$. Consequently, the bound (9) reduces to $|||\widetilde{\Delta}|||_{\omega(F)} \leq c_1\,\alpha^*\,\lambda_n^*\sqrt{r}$. To complete the proof, it suffices to show that

$$\mathbb{P}\big[|||\frac{1}{n}\sum_{i=1}^n \xi_i R^{-1/2}X^{(i)}C^{-1/2}|||_2 \geq c_1\,\nu\sqrt{\frac{d\log d}{n}}\big] \leq c_2\exp(-c_2 d\log d).$$

We do so via the Ahlswede-Winter matrix bound, as stated in Appendix F. Defining the random matrix $Y^{(i)} := \xi_i R^{-1/2} X^{(i)} C^{-1/2}$, we first note that ξ_i is sub-exponential with parameter 1, and $|R^{-1/2} X^{(i)} C^{-1/2}|$ has a single entry with magnitude at most $L\sqrt{d_r d_c}$, which implies that

$$\|Y^{(i)}\|_{\psi_1} \leq L\nu \sqrt{d_r d_c} \leq 2\nu L d.$$

(Here $\|\cdot\|_{\psi_1}$ denotes the Orlicz norm (Ledoux and Talagrand, 1991) of a random variable, as defined by the function $\psi_1(x) = \exp(x) - 1$; see Appendix F). Moreover, we have

$$\begin{aligned}
\mathbb{E}[(Y^{(i)})^T Y^{(i)}] &= \nu^2 \, \mathbb{E}\Big[\frac{d_r d_c}{R_{j(i)} C_{k(i)}} e_{k(i)} e_{j(i)}^T e_{j(i)} e_{k(i)}^T\Big] \\
&= \nu^2 \, \mathbb{E}\Big[\frac{d_r d_c}{R_{j(i)} C_{k(i)}} e_{k(i)} e_{k(i)}^T\Big] \\
&= \nu^2 \, d_r I_{d_c \times d_c}.
\end{aligned}$$

so that $|||\mathbb{E}[(Y^{(i)})^T Y^{(i)}]|||_2 \leq 2\nu^2 d$, recalling that $2d = d_r + d_c \geq d_r$. The same bound applies to $|||\mathbb{E}[Y^{(i)}(Y^{(i)})^T]|||_2$, so that applying Lemma 7 with $t = n\delta$, we conclude that

$$\mathbb{P}\big[|||\frac{1}{n}\sum_{i=1}^n \xi_i R^{-1/2} X^{(i)} C^{-1/2}|||_2 \geq \delta\big] \leq (d_r \times d_c) \max\big\{\exp(-n\delta^2/(16\nu^2 d), \exp(-\frac{n\delta}{4\nu L d})\big\}.$$

Since $\sqrt{d_r d_c} \leq d_r + d_c = 2d$, if we set $\delta^2 = c_1^2 \nu^2 \frac{d \log d}{n}$ for a sufficiently large constant c_1, the result follows. (Here we also use the assumption that $n = \Omega(Ld \log d)$, so that the term $\sqrt{\frac{d\log d}{n}}$ is dominant.)

4.4 Proof of Corollary 2

For this corollary, we need to determine an appropriate choice of r so as to optimize the bound (9). To ease notation, let us make use of the shorthand notation $\Gamma^* = \sqrt{R}\Theta^*\sqrt{C}$. With the singular values of Γ^* ordered in non-increasing order, fix some threshold $\tau > 0$ to be determined, and set $r = \max\{j \mid \sigma_j(\Gamma^*) > \tau\}$. This choice ensures that

$$\sum_{j=r+1}^{d_r} \sigma_j(\Gamma^*) = \tau \sum_{j=r+1}^{d_r} \frac{\sigma_j(\Gamma^*)}{\tau} \leq \tau \sum_{j=r+1}^{d_r} \big(\frac{\sigma_j(\Gamma^*)}{\tau}\big)^q \leq \tau^{1-q}\rho_q.$$

Moreover, we have $r\tau^q \leq \sum_{j=1}^r \big\{\sigma_j(\Gamma^*)\big\}^q \leq \rho_q$, which implies that $\sqrt{r} \leq \sqrt{\rho_q}\tau^{-q/2}$. Substituting these relations into the upper bound (9) leads to

$$|||\widetilde{\Delta}|||^2_{\omega(F)} \leq c_1 \, \alpha^* \, \lambda_n^* \, \big[\sqrt{\rho_q}\tau^{-q/2}|||\widetilde{\Delta}|||_{\omega(F)} + \tau^{1-q}\rho_q\big\}.$$

In order to obtain the sharpest possible upper bound, we set $\tau = \alpha^*\lambda_n^*$. Following some algebra, we find that there is a universal constant c_1 such that

$$|||\widetilde{\Delta}|||^2_{\omega(F)} \leq c_1 \rho_q \left((\alpha^*)^2(\lambda_n^*)^2\right)^{1-\frac{q}{2}}.$$

As in the proof of Corollary 1, it suffices to choose $\lambda_n = \Omega(\nu\sqrt{\frac{d\log d}{n}})$, so that $\lambda_n^* = O\sqrt{(\nu^2 + L)\frac{d\log d}{n}})$, from which the claim follows.

4.5 Proof of Theorem 3

Our proof of this lower bound exploits a combination of information-theoretic methods (Yu, 1997; Yang and Barron, 1999), which allow us to reduce to a multiway hypothesis test, and an application of the probabilistic method so as to construct a suitably large packing set. By Markov's inequality, it suffices to prove that

$$\sup_{\Theta^* \in \widetilde{\mathbb{B}}(\rho_q)} \mathbb{P}\left[|||\widehat{\Theta} - \Theta^*|||_F^2 \geq \frac{\delta^2}{4} \right] \geq \frac{1}{2}.$$

In order to do so, we proceed in a standard way—namely, by reducing the estimation problem to a testing problem over a suitably constructed packing set contained within $\widetilde{\mathbb{B}}(\rho_q)$. In particular, consider a set $\{\Theta^1, \ldots, \Theta^{M(\delta)}\}$ of matrices, contained within $\widetilde{\mathbb{B}}(\rho_q)$, such that $|||\Theta^k - \Theta^\ell|||_F \geq \delta$ for all $\ell \neq k$. To ease notation, we use M as shorthand for $M(\delta)$ through much of the argument. Suppose that we choose an index $V \in \{1, 2, \ldots, M\}$ uniformly at random (u.a.r.), and we are given observations $y \in \mathbb{R}^n$ from the observation model (2) with $\Theta^* = \Theta^V$. Then triangle inequality yields the lower bound

$$\sup_{\Theta^* \in \widetilde{\mathbb{B}}(\rho_q)} \mathbb{P}\left[|||\widehat{\Theta} - \Theta^*|||_F \geq \frac{\delta}{2} \right] \geq \mathbb{P}[\widehat{V} \neq V].$$

If we condition on $\mathfrak{X}_n$, a variant of Fano's inequality yields

$$\mathbb{P}[\widehat{V} \neq V \mid \mathfrak{X}_n] \geq 1 - \frac{\binom{M}{2}^{-1} \sum_{\ell \neq k} D(\Theta^k \,\|\, \Theta^\ell) + \log 2}{\log M}, \tag{26}$$

where $D(\Theta^k \,\|\, \Theta^\ell)$ denotes the Kullback-Leibler divergence between the distributions of $(y|\mathfrak{X}_n, \Theta^k)$ and $(y|\mathfrak{X}_n, \Theta^\ell)$. In particular, for additive Gaussian noise with variance ν^2, we have

$$D(\Theta^k \,\|\, \Theta^\ell) = \frac{1}{2\nu^2} \|\mathfrak{X}_n(\Theta^k) - \mathfrak{X}_n(\Theta^\ell)\|_2^2,$$

and moreover,

$$\mathbb{E}_{\mathfrak{X}_n}\big[D(\Theta^k \,\|\, \Theta^\ell)\big] = \frac{1}{2\nu^2} |||\Theta^k - \Theta^\ell|||_F^2.$$

Combined with the bound (26), we obtain the bound

$$\begin{aligned} \mathbb{P}[\widehat{V} \neq V] &= \mathbb{E}_{\mathfrak{X}_n}\big\{\mathbb{P}[\widehat{V} \neq V \mid \mathfrak{X}_n]\big\} \\ &\geq 1 - \frac{\binom{M}{2}^{-1} \sum_{\ell \neq k} \frac{n}{2\nu^2} |||\Theta^k - \Theta^\ell|||_F^2 + \log 2}{\log M}, \end{aligned} \tag{27}$$

The remainder of the proof hinges on the following technical lemma, which we prove in Appendix A.

Lemma 2 *Let $d \geq 10$ be a positive integer, and let $\delta > 0$. Then for each $r = 1, 2, \ldots, d$, there exists a set of d-dimensional matrices $\{\Theta^1, \ldots, \Theta^M\}$ with cardinality $M = \lfloor \frac{1}{4} \exp\left(\frac{rd}{128}\right) \rfloor$ such that each*

matrix has rank r, and moreover

$$\begin{aligned} |||\Theta^\ell|||_F &= \delta \qquad \text{for all } \ell = 1,2,\ldots,M, \\ |||\Theta^\ell - \Theta^k|||_F &\geq \delta \qquad \text{for all } \ell \neq k, \\ \alpha_{\mathrm{sp}}(\Theta^\ell) &\leq \sqrt{32 \log d} \quad \text{for all } \ell = 1,2,\ldots,M, \text{ and} \\ |||\Theta^\ell|||_2 &\leq \frac{4\delta}{\sqrt{r}} \quad \text{for all } \ell = 1,2,\ldots,M. \end{aligned}$$

We now show how to use this packing set in our Fano bound. To avoid technical complications, we assume throughout that $rd > 1024 \log 2$. Note that packing set from Lemma 2 satisfies $|||\Theta^k - \Theta^\ell|||_F \leq 2\delta$ for all $k \neq \ell$, and hence from Fano bound (27), we obtain

$$\begin{aligned} \mathbb{P}[\widehat{V} \neq V] &\geq 1 - \frac{2\frac{n\delta^2}{\nu^2} + \log 2}{\frac{rd}{128} - \log 4} \\ &\geq 1 - \frac{2\frac{n\delta^2}{\nu^2} + \log 2}{\frac{rd}{256}} \\ &\geq 1 - \frac{512\frac{n\delta^2}{\nu^2} + 256 \log 2}{rd}. \end{aligned}$$

If we now choose $\delta^2 = \frac{\nu^2}{2048}\frac{rd}{n}$, then

$$\mathbb{P}[\widehat{V} \neq V] \geq 1 - \frac{\frac{rd}{4} + 256 \log 2}{rd} \geq \frac{1}{2},$$

where the final inequality again uses the bound $rd \geq 1024 \log 2$.

In the special case $q = 0$, the proof is complete, since the elements Θ^ℓ all have rank $r = R_0$, and satisfy the bound $\alpha_{\mathrm{sp}}(\Theta^\ell) \leq \sqrt{32 \log d}$. For $q \in (0,1]$, consider the matrix class $\widetilde{\mathbb{B}}(\rho_q)$, and let us set $r = \min\{d, \lceil \rho_q \left(\frac{d}{n}\right)^{-\frac{q}{2}} \rceil\}$ in Lemma 2. With this choice, since each matrix Θ^ℓ has rank r, we have

$$\sum_{j=1}^{p} \sigma_i(\Theta^\ell)^q \leq r\left(\frac{\delta}{\sqrt{r}}\right)^q = r\left(\frac{1}{2048}\sqrt{\frac{d}{n}}\right)^q \leq \rho_q,$$

so that we are guaranteed that $\Theta^\ell \in \widetilde{\mathbb{B}}(\rho_q)$. Finally, we note that

$$\frac{rd}{n} \geq \min\left\{\rho_q\left(\frac{d}{n}\right)^{1-\frac{q}{2}}, \frac{d^2}{n}\right\},$$

so that we conclude that the minimax error is lower bounded by

$$\frac{1}{4096}\min\left\{\rho_q\left(\frac{\nu^2 d}{n}\right)^{1-\frac{q}{2}}, \frac{\nu^2 d^2}{n}\right\}$$

for dr sufficiently large. (At the expense of a worse pre-factor, the same bound holds for all $d \geq 10$.)

5. Proof of Theorem 1

We now turn to the proof that the sampling operator in weighted matrix completion satisfies restricted strong convexity over the set $\mathfrak{C}$, as stated in Theorem 1. In order to lighten notation, we prove the theorem in the case $d_r = d_c$. In terms of rates, this is a worst-case assumption, effectively amounting to replacing both d_r and d_c by the worst-case $\max\{d_r, d_c\}$. However, since our rates are driven by $d = \frac{1}{2}(d_r + d_c)$ and we have the inequalities

$$\frac{1}{2}\max\{d_r, d_c\} \ \leq \frac{1}{2}(d_r + d_c) \ \leq \max\{d_r, d_c\},$$

this change has only an effect on the constant factors. The proof can be extended to the general setting $d_r \neq d_c$ by appropriate modifications if these constant factors are of interest.

5.1 Reduction to Simpler Events

In order to prove Theorem 1, it is equivalent to show that, with high probability, we have

$$\frac{\|\mathfrak{X}_n'(\Gamma)\|_2}{\sqrt{n}} \geq \frac{1}{8}|||\Gamma|||_F - \frac{48L\, d\, \|\Gamma\|_\infty}{\sqrt{n}} \qquad \text{for all } \Gamma \in \mathfrak{C}'(n; c_0). \tag{28}$$

The remainder of the proof is devoted to studying the "bad" event

$$\mathcal{E}(\mathfrak{X}_n') := \left\{ \exists \quad \Gamma \in \mathfrak{C}'(n; c_0) \ \mid \ \left| \frac{\|\mathfrak{X}_n'(\Gamma)\|_2}{\sqrt{n}} - |||\Gamma|||_F \right| > \frac{7}{8}|||\Gamma|||_F + \frac{48L\, d\, \|\Gamma\|_\infty}{\sqrt{n}} \right\}.$$

Suppose that $\mathcal{E}(\mathfrak{X}_n')$ does *not* hold: then we have

$$\left| \frac{\|\mathfrak{X}_n'(\Gamma)\|_2}{\sqrt{n}} - |||\Gamma|||_F \right| \leq \frac{7}{8}|||\Gamma|||_F + \frac{48L\, d\, \|\Gamma\|_\infty}{\sqrt{n}} \qquad \text{for all } \Gamma \in \mathfrak{C}'(n; c_0),$$

which implies that the bound (28) holds. Consequently, in terms of the "bad" event, the claim of Theorem 1 is implied by the tail bound $\mathbb{P}[\mathcal{E}(\mathfrak{X}_n')] \leq 16\exp(-c' d \log d)$.

We now show that in order to establish a tail bound on $\mathcal{E}(\mathfrak{X}_n')$, it suffices to bound the probability of some simpler events $\mathcal{E}(\mathfrak{X}_n'; D)$, defined below. Since the definition of the set $\mathfrak{C}'(n; c_0)$ and event $\mathcal{E}(\mathfrak{X}_n')$ is invariant to rescaling of Γ, we may assume without loss of generality that $\|\Gamma\|_\infty = \frac{1}{d}$. The remaining degrees of freedom in the set $\mathfrak{C}'(n; c_0)$ can be parameterized in terms of the quantities $D = |||\Gamma|||_F$ and $\rho = |||\Gamma|||_1$. For any $\Gamma \in \mathfrak{C}'(n; c_0)$ with $\|\Gamma\|_\infty = \frac{1}{d}$ and $|||\Gamma|||_F \leq D$, we have $|||\Gamma|||_1 \leq \rho(D)$, where

$$\rho(D) := \frac{D^2}{c_0 L \sqrt{\frac{d \log d}{n}}}.$$

For each radius $D > 0$, consider the set

$$\mathfrak{B}(D) := \left\{ \Gamma \in \mathfrak{C}'(n; c_0) \ \mid \ \|\Gamma\|_\infty = \frac{1}{d}, \ |||\Gamma|||_F \leq D, \ |||\Gamma|||_1 \leq \rho(D) \right\}, \tag{29}$$

and the associated event

$$\mathcal{E}(\mathfrak{X}_n'; D) := \left\{ \exists \quad \Gamma \in \mathfrak{B}(D) \ \mid \ \left| \frac{\|\mathfrak{X}_n'(\Gamma)\|_2}{\sqrt{n}} - |||\Gamma|||_F \right| \geq \frac{3}{4}D + \frac{48L}{\sqrt{n}} \right\}.$$

The following lemma shows that it suffices to upper bound the probability of the event $\mathcal{E}(\mathfrak{X}_n'; D)$ for each fixed $D > 0$.

Lemma 3 *Suppose that are universal constants (c_1, c_2) such that*

$$\mathbb{P}[\mathcal{E}(\mathfrak{X}_n{}'; D)] \leq c_1 \exp(-c_2 n D^2) \tag{30}$$

for each fixed $D > 0$. Then there is a universal constant c_2' such that

$$\mathbb{P}[\mathcal{E}(\mathfrak{X}_n{}')] \leq c_1 \frac{\exp(-c_2' d \log d)}{1 - \exp(-c_2' d \log d)}.$$

The proof of this claim, provided in Appendix B, follows by a peeling argument.

5.2 Bounding the Probability of $\mathcal{E}(\mathfrak{X}_n{}'; D)$

Based on Lemma 3, it suffices to prove the tail bound (30) on the event $\mathcal{E}(\mathfrak{X}_n{}'; D)$ for each fixed $D > 0$. Let us define

$$Z_n(D) := \sup_{\Gamma \in \overline{\mathfrak{B}}(D)} \left| \frac{\|\mathfrak{X}_n{}'(\Gamma)\|_2}{\sqrt{n}} - |||\Gamma|||_F \right|,$$

where

$$\overline{\mathfrak{B}}(D) := \big\{ \Gamma \in \mathfrak{C}'(n; c_0) \mid \|\Gamma\|_\infty \leq \frac{1}{d}, \, |||\Gamma|||_F \leq D, \, |||\Gamma|||_1 \leq \rho(D) \big\}. \tag{31}$$

(The only difference from $\mathfrak{B}(D)$ is that we have relaxed to the inequality $\|\Gamma\|_\infty \leq \frac{1}{d}$.) In the remainder of this section, we prove that there are universal constants (c_1, c_2) such that

$$\mathbb{P}\big[Z_n(D) \geq \frac{3}{4} D + \frac{48 L}{\sqrt{n}}\big] \leq c_1 \exp(-c_2 \frac{n D^2}{L^2}) \quad \text{for each fixed } D > 0. \tag{32}$$

This tail bound means that the condition of Lemma 3 is satisfied, and so completes the proof of Theorem 1.

In order to prove (32), we begin with a discretization argument. Let $\Gamma^1, \ldots, \Gamma^{N(\delta)}$ be a δ-covering of $\overline{\mathfrak{B}}(D)$ in the Frobenius norm. By definition, given an arbitrary $\Gamma \in \overline{\mathfrak{B}}(D)$, there exists some index $k \in \{1, \ldots, N(\delta)\}$ and a matrix $\Delta \in \mathbb{R}^{d \times d}$ with $|||\Delta|||_F \leq \delta$ such that $\Gamma = \Gamma^k + \Delta$. Therefore, we have

$$\begin{aligned}
\frac{\|\mathfrak{X}_n{}'(\Gamma)\|_2}{\sqrt{n}} - |||\Gamma|||_F &= \frac{\|\mathfrak{X}_n{}'(\Gamma^k + \Delta)\|_2}{\sqrt{n}} - |||\Gamma^k + \Delta|||_F \\
&\leq \frac{\|\mathfrak{X}_n{}'(\Gamma^k)\|_2}{\sqrt{n}} + \frac{\|\mathfrak{X}_n{}'(\Delta)\|_2}{\sqrt{n}} - |||\Gamma^k|||_F + |||\Delta|||_F \\
&\leq \Big| \frac{\|\mathfrak{X}_n{}'(\Gamma^k)\|_2}{\sqrt{n}} - |||\Gamma^k|||_F \Big| + \frac{\|\mathfrak{X}_n{}'(\Delta)\|_2}{\sqrt{n}} + \delta,
\end{aligned}$$

where we have used the triangle inequality. Following the same steps establishes that this inequality holds for the absolute value of the difference.

Moreover, since $\Delta = \Gamma^k - \Gamma$ with both Γ^k and Γ belonging to $\overline{\mathfrak{B}}(D)$, we have $|||\Delta|||_1 \leq 2\rho(D)$ and $\|\Delta\|_\infty \leq \frac{2}{d}$, where we have used the definition (29). Putting together the pieces, we conclude that

$$Z_n(D) \quad \leq \quad \delta \quad + \quad \max_{k=1,\ldots,N(\delta)} \Big| \frac{\|\mathfrak{X}_n{}'(\Gamma^k)\|_2}{\sqrt{n}} \quad - \quad |||\Gamma^k|||_F \Big| \quad + \quad \sup_{\Delta \in \mathfrak{D}(\delta, R)} \big| \frac{\|\mathfrak{X}_n{}'(\Delta)\|_2}{\sqrt{n}} \big|, \tag{33}$$

where

$$\mathfrak{D}(\delta, R) := \big\{\Delta \in \mathbb{R}^{d\times d} \mid |||\Delta|||_F \leq \delta,\, |||\Delta|||_1 \leq 2\rho(D),\, \|\Delta\|_\infty \leq \frac{2}{d}\big\}.$$

Note that the bound (33) holds for any choice of $\delta > 0$. We establish the tail bound (32) with the choice $\delta = D/8$, and using the following two lemmas. The first lemma provides control of the maximum over the covering set:

Lemma 4 *As long $d \geq 10$, we have*

$$\max_{k=1,\ldots,N(D/8)} \Big| \frac{\|\mathfrak{X}_n'(\Gamma^k)\|_2}{\sqrt{n}} - |||\Gamma^k|||_F \Big| \leq \frac{D}{8} + \frac{48L}{\sqrt{n}}$$

with probability greater than $1 - c\exp\big(-\frac{nD^2}{2048L^2}\big)$.

See Appendix C for the proof of this claim.

Our second lemma, proved in Appendix D, provides control over the final term in the upper bound (33).

Lemma 5

$$\sup_{\Delta\in\mathfrak{D}(\frac{D}{8},R)} \Big| \frac{\|\mathfrak{X}_n'(\Delta)\|_2}{\sqrt{n}} \Big| \leq \frac{D}{2}$$

with probability at least $1 - 2\exp\big(-\frac{nD^2}{8192L^2}\big)$.

Combining these two lemmas with the upper bound (33) with $\delta = D/8$, we obtain

$$\begin{aligned} Z_n(D) &\leq \frac{D}{8} + \frac{D}{8} + \frac{48L}{\sqrt{n}} + \frac{D}{2} \\ &\leq \frac{3D}{4} + \frac{48L}{\sqrt{n}} \end{aligned}$$

with probability at least $1 - 4\exp\big(-\frac{nD^2}{8192}\big)$, thereby establishing the tail bound (32) and completing the proof of Theorem 1.

6. Discussion

In this paper, we have established error bounds for the problem of weighted matrix completion based on partial and noisy observations. We proved both a general result, one which applies to any matrix, and showed how it yields corollaries for both the cases of exactly low-rank and approximately low-rank matrices. A key technical result is establishing that the matrix sampling operator satisfies a suitable form of restricted strong convexity (Negahban et al., 2009) over a set of matrices with controlled rank and spikiness. Since more restrictive properties such as RIP do not hold for matrix completion, this RSC ingredient is essential to our analysis. Our proof of the RSC condition relied on a number of techniques from empirical process and random matrix theory, including concentration of measure, contraction inequalities and the Ahlswede-Winter bound. Using information-theoretic

methods, we also proved that up to logarithmic factors, our error bounds cannot be improved upon by any algorithm, showing that our method is essentially minimax-optimal.

There are various open questions that remain to be studied. Although our analysis applies to both uniform and non-uniform sampling models, it is limited to the case where each row (or column) is sampled with a certain probability. It would be interesting to consider extensions to settings in which the sampling probability differed from entry to entry, as investigated empirically by Salakhutdinov and Srebro (2010). Although we have focused on least-squares losses in this paper, the notion of restricted strong convexity applies to more general loss functions. Indeed, it should be possible to combine the results of this paper with Proposition 2 in Negahban et al. (2009) so as to obtain bounds for matrix completion with general losses. Lastly, although this paper has focused on statistical consequences, the RSC property also has implications for the fast convergence of gradient-type algorithms for solving matrix completion problems (Agarwal et al., 2011).

Acknowledgments

SN and MJW were partially supported by NSF grants DMS-0907632, NSF-CDI-0941742, and Air Force Office of Scientific Research AFOSR-09NL184. SN was also partially supported by a Yahoo! KSC Award.

Appendix A. Proof of Lemma 2

We proceed via the probabilistic method, in particular by showing that a random procedure succeeds in generating such a set with probability at least $1/2$. Let $M' = \exp\left(\frac{rd}{128}\right)$, and for each $\ell = 1, \ldots, M'$, we draw a random matrix $\widetilde{\Theta}^\ell \in \mathbb{R}^{d \times d}$ according to the following procedure:

(a) For rows $i = 1, \ldots, r$ and for each column $j = 1, \ldots, d$, choose each $\widetilde{\Theta}^\ell_{ij} \in \{-1, +1\}$ uniformly at random, independently across (i, j).

(b) For rows $i = r+1, \ldots, d$, set $\widetilde{\Theta}^\ell_{ij} = 0$.

We then let $Q \in \mathbb{R}^{d \times d}$ be a random unitary matrix, and define $\Theta^\ell = \frac{\delta}{\sqrt{rd}} Q\widetilde{\Theta}^\ell$ for all $\ell = 1, \ldots, M'$. The remainder of the proof analyzes the random set $\{\Theta^1, \ldots, \Theta^{M'}\}$, and shows that it contains a subset of size at least $M = M'/4$ that has properties (a) through (d) with probability at least $1/2$.

By construction, each matrix $\widetilde{\Theta}^\ell$ has rank at most r, and Frobenius norm $|||\widetilde{\Theta}^\ell|||_F = \sqrt{rd}$. Since Q is unitary, the rescaled matrices Θ^ℓ have Frobenius norm $|||\Theta^\ell|||_F = \delta$. We now prove that

$$|||\Theta^\ell - \Theta^k|||_F \geq \delta \qquad \text{for all } \ell \neq k$$

with probability at least $7/8$. Again, since Q is unitary, it suffices to show that $|||\widetilde{\Theta}^\ell - \widetilde{\Theta}^k|||_F \geq \sqrt{rd}$ for any pair $\ell \neq k$. We have

$$\frac{1}{rd}|||\widetilde{\Theta}^k - \widetilde{\Theta}^\ell|||_F^2 = \frac{1}{rd}\sum_{i=1}^{r}\sum_{j=1}^{d}\left(\widetilde{\Theta}^\ell_{ij} - \widetilde{\Theta}^k_{ij}\right)^2.$$

This is a sum of rd i.i.d. variables, each bounded by 4. The mean of the sum is 2, so that the Hoeffding bound implies that

$$\mathbb{P}\big[\frac{1}{rd}|||\widetilde{\Theta}^k - \widetilde{\Theta}^\ell|||_F^2 \leq 2 - t\big] \leq 2\exp\left(-rdt^2/32\right).$$

Since there are less than $(M')^2$ pairs of matrices in total, setting $t = 1$ yields

$$\mathbb{P}\Big[\min_{\ell,k=1,\ldots,M'} \frac{|||\widetilde{\Theta}^\ell - \widetilde{\Theta}^k|||_F^2}{rd} \geq 1\Big] \geq 1 - 2\exp\big(-\frac{rd}{32} + 2\log M'\big) \geq \frac{7}{8},$$

where we have used the facts $\log M' = \frac{rd}{128}$ and $d \geq 10$. Recalling the definition of Θ^ℓ, we conclude that

$$\mathbb{P}\Big[\min_{\ell,k=1,\ldots,M'} |||\Theta^\ell - \Theta^k|||_F^2 \geq \delta^2\Big] \geq \frac{7}{8}.$$

We now establish bounds on $\alpha_{\rm sp}(\Theta^\ell)$ and $|||\Theta^\ell|||_2$. We first prove that for any fixed index $\ell \in \{1, 2, \ldots, M'\}$, our construction satisfies

$$\mathbb{P}\Big[\alpha_{\rm sp}(\Theta^\ell) \leq \sqrt{32\log d}\Big] \geq \frac{3}{4}. \tag{34}$$

Indeed, for any pair of indices (i, j), we have $|\Theta^\ell_{ij}| = |\langle q_i, v_j\rangle|$, where $q_i \in \mathbb{R}^d$ is drawn from the uniform distribution over the d-dimensional sphere, and $\|v_j\|_2 = \sqrt{r}\,\frac{\delta}{\sqrt{rd}} = \frac{|||\Theta^\ell|||_F}{\sqrt{d}}$. By Levy's theorem for concentration on the sphere (Ledoux, 2001), we have

$$\mathbb{P}\big[|\langle q_i, v_j\rangle| \geq t\big] \leq 2\exp\big(-\frac{d^2}{8\,|||\Theta^\ell|||_F^2}t^2\big).$$

Setting $t = s/d$ and taking the union bound over all d^2 indices, we obtain

$$\mathbb{P}\big[d\,\|\Theta^\ell\|_\infty \geq s\big] \leq 2\exp\bigg(-\frac{1}{8\,|||\Theta^\ell|||_F^2}s^2 + 2\log d\bigg).$$

This probability is less than $1/2$ for $s = |||\Theta^\ell|||_F\sqrt{32\log d}$ and $d \geq 2$, which establishes the intermediate claim (34).

Finally, we turn to property (d). For each fixed ℓ, by definition of Θ^ℓ and the unitary nature of Q, we have $|||\Theta^\ell|||_2 = \frac{\delta}{\sqrt{rd}}|||U|||_2$, where $U \in \{-1, +1\}^{r\times d}$ is a random matrix with i.i.d. Rademacher (and hence sub-Gaussian) entries. Known results on sub-Gaussian matrices (Vershynin, 2012) yield

$$\mathbb{P}\Big[\frac{\delta}{\sqrt{rd}}|||U|||_2 \leq \frac{2\delta}{\sqrt{rd}}\big(\sqrt{r} + \sqrt{d}\big)\Big] \geq 1 - 2\exp\big(-\frac{1}{4}(\sqrt{r} + \sqrt{d})^2\big) \geq \frac{3}{4}$$

for $d \geq 10$. Since $r \leq d$, we conclude that

$$\mathbb{P}\Big[|||\Theta^\ell|||_2 \leq \frac{4\delta}{\sqrt{r}}\Big] \geq \frac{3}{4}. \tag{35}$$

By combining the bounds (34) and (35), we find that for each fixed $\ell = 1, \ldots, M'$, we have

$$\mathbb{P}\bigg[|||\Theta^\ell|||_2 \leq \frac{4\delta}{\sqrt{r}},\ \frac{\alpha_{\rm sp}(\Theta^\ell)}{|||\Theta|||_F} \leq \sqrt{32\log d}\bigg] \geq \frac{1}{2}. \tag{36}$$

Consider the event $\mathcal{E}$ that there exists a subset $S \subset \{1, \ldots, M'\}$ of cardinality $M = \frac{1}{4}M'$ such that

$$|||\Theta^\ell|||_2 \leq 4\frac{\delta}{\sqrt{n}}, \quad \text{and} \quad \frac{\alpha_{\rm sp}(\Theta^\ell)}{|||\Theta|||_F} \leq \sqrt{32\log d} \qquad \text{for all } \ell \in S.$$

By the bound (36), we have

$$\mathbb{P}[\mathcal{E}] \geq \sum_{k=M}^{M'} \binom{M'}{k} (1/2)^k.$$

Since we have chosen $M < M'/2$, we are guaranteed that $\mathbb{P}[\mathcal{E}] \geq 1/2$, thereby completing the proof.

Appendix B. Proof of Lemma 3

We first observe that for any $\Gamma \in \mathfrak{C}'(n; c_0)$ with $\|\Gamma\|_\infty = \frac{1}{d}$, we have

$$|||\Gamma|||_F^2 \geq c_0 \, |||\Gamma|||_1 \sqrt{\frac{d \log d}{n}} \geq c_0 |||\Gamma|||_F \sqrt{\frac{d \log d}{n}},$$

whence $|||\Gamma|||_F \geq c_0 \sqrt{\frac{d \log d}{n}}$. Accordingly, recalling the definition (29), it suffices to restrict our attention to sets $\mathfrak{B}(D)$ with $D \geq \mu := c_0 \sqrt{\frac{d \log d}{n}}$. For $\ell = 1, 2, \ldots$ and $\alpha = \frac{7}{6}$, define the sets

$$\mathbb{S}_\ell := \big\{ \Gamma \in \mathfrak{C}'(n; c_0) \mid \|\Gamma\|_\infty = \frac{1}{d}, \quad \alpha^{\ell-1}\mu \leq |||\Gamma|||_F \leq \alpha^\ell \mu, \text{ and } |||\Gamma|||_1 \leq \rho(\alpha^\ell \mu) \big\}. \tag{37}$$

From the definition (29), note that by construction, we have $\mathbb{S}_\ell \subset \mathfrak{B}(\alpha^\ell \mu)$.

Now if the event $\mathcal{E}(\mathfrak{X}_n{}')$ holds for some matrix Γ, then this matrix Γ must belong to some set $\mathbb{S}_\ell$. When $\Gamma \in \mathbb{S}_\ell$, then we are guaranteed the existence of a matrix $\Gamma \in \mathfrak{B}(\alpha^\ell \mu)$ such that

$$\begin{aligned} \big| \frac{\|\mathfrak{X}_n{}'(\Gamma)\|_2}{\sqrt{n}} - |||\Gamma|||_F \big| &> \frac{7}{8} |||\Gamma|||_F + \frac{48L}{\sqrt{n}} \\ &\geq \frac{7}{8} \alpha^{\ell-1} \mu + \frac{48L}{\sqrt{n}} \\ &= \frac{3}{4} \alpha^\ell \mu + \frac{48L}{\sqrt{n}}, \end{aligned}$$

where the final equality follows since $\alpha = 7/6$. Thus, we have shown that when the violating matrix $\Gamma \in \mathbb{S}_\ell$, then event $\mathcal{E}(\mathfrak{X}_n{}'; \alpha^\ell \mu)$ must hold. Since any violating matrix must fall into some set $\mathbb{S}_\ell$, the union bound implies that

$$\begin{aligned} \mathbb{P}[\mathcal{E}(\mathfrak{X}_n{}')] &\leq \sum_{\ell=1}^{\infty} \mathbb{P}[\mathcal{E}(\mathfrak{X}_n{}'; \alpha^\ell \mu)] \\ &\leq c_1 \sum_{\ell=1}^{\infty} \exp\big(-c_2 n \alpha^{2\ell} \mu^2\big) \\ &\leq c_1 \sum_{\ell=1}^{\infty} \exp\big(-2 c_2 \log(\alpha) \, \ell \, n \mu^2\big) \\ &\leq c_1 \frac{\exp(-c_2' n \mu^2)}{1 - \exp(-c_2' n \mu^2)}. \end{aligned}$$

Since $n\mu^2 = \Omega(d \log d)$, the claim follows.

Appendix C. Proof of Lemma 4

For a fixed matrix Γ, define the function $F_\Gamma(\mathfrak{X}_n') = \frac{1}{\sqrt{n}}\|\mathfrak{X}_n'(\Gamma)\|_2$. We prove the lemma in two parts: first, we establish that for any fixed Γ, the function F_Γ satisfies the tail bound

$$\mathbb{P}\big[|F_\Gamma(\mathfrak{X}_n') - |||\Gamma|||_F| \geq \delta + \frac{48L}{\sqrt{n}}\big] \leq 4\exp\big(-\frac{n\delta^2}{4L^2}\big). \tag{38}$$

We then show that there exists a δ-covering of $\overline{\mathfrak{B}}(D)$ such that

$$\log N(\delta) \leq 36\big(\rho(D)/\delta\big)^2\ d. \tag{39}$$

Combining the tail bound (38) with the union bound, we obtain

$$\begin{aligned}\mathbb{P}\big[\max_{k=1,\ldots,N(\delta)} |F_\Gamma(\mathfrak{X}_n') - |||\Gamma^k|||_F| \geq \delta + \frac{16L}{\sqrt{n}}\big] &\leq 4\exp\big(-\frac{n\delta^2}{4L^2} + \log N(\delta)\big) \\ &\leq 4\exp\Big\{-\frac{n\delta^2}{4L^2} + 36\big(\rho(D)/\delta\big)^2\, d\Big\}\end{aligned}$$

where the final inequality uses the bound (39). Since Lemma 4 is based on the choice $\delta = D/8$, it suffices to show that

$$\begin{aligned}\frac{nD^2}{512L^2} &\geq 36\left(\rho(D)/(D/8)\right)^2 d \\ &= 36\left(\frac{8D}{c_0 L}\sqrt{\frac{n}{d\log d}}\right)^2 d \\ &= \frac{2304D^2}{c_0^2L^2}\,\frac{n}{\log d}.\end{aligned}$$

Noting that the terms involving D^2, L^2, and n both cancel out, we see that for any fixed c_0, this inequality holds once $\log d$ is sufficiently large. By choosing c_0 sufficiently large, we can ensure that it holds for all $d \geq 2$.

It remains to establish the two intermediate claims (38) and (39).

C.1 Upper Bounding the Covering Number (39)

We start by proving the upper bound (39) on the covering number. To begin, let $\widetilde{N}(\delta)$ denote the δ-covering number (in Frobenius norm) of the nuclear norm ball $\mathbb{B}_1(\rho(D)) = \big\{\Delta \in \mathbb{R}^{d\times d} \mid |||\Delta|||_1 \leq \rho(D)\big\}$, and let $N(\delta)$ be the covering number of the set $\overline{\mathfrak{B}}(D)$. We first claim that $N(\delta) \leq \widetilde{N}(\delta)$. Let $\{\Gamma^1,\ldots,\Gamma^{\widetilde{N}(\delta)}\}$ be a δ-cover of $\mathbb{B}_1(\rho(D))$, From Equation (31), note that the set $\overline{\mathfrak{B}}(D)$ is contained within $\mathbb{B}_1(\rho(D))$; in particular, it is obtained by intersecting the latter set with the set

$$\mathcal{S} := \big\{\Delta \in \mathbb{R}^{d\times d} \mid \|\Delta\|_\infty \leq \frac{1}{d},\ |||\Delta|||_F \leq D\big\}.$$

Letting $\Pi_{\mathcal{S}}$ denote the projection operator under Frobenius norm onto this set, we claim that $\{\Pi_{\mathcal{S}}(\Gamma^j), j = 1, \ldots, \widetilde{N}(\delta)\}$ is a δ-cover of $\overline{\mathfrak{B}}(D)$. Indeed, since $\mathcal{S}$ is non-empty, closed and convex, the projection operator is non-expansive (Bertsekas, 1995), and thus for any $\Gamma \in \overline{\mathfrak{B}}(D) \subset \mathcal{S}$, we have

$$|||\Pi_{\mathcal{S}}(\Gamma^j) - \Gamma|||_F \;=\; |||\Pi_{\mathcal{S}}(\Gamma^j) - \Pi_{\mathcal{S}}(\Gamma)|||_F \leq |||\Gamma^j - \Gamma|||_F,$$

which establishes the claim.

We now upper bound $\widetilde{N}(\delta)$. Let $G \in \mathbb{R}^{d \times d}$ be a random matrix with i.i.d. $N(0,1)$ entries. By Sudakov minoration (cf. Theorem 5.6 in Pisier, 1989), we have

$$\begin{aligned}\sqrt{\log \widetilde{N}(\delta)} \;&\leq \frac{3}{\delta}\, \mathbb{E}\big[\sup_{|||\Delta|||_1 \leq \rho(D)} \langle\langle G, \Delta \rangle\rangle\big] \\ &\leq \frac{3\rho(D)}{\delta}\, \mathbb{E}\big[|||G|||_2\big],\end{aligned}$$

where the second inequality follows from the duality between the nuclear and operator norms. From known results on the operator norms of Gaussian random matrices (Davidson and Szarek, 2001), we have the upper bound $\mathbb{E}[|||G|||_2] \leq 2\sqrt{d}$, so that

$$\sqrt{\log \widetilde{N}(\delta)} \;\leq\; \frac{6\rho(D)}{\delta}\, \sqrt{d},$$

thereby establishing the bound (39).

C.2 Establishing the Tail Bound (38)

Recalling the definition of the operator $\mathfrak{X}_n{}'$, we have

$$\begin{aligned}F_\Gamma(\mathfrak{X}_n{}') &= \frac{1}{\sqrt{n}} \Big\{ \sum_{i=1}^n \langle\langle \widetilde{X}^{(i)}, \Gamma \rangle\rangle^2 \Big\}^{1/2} \\ &= \frac{1}{\sqrt{n}} \sup_{\|u\|_2 = 1} \sum_{i=1}^n u_i \langle\langle \widetilde{X}^{(i)}, \Gamma \rangle\rangle \\ &= \frac{1}{\sqrt{n}} \sup_{\|u\|_2 = 1} \sum_{i=1}^n u_i Y_i\end{aligned}$$

where we have defined the random variables $Y_i := \langle\langle \widetilde{X}^{(i)}, \Gamma \rangle\rangle$. Note that each Y_i is zero-mean, and bounded by $2L$ since

$$\begin{aligned}|Y_i| &= |\langle\langle \widetilde{X}^{(i)}, \Gamma \rangle\rangle| \\ &\leq \big(\sum_{a,b} |\widetilde{X}^{(i)}|_{ab}\big)\, \|\Gamma\|_\infty \;\leq\; 2L.\end{aligned}$$

where we have used the facts that $\|\Gamma\|_\infty \leq 2/d$, and $\sum_{a,b} |\widetilde{X}^{(i)}|_{ab} \leq L\, d$, by definition of the matrices $\widetilde{X}^{(i)}$.

Therefore, applying Corollary 4.8 from Ledoux (2001), we conclude that

$$\mathbb{P}\big[|F_\Gamma(\mathfrak{X}_n{}') - \mathbb{E}[F_\Gamma(\mathfrak{X}_n{}')]| \geq \delta + \frac{32L}{\sqrt{n}}\big] \leq 4\exp\big(-\frac{n\delta^2}{4L^2}\big).$$

The same corollary implies that

$$\Big|\sqrt{\mathbb{E}[F_\Gamma^2(\mathfrak{X}_n{}')]} - \mathbb{E}[F_\Gamma(\mathfrak{X}_n{}')]\Big| \leq \frac{16L}{\sqrt{n}}.$$

Since $\mathbb{E}[F_\Gamma^2(\mathfrak{X}_n{}')] = |||\Gamma|||_F^2$, the tail bound (38) follows.

Appendix D. Proof of Lemma 5

From the proof of Lemma 4, recall the definition $F_\Gamma(\mathfrak{X}_n{}') = \frac{1}{\sqrt{n}}\|\mathfrak{X}_n{}'(\Gamma)\|_2$ where $\mathfrak{X}_n{}'$ is the random sampling operator defined by the n matrices $(\widetilde{X}^{(1)}, \ldots, \widetilde{X}^{(n)})$. Using this notation, our goal is to bound the function

$$G(\mathfrak{X}_n{}') := \sup_{\Delta \in \mathfrak{D}(\delta, R)} F_\Delta(\mathfrak{X}_n{}'),$$

where we recall that $\mathfrak{D}(\delta, R) := \big\{\Delta \in \mathbb{R}^{d_r \times d_c} \mid |||\Delta|||_F \leq \delta,\ |||\Delta|||_1 \leq 2\rho(D),\ \|\Delta\|_\infty \leq \frac{2}{d}\big\}$. Ultimately, we will set $\delta = \frac{D}{8}$, but we use δ until the end of the proof for compactness in notation.

Our approach is a standard one: first show concentration of G around its expectation $\mathbb{E}[G(\mathfrak{X}_n{}')]$, and then upper bound the expectation. We show concentration via a bounded difference inequality; since G is a symmetric function of its arguments, it suffices to establish the bounded difference property with respect to the first co-ordinate. In order to do so, consider a second operator $\widetilde{\mathfrak{X}_n{}'}$ defined by the matrices $(Z^{(1)}, \widetilde{X}^{(2)}, \ldots, \widetilde{X}^{(n)})$, differing from $\mathfrak{X}_n{}'$ only in the first matrix. Given the pair $(\mathfrak{X}_n{}', \widetilde{\mathfrak{X}_n{}'})$, we have

$$\begin{aligned}
G(\mathfrak{X}_n{}') - G(\widetilde{\mathfrak{X}_n{}'}) &= \sup_{\Delta \in \mathfrak{D}(\delta, R)} F_\Delta(\mathfrak{X}_n{}') - \sup_{\Theta \in \mathfrak{D}(\delta, R)} F_\Theta(\widetilde{\mathfrak{X}_n{}'}) \\
&\leq \sup_{\Delta \in \mathfrak{D}(\delta, R)} \big[F_\Delta(\mathfrak{X}_n{}') - F_\Delta(\widetilde{\mathfrak{X}_n{}'})\big] \\
&\leq \sup_{\Delta \in \mathfrak{D}(\delta, R)} \frac{1}{\sqrt{n}}\|\mathfrak{X}_n{}'(\Delta) - \widetilde{\mathfrak{X}_n{}'}(\Delta)\|_2 \\
&= \sup_{\Delta \in \mathfrak{D}(\delta, R)} \frac{1}{\sqrt{n}}\big|\langle\langle \widetilde{X}^{(1)} - Z^{(1)},\, \Delta\rangle\rangle\big|.
\end{aligned}$$

For any fixed $\Delta \in \mathfrak{D}(\delta, R)$, we have

$$\big|\langle\langle \widetilde{X}^{(1)} - Z^{(1)},\, \Delta\rangle\rangle\big| \leq 2Ld\,\|\Delta\|_\infty \ \leq\ 4L,$$

where we have used the fact that the matrix $\widetilde{X}^{(1)} - Z^{(1)}$ is non-zero in at most two entries with values upper bounded by $2Ld$. Combining the pieces yields $G(\mathfrak{X}_n{}') - G(\widetilde{\mathfrak{X}_n{}'}) \leq \frac{4L}{\sqrt{n}}$. Since the same argument can be applied with the roles of $\mathfrak{X}_n{}'$ and $\widetilde{\mathfrak{X}_n{}'}$ interchanged, we conclude that $|G(\mathfrak{X}_n{}') - G(\widetilde{\mathfrak{X}_n{}'})| \leq \frac{4L}{\sqrt{n}}$. Therefore, by the bounded differences variant of the Azuma-Hoeffding inequality (Ledoux, 2001), we have

$$\mathbb{P}\big[|G(\mathfrak{X}_n{}') - \mathbb{E}[G(\mathfrak{X}_n{}')]| \geq t\big] \leq 2\exp\big(-\frac{nt^2}{32L^2}\big). \tag{40}$$

Next we bound the expectation. First applying Jensen's inequality, we have

$$\begin{aligned}(\mathbb{E}[G(\mathfrak{X}_n')])^2 &\leq \mathbb{E}[G^2(\mathfrak{X}_n')] \\ &= \mathbb{E}\big[\sup_{\Delta\in\mathfrak{D}(\delta,R)} \frac{1}{n}\sum_{i=1}^n \langle\langle \widetilde{X}^{(i)}, \Delta\rangle\rangle^2\big] \\ &= \mathbb{E}\Big[\sup_{\Delta\in\mathfrak{D}(\delta,R)} \Big\{\frac{1}{n}\sum_{i=1}^n \big[\langle\langle \widetilde{X}^{(i)}, \Delta\rangle\rangle^2 - \mathbb{E}[\langle\langle \widetilde{X}^{(i)}, \Delta\rangle\rangle^2]\big] + |||\Delta|||_F^2\Big\}\Big] \\ &\leq \mathbb{E}\Big[\sup_{\Delta\in\mathfrak{D}(\delta,R)} \Big\{\frac{1}{n}\sum_{i=1}^n \big[\langle\langle \widetilde{X}^{(i)}, \Delta\rangle\rangle^2 - \mathbb{E}[\langle\langle \widetilde{X}^{(i)}, \Delta\rangle\rangle^2]\big]\Big\}\Big] + \delta^2,\end{aligned}$$

where we have used the fact that $\mathbb{E}[\langle\langle \widetilde{X}^{(i)}, \Delta\rangle\rangle^2 = |||\Delta|||_F^2 \leq \delta^2$. Now a standard symmetrization argument (Ledoux and Talagrand, 1991) yields

$$\mathbb{E}_{\mathfrak{X}_n'}[G^2(\mathfrak{X}_n')] \leq 2\,\mathbb{E}_{\mathfrak{X}_n',\varepsilon}\big[\sup_{\Delta\in\mathfrak{D}(\delta,R)} \frac{1}{n}\sum_{i=1}^n \varepsilon_i \langle\langle \widetilde{X}^{(i)}, \Delta\rangle\rangle^2\big] + \delta^2,$$

where $\{\varepsilon_i\}_{i=1}^n$ is an i.i.d. Rademacher sequence. Since $|\langle\langle \widetilde{X}^{(i)}, \Delta\rangle\rangle| \leq 2L$ for all i, the Ledoux-Talagrand contraction inequality (p. 112, Ledoux and Talagrand, 1991) implies that

$$\mathbb{E}[G^2(\mathfrak{X}_n')] \leq 16L\,\mathbb{E}\big[\sup_{\Delta\in\mathfrak{D}(\delta,R)} \big\{\frac{1}{n}\sum_{i=1}^n \varepsilon_i \langle\langle \widetilde{X}^{(i)}, \Delta\rangle\rangle\big\}\big] + \delta^2.$$

By the duality between operator and nuclear norms, we have

$$\big|\frac{1}{n}\sum_{i=1}^n \varepsilon_i \langle\langle \widetilde{X}^{(i)}, \Delta\rangle\rangle\big| \leq |||\frac{1}{n}\sum_{i=1}^n \varepsilon_i \widetilde{X}^{(i)}|||_2\; |||\Delta|||_1,$$

and hence, since $|||\Delta|||_1 \leq \rho(D)$ for all $\Delta \in \mathfrak{D}(\delta, R)$, we have

$$\mathbb{E}[G^2(\mathfrak{X}_n')] \leq 16L\rho(D)\,\mathbb{E}\big[|||\frac{1}{n}\sum_{i=1}^n \varepsilon_i \widetilde{X}^{(i)}|||_2\big] + \delta^2. \tag{41}$$

It remains to bound the operator norm $\mathbb{E}\big[|||\frac{1}{n}\sum_{i=1}^n \varepsilon_i \widetilde{X}^{(i)}|||_2\big]$. The following lemma, proved in Appendix E, provides a suitable upper bound:

Lemma 6 *We have the upper bound*

$$\mathbb{E}\big[|||\frac{1}{n}\sum_{i=1}^n \varepsilon_i \widetilde{X}^{(i)}|||_2\big] \leq 10\max\Big\{\sqrt{\frac{d\log d}{n}},\ \frac{Ld\log d}{n}\Big\}.$$

Thus, as long as $n = \Omega(Ld\log d)$, combined with the earlier bound (41), we conclude that

$$\mathbb{E}[G(\mathfrak{X}_n')] \leq \sqrt{\mathbb{E}[G^2(\mathfrak{X}_n')]} \leq \big[160L\rho(D)\sqrt{\frac{d\log d}{n}} + \delta^2\big]^{1/2},$$

where we have used the fact that $L \geq 1$. By definition of $\rho(D)$, we have

$$160L^2\rho(D)\sqrt{\frac{d\log d}{n}} = \frac{160}{c_0}D^2 \leq \big(\frac{5D}{16}\big)^2,$$

where the final inequality can be guaranteed by choosing c_0 sufficiently large.

Consequently, recalling our choice $\delta = D/8$ and using the inequality $\sqrt{a^2+b^2} \leq |a|+|b|$, we obtain

$$\mathbb{E}[G(\mathfrak{X}_n')] \leq \frac{5}{16}D + \frac{D}{8} = \frac{7}{16}D.$$

Finally, setting $t = \frac{D}{16}$ in the concentration bound (40) yields

$$G(\mathfrak{X}_n') \leq \frac{D}{16} + \frac{7}{16}D = \frac{D}{2}$$

with probability at least $1 - 2\exp\big(-c'\frac{nD^2}{L^2}\big)$ as claimed.

Appendix E. Proof of Lemma 6

We prove this lemma by applying a form of Ahlswede-Winter matrix bound (2002), as stated in Appendix F, to the matrix $Y^{(i)} := \varepsilon_i \widetilde{X}^{(i)}$. We first compute the quantities involved in Lemma 7. Note that $Y^{(i)}$ is a zero-mean random matrix, and satisfies the bound

$$|||Y^{(i)}|||_2 = d\frac{1}{\sqrt{R_{j(i)}}\sqrt{C_{k(i)}}}|||\varepsilon_i\, e_{j(i)}\, e_{k(i)}^T|||_2 \;\leq\; Ld.$$

Let us now compute the quantities σ_i in Lemma 7. We have

$$\mathbb{E}\big[(Y^{(i)^T})Y^{(i)}\big] = \mathbb{E}\left[\frac{d^2}{R_{j(i)}C_{k(i)}} e_{k(i)}e_{k(i)}^T\right] \;=\; dI_{d\times d}$$

and similarly, $\mathbb{E}\big[Y^{(i)}\,(Y^{(i)})^T\big] = dI_{d\times d}$, so that

$$\sigma_i^2 = \max\Big\{|||\mathbb{E}\big[Y^{(i)}\,(Y^{(i)})^T\big]|||_2,\; |||\mathbb{E}\big[(Y^{(i)})^T\,Y^{(i)}\big]|||_2\Big\} \;=\; d.$$

Thus, applying Lemma 7 yields the tail bound

$$\mathbb{P}\big[|||\sum_{i=1}^n \varepsilon_i\widetilde{X}^{(i)}|||_2 \geq t\big] \leq 2d\,\max\big\{\exp(-\frac{t^2}{4nd}),\,\exp(-\frac{t}{2Ld})\big\}.$$

Setting $t = n\delta$, we obtain

$$\mathbb{P}\big[|||\frac{1}{n}\sum_{i=1}^n \varepsilon_i\widetilde{X}^{(i)}|||_2 \geq \delta\big] \leq 2d\,\max\big\{\exp(-\frac{n\delta^2}{4d}),\exp(-\frac{n\delta}{2Ld})\big\}.$$

Recall that for any non-negative random variable T, we have $\mathbb{E}[T] = \int_0^\infty \mathbb{P}[T \geq s]ds$. Applying this fact to $T := |||\frac{1}{n}\sum_{i=1}^n \varepsilon_i\widetilde{X}^{(i)}|||_2$ and integrating the tail bound, we obtain

$$\mathbb{E}\big[|||\frac{1}{n}\sum_{i=1}^n \varepsilon_i\widetilde{X}^{(i)}|||_2\big] \leq 10\,\max\big\{\sqrt{\frac{d\log d}{n}},\;\frac{Ld\log d}{n}\big\}.$$

Appendix F. Ahlswede-Winter Matrix Bound

Here we state a Bernstein version of the Ahlswede-Winter (2002) tail bound for the operator norm of a sum of random matrices. The version here is a slight weakening (but sufficient for our purposes) of a result due to Recht (2011); we also refer the reader to the chapter of Vershynin (2012), and the strengthened results provided by Tropp (2010).

Let $Y^{(i)}$ be independent $d_r \times d_c$ zero-mean random matrices such that $|||Y^{(i)}|||_2 \leq M$, and define $\sigma_i^2 := \max\{|||\mathbb{E}[(Y^{(i)})^T Y^{(i)}]|||_2, \quad |||\mathbb{E}[Y^{(i)}(Y^{(i)})^T]|||_2\}$ as well as $\sigma^2 := \sum_{i=1}^n \sigma_i^2$.

Lemma 7 *We have*

$$\mathbb{P}\big[|||\sum_{i=1}^n Y^{(i)}|||_2 \geq t\big] \leq (d_r \times d_c) \max\big\{\exp(-t^2/(4\sigma^2), \exp(-\frac{t}{2M})\big\}.$$

As noted by Vershynin (2009), the same bound also holds under the assumption that each $Y^{(i)}$ is sub-exponential with parameter $M = \|Y^{(i)}\|_{\psi_1}$. Here we are using the Orlicz norm

$$\|Z\|_{\psi_1} := \inf\{t > 0 \mid \mathbb{E}[\psi(|Z|/t)] < \infty\},$$

defined by the function $\psi_1(x) = \exp(x) - 1$, as is appropriate for sub-exponential variables (e.g., see Ledoux and Talagrand, 1991).

References

A. Agarwal, S. Negahban, and M. J. Wainwright. Noisy matrix decomposition via convex relaxation: Optimal rates in high dimensions. *Annals of Statistics*, 2011. To appear.

R. Ahlswede and A. Winter. Strong converse for identification via quantum channels. *IEEE Transactions on Information Theory*, 48(3):569–579, March 2002.

F. Bach. Consistency of trace norm minimization. *Journal of Machine Learning Research*, 9: 1019–1048, June 2008.

D.P. Bertsekas. *Nonlinear programming*. Athena Scientific, Belmont, MA, 1995.

E. Candès and Y. Plan. Matrix completion with noise. *Proceedings of the IEEE*, 98(6):925–936, 2010.

E. J. Candès and B. Recht. Exact matrix completion via convex optimization. *Found. Comput. Math.*, 9(6):717–772, 2009.

E. J. Candes and T. Tao. The power of convex relaxation: Near-optimal matrix completion. *IEEE Transactions on Information Theory*, 56(5):2053–2080, 2010.

K. R. Davidson and S. J. Szarek. Local operator theory, random matrices, and Banach spaces. In *Handbook of Banach Spaces*, volume 1, pages 317–336. Elsevier, Amsterdam, NL, 2001.

M. Deza and M. Laurent. *Geometry of Cuts and Metric Embeddings*. Springer-Verlag, New York, 1997.

M. Fazel. *Matrix Rank Minimization with Applications*. PhD thesis, Stanford, 2002. Available online: http://faculty.washington.edu/mfazel/thesis-final.pdf.

D. Gross. Recovering low-rank matrices from few coefficients in any basis. *IEEE Transactions on Information Theory*, 57(3):1548–1566, March 2011.

R. A. Horn and C. R. Johnson. *Matrix Analysis*. Cambridge University Press, Cambridge, 1985.

R. H. Keshavan, A. Montanari, and S. Oh. Matrix completion from few entries. *IEEE Transactions on Information Theory*, 56(6):2980–2998, June 2010a.

R. H. Keshavan, A. Montanari, and S. Oh. Matrix completion from noisy entries. *Journal of Machine Learning Research*, 11:2057–2078, July 2010b.

M. Laurent. Matrix completion problems. In *The Encyclopedia of Optimization*, pages 221—229. Kluwer Academic, 2001.

M. Ledoux. *The Concentration of Measure Phenomenon*. Mathematical Surveys and Monographs. American Mathematical Society, Providence, RI, 2001.

M. Ledoux and M. Talagrand. *Probability in Banach Spaces: Isoperimetry and Processes*. Springer-Verlag, New York, NY, 1991.

Z. Lin, A. Ganesh, J. Wright, L. Wu, M. Chen, and Y. Ma. Fast convex optimization algorithms for exact recovery of a corrupted low-rank matrix. Technical Report UILU-ENG-09-2214, Univ. Illinois, Urbana-Champaign, July 2009.

R. Mazumber, T. Hastie, and R. Tibshirani. Spectral regularization algorithms for learning large incomplete matrices. *Journal of Machine Learning Research*, 11:2287–2322, August 2010.

S. Negahban and M. J. Wainwright. Estimation of (near) low-rank matrices with noise and high-dimensional scaling. *Annals of Statistics*, 39(2):1069–1097, 2011.

S. Negahban, P. Ravikumar, M. J. Wainwright, and B. Yu. A unified framework for high-dimensional analysis of M-estimators with decomposable regularizers. In *Neural Information Processing Systems (NIPS)*, Vancouver, Canada, December 2009. To appear in Statistical Science.

Y. Nesterov. Gradient methods for minimizing composite objective function. Technical Report 2007/76, CORE, Universit'e catholique de Louvain, 2007.

G. Pisier. *The Volume of Convex Bodies and Banach Space Geometry*, volume 94 of *Cambridge Tracts in Mathematics*. Cambridge University Press, Cambridge, UK, 1989.

B. Recht. A simpler approach to matrix completion. *Journal of Machine Learning Research*, 12: 3413–3430, 2011.

B. Recht, M. Fazel, and P. Parrilo. Guaranteed minimum-rank solutions of linear matrix equations via nuclear norm minimization. *SIAM Review*, 52(3):471–501, 2010.

A. Rohde and A. Tsybakov. Estimation of high-dimensional low-rank matrices. *Annals of Statistics*, 39(2):887–930, 2011.

R. Salakhutdinov and N. Srebro. Collaborative filtering in a non-uniform world: Learning with the weighted trace norm. Technical Report abs/1002.2780v1, Toyota Institute of Technology, 2010.

N. Srebro. *Learning with Matrix Factorizations*. PhD thesis, MIT, 2004. Available online: http://ttic.uchicago.edu/ nati/Publications/thesis.pdf.

N. Srebro, J. Rennie, and T. S. Jaakkola. Maximum-margin matrix factorization. In *Neural Information Processing Systems (NIPS)*, Vancouver, Canada, December 2004.

N. Srebro, N. Alon, and T. S. Jaakkola. Generalization error bounds for collaborative prediction with low-rank matrices. In *Neural Information Processing Systems (NIPS)*, Vancouver, Canada, December 2005.

S. J. Szarek. The finite dimensional basis problem with an appendix on nets of grassmann manifolds. *Acta Mathematica*, 151:153–179, 1983.

J. Tropp. User-friendly tail bounds for matrix martingales. Technical report, Caltech, April 2010.

R. Vershynin. A note on sums of independent random matrices after Ahlswede-Winter. Technical report, Univ. Michigan, December 2009.

R. Vershynin. Introduction to the non-asymptotic analysis of random matrices. *Compressed Sensing: Theory and Applications*, 2012. Available at `http://www-personal.umich.edu/~romanv/papers/non-asymptotic-rmt-plain.pdf`.

Y. Yang and A. Barron. Information-theoretic determination of minimax rates of convergence. *Annals of Statistics*, 27(5):1564–1599, 1999.

B. Yu. Assouad, Fano and Le Cam. In *Festschrift for Lucien Le Cam*, pages 423–435. Springer-Verlag, Berlin, 1997.

Journal of Machine Learning Research 13 (2012) 1699-1703 Submitted 11/10; Revised 10/11; Published 5/12

glm-ie: Generalised Linear Models Inference & Estimation Toolbox

Hannes Nickisch HANNES@NICKISCH.ORG
Max Planck Institute for Biological Cybernetics
Spemannstraße 38
72076 Tübingen, Germany

Editor: Mikio Braun

Abstract

The glm-ie toolbox contains functionality for estimation and inference in generalised linear models over continuous-valued variables. Besides a variety of penalised least squares solvers for estimation, it offers inference based on (convex) variational bounds, on expectation propagation and on factorial mean field. Scalable and efficient inference in fully-connected undirected graphical models or Markov random fields with Gaussian and non-Gaussian potentials is achieved by casting all the computations as matrix vector multiplications. We provide a wide choice of penalty functions for estimation, potential functions for inference and matrix classes with lazy evaluation for convenient modelling. We designed the glm-ie package to be simple, generic and easily expansible. Most of the code is written in Matlab including some MEX files to be fully compatible to both Matlab 7.x and GNU Octave 3.3.x. Large scale probabilistic classification as well as sparse linear modelling can be performed in a common algorithmical framework by the glm-ie toolkit.

Keywords: sparse linear models, generalised linear models, Bayesian inference, approximate inference, probabilistic regression and classification, penalised least squares estimation, lazy evaluation matrix class

1. Introduction

Generalised Linear Models (GLMs) are a widely used class of probabilistic graphical models over continuous variables allowing a unified treatment of linear, logistic and Poisson regression and applications range from simple binary pattern classification over continuous regression to image reconstruction. GLMs combine the computational and analytical simplicity of linear functions with the expressivity of pointwise nonlinear link functions.

Estimation of the unknown parameters by maximum likelihood and penalised variants thereof can be done by iteratively reweighted least squares (IRLS). Penalised least squares (PLS) corresponds to maximum a posteriori estimation (MAP) in a Bayesian model.

(Approximate) Bayesian inference as opposed to MAP places the parameter estimate at the centre of mass rather than at the mode of the posterior distribution.

- We support Expectation Propagation (EP) or TAP (Minka, 2001; Opper and Winther, 2001) using parallel moment matching (van Gerven et al., 2010),
- Variational Bounding (VB) (Seeger and Nickisch, 2011) based on decoupling (Wipf and Nagarajan, 2008) and (convex) (Nickisch and Seeger, 2009) optimisation, and
- Mean field (MF) (Miskin and MacKay, 2000) factorial inference.

While EP yields very accurate approximations for small models, VB allows for efficient computations in large-scale models by a sequence of variance-smoothed PLS problems, which makes experimental design for imaging applications (Seeger et al., 2010) possible.

Related codes are the GPML library and the Infer.NET framework both of which do not offer scalable matrix vector multiplication (MVM) based approximate inference.

2. Implementation and Model Class

The `glm-ie` toolbox can be obtained from `http://mloss.org/software/view/269/` and from `http://hannes.nickisch.org/code/glm-ie/` under the FreeBSD license. Based on simple interfaces for *potential*, *penalty*, and *estimation* functions as well as *inference* methods and *matrix* classes, we offer full compatibility to Matlab 7.x[1] and GNU Octave 3.3.x[2].

We provide modular, extensible and tested code. The algorithms rely on PLS estimations based on MVMs in turn. Our documentation comes in two parts: (i) a hypertext document[3] `doc/index.html` with detailed examples and (ii) a technical documentation[4] `doc/manual.pdf` explaining the interfaces to allow inclusion of new functionality.

The `glm-ie` toolbox deals with inference and estimation in GLMs of unknown hidden parameters $\mathbf{u} \in \mathbb{R}^n$, Gaussian observations $\mathbf{y} \in \mathbb{R}^m$ and non-Gaussian potentials $\mathcal{T}_j(s_j)$

$$\mathbf{y} = \mathbf{X}\mathbf{u} + \varepsilon,\ \varepsilon \sim \mathcal{N}(\mathbf{0}, \sigma^2\mathbf{I}), \quad \mathbf{s} = \mathbf{B}\mathbf{u} \in \mathbb{R}^q,$$

leading to a posterior of the form

$$\mathbb{P}(\mathbf{u}|\mathcal{D}) = \mathbb{P}(\mathbf{u}|\mathbf{X},\mathbf{y},\sigma) = \frac{1}{Z}\mathcal{N}(\mathbf{y}|\mathbf{X}\mathbf{u},\sigma^2\mathbf{I})\prod_{j=1}^{q}\mathcal{T}_j(s_j), \quad Z = \int \mathcal{N}(\mathbf{y}|\mathbf{X}\mathbf{u},\sigma^2\mathbf{I})\prod_{j=1}^{q}\mathcal{T}_j(s_j)\mathrm{d}\mathbf{u}. \tag{1}$$

A MAP estimate $\hat{\mathbf{u}}_{\mathrm{MAP}} \in \mathbb{R}^n$ is the parameter value with highest posterior density; finding $\hat{\mathbf{u}}_{\mathrm{MAP}}$ is equivalent to solving the PLS problem

$$\hat{\mathbf{u}}_{\mathrm{MAP}} = \arg\max_{\mathbf{u}} \mathbb{P}(\mathbf{u}|\mathcal{D}) = \arg\min_{\mathbf{u}} \|\mathbf{X}\mathbf{u}-\mathbf{y}\|^2 + 2\lambda\cdot\rho(\mathbf{s}),\ \mathbf{s} = \mathbf{B}\mathbf{u},\ \lambda \in \mathbb{R}_+ \tag{2}$$

with *penalty function* $\rho(\mathbf{s}) = -\sum_{j=1}^{q}\ln\mathcal{T}_j(s_j)$ derived from the *potential function* $\mathcal{T}(\mathbf{s})$ and weight $\lambda = \sigma^2$. The normalisation constant Z is called the model evidence or equivalently the marginal likelihood and can be used to compare models and adjust free parameters such as σ.

Our approximate inference algorithms replace the non-Gaussian potentials $\mathcal{T}_j(s_j)$ by Gaussians $\mathcal{N}(s_j|\beta_j\gamma_j,\gamma_j) \propto \exp(-s_j^2/(2\gamma_j)+\beta_j s_j)$ resulting in an overall Gaussian approximation $\mathbb{P}(\mathbf{u}|\mathcal{D}) \approx \mathbb{Q}(\mathbf{u}) = \mathcal{N}(\mathbf{u}|\mathbf{m},\mathbf{V})$, where $\mathbf{V}^{-1} = \mathbf{A} = \mathbf{X}^\top\mathbf{X}/\sigma^2 + \mathbf{B}^\top\boldsymbol{\Gamma}^{-1}\mathbf{B}$ and $\mathbf{m} = \mathbf{A}^{-1}(\mathbf{X}^\top\mathbf{y}/\sigma^2 + \mathbf{B}^\top\boldsymbol{\beta})$ correspond to the mean and (co)variance and $\boldsymbol{\Gamma} := \mathrm{dg}(\gamma_1,..,\gamma_q)$.

Overall, a GLM can be specified by three kinds of objects: (i) *potentials* $\mathcal{T}(s)$ and *penalties* $\rho(s)$, (ii) *matrices* $\mathbf{X}$, $\mathbf{B}$ and (iii) *PLS algorithms*. Together with the responses $\mathbf{y}$, scalar parameters and optimisation options, these three constituents serve as inputs to the double loop inference engine `dli` computing approximations to $\ln Z$, $\mathbf{m}$ and $\mathbf{V}$.

1. Matlab is available from MathWorks, `http://www.mathworks.com/`.
2. Octave is available from the Free Software Foundation, `http://www.gnu.org/software/octave/`.
3. Documentation can be found at `http://people.kyb.tuebingen.mpg.de/hn/glm-ie/doc/index.html`.
4. Technical docs are available at `http://people.kyb.tuebingen.mpg.de/hn/glm-ie/manual.pdf`.

2.1 Potential and Penalty Functions

Several *non-Gaussian potentials* $\mathcal{T}(s)$[5] can be used to shape the posterior distribution of Equation (1). In addition to the Gaussian potential `potGauss`, we provide several sparse potentials (exponential power `potExpPow`, Laplace `potLaplace`, Sech-squared[6] `potSech2`, Student's t `potT`) and the logistic potential `potLogistic` for binary classification.

In MAP estimation, we most naturally use the *penalty function* $\rho(s) = -\ln \mathcal{T}(s)$ in Equation (2) which includes penalty functions like p-norms `penAbs`, `penQuad`, `penPow`. Approximate inference by variational bounding requires *penalty functions* derived from a *potential function*, for example, `penVB` or `penVBNorm`.

2.2 General Matrix Class and Implementations

To facilitate the specification and composition of system matrices $\mathbf{X}$, $\mathbf{B}$ (and their respective transposes) in a GLM, the `glm-ie` toolbox contains a specialised matrix class `mat`. As MVMs form the most important computations in both estimation and inference, the class `mat` provides addition, transposition, scaling, composition, and concatenation. Thus, expressions like `A+B'`, `A*B`, `A*x`, `a*A`, `[A,B]`, `[A;B]`, `A(:,1)`, `kron(A,B)`, `repmat(A,[2,3])` are possible even though `A` or `B` are of type `mat` and have a size that would not fit into memory if stored as a dense array. Combinations are also possible. We provide several matrix classes that are derived from `mat` and implement their own MVM. Besides 2d convolution `matConv2`, diagonal `matDiag`, finite difference `matFD2` and (quadrature mirror) wavelet matrices `matWav`, we offer three kinds of Fourier matrices `matFFT2line`, `matFFTNmask` and `matFFT2nu` allowing for nonuniform spacing. Computations only take place through MVMs; the other operations (addition, composition etc.) are pure bookkeeping.

2.3 Penalised Least Squares Solvers

The `glm-ie` toolbox contains several solvers for the PLS estimation problem of Equation (2):

- `plsCG`: Conjugate Gradients (CG) using a standalone solver (Rasmussen, 2006),
- `plsCGBT`: CG with an Armijo backtracking rule (Lustig et al., 2007),
- `plsTN`: Truncated Newton or IRLS (Seeger et al., 2009),
- `plsLBFGS`: uses a wrapper for the famous LBFGSB code (L-BFGS-B, 1997),
- `plsBB`: first order two-point step size rule (Barzilai and Borwein, 1988), and
- `plsSB`: Bregman Splitting (Goldstein and Osher, 2009).

The solvers can be used for a standalone estimation task or—together with the `penVB` penalty function—as the inner loop of the double loop variational inference algorithm.

5. We use an additional scale parameter τ, that is, the rescaled potential $\mathcal{T}(\tau s)$.
6. The sech-square distribution is another name for the logistic distribution. We use sech-square to avoid a name clash with the logistic classification potential.

3. Example and Code

To illustrate the modular structure of the `glm-ie` toolbox, we provide a simple code example[7]. We use a sparse linear model to describe Fourier measurements $\mathbf{y} = \mathbf{X}\mathbf{u} + \boldsymbol{\varepsilon} \in \mathbb{C}^m$ of an unknown pixel image $\mathbf{u} \in \mathbb{R}^n$ where the design matrix $\mathbf{X}$ contains a subset of the rows of the Fourier matrix $\mathbf{F}$, that is, $\mathbf{X} = \mathbf{M}\mathbf{F}$ as specified by a measurement masking matrix $\mathbf{M} \in \{0,1\}^{m \times n}$. As a prior, we employ the knowledge that the filter responses $\mathbf{s} = \mathbf{B}\mathbf{u}$ of natural images with zero mean filters $\mathbf{b}_j$, $j{=}1..q$ follow a sparse distribution imitated by the Laplace potential $\mathcal{T}_j(s_j) = e^{-\tau|s_j|}$. More specifically, the filter matrix $\mathbf{B}$ contains multiscale derivatives; it is a concatenation of finite differences in both image directions and wavelet coefficients. This model allows to reconstruct images from undersampled magnetic resonance imaging scanner measurements, where $m < n$.

These steps are illustrated below, and following the code, we discuss in some detail, the meaning and role of each line, and mention in passing some of the alternative possibilities. Note that, full specification and prediction can be done in as little as nine lines of code:

```
1 [y,mask] = read_data; su = size(mask);          % load measurements y and mask
2 X    = matFFTNmask(mask);                    % construct a partial Fourier matrix
3 s2   = 1e-5;                               % define the observation noise variance
4 B    = [matWav(su); matFD2(su)];  % conc. wavelet and finite difference matrix
5 pen  = @(s) penAbs(s);                 % define l1-norm penalty function (LASSO)
6 [u,phi]  = plsLBFGS(u0,X,y,B,opt,s2,pen);             % perform PLS estimation
7 pot  = @potLaplace;       % define a Laplace potential (corresponds to l1-norm)
8 tau  = 15;                               % declare width of the Laplace potential
9 [m,ga,be,z,zu,nlZ] = dli(X,y,s2,B,pot,tau,opts);   % double loop VB inference
```

Data $\mathbf{y}$ is loaded in **line 1**. The observation noise variance σ^2 and the design matrix $\mathbf{X}$ as declared in **lines 2-3** form the Gaussian part. The filter matrix $\mathbf{B}$ for the non-Gaussian part is constructed in **line 4** as the concatenation of two matrices. More involved compositions such as `[W; a*D]` are possible. In the next two lines, we perform estimation by optimising equation (2) using the `plsLBFGS` solver in **line 6** for the penalty function $\rho(\mathbf{s}) = \sum_j |s_j|$ as declared in **line 5**. Variational inference requires a potential $\mathcal{T}(\tau\mathbf{s})$; here $\mathcal{T}(\tau s) = \exp(-\tau|s|)$, with scale τ, is defined in **lines 7-8**. The engine for double loop inference `dli` is finally called in **line 9** yielding the posterior mean estimate $\mathbf{m}$, the variational parameters $\boldsymbol{\gamma}$ and $\boldsymbol{\beta}$, the marginal variances $\mathbf{z} = \mathrm{var}(\mathbf{B}\mathbf{u})$, $\mathbf{z}_\mathbf{u} = \mathrm{var}(\mathbf{u})$ and the negative log evidence $-\ln Z \equiv$ `nlZ` of the model.

Acknowledgments

Thanks go to George Papandreou for contributing his marginal variance estimator (Papandreou and Yuille, 2011), to Matthias Seeger for helpful suggestions, Michael Hirsch for testing and contributing the convolution code and Wolf Blecher for testing.

References

Jonathan Barzilai and Jonathan M. Borwein. Two-point step size gradient methods. *IMA Journal of Numerical Analysis*, 8:141–148, 1988.

7. The example is a shortened version of one the detailed application examples, which are part of the documentation of the `glm-ie` package.

Tom Goldstein and Stanley Osher. The split Bregman method for l1 regularized problems. *SIAM Journal on Imaging Sciences*, 2(2):323–343, 2009. URL ftp://ftp.math.ucla.edu/pub/camreport/cam08-29.pdf.

Michael Lustig, David L. Donoho, and John M. Pauly. Sparse MRI: The application of compressed sensing for rapid MR imaging. *Magnetic Resonance in Medicine*, 58(6):1182–1195, 2007.

Tom Minka. Expectation propagation for approximate Bayesian inference. In *UAI*, 2001.

James Miskin and David J.C. MacKay. Ensemble learning for blind image separation and deconvolution. In *Advances in Independent Component Analysis*, 2000.

Hannes Nickisch and Matthias Seeger. Convex variational Bayesian inference for large scale generalized linear models. In *ICML*, 2009.

Manfred Opper and Ole Winther. Adaptive and selfaveraging Thouless-Anderson-Palmer mean field theory for probabilistic modeling. *Physical Review E*, 64:056131, 2001.

George Papandreou and Alan Yuille. Efficient variational inference in large-scale Bayesian compressed sensing. In *Proc. IEEE Workshop on Information Theory in Computer Vision and Pattern Recognition (in conjunction with ICCV)*, 2011.

Carl E. Rasmussen. minimize.m, September 2006. URL http://www.kyb.tuebingen.mpg.de/bs/people/carl/code/minimize/.

Matthias W. Seeger and Hannes Nickisch. Large scale variational inference and experimental design for sparse generalized linear models. *SIAM Journal on Imaging Sciences*, 4(1):166–199, 2011.

Matthias W. Seeger, Hannes Nickisch, Rolf Pohmann, and Bernhard Schölkopf. Bayesian experimental design of magnetic resonance imaging sequences. In *NIPS*, 2009.

Matthias W. Seeger, Hannes Nickisch, Rolf Pohmann, and Bernhard Schölkopf. Optimization of k-space trajectories for compressed sensing by bayesian experimental design. *Magnetic Resonance in Medicine*, 63(1):116–126, 2010.

L-BFGS-B. by Ciyou Zhu, Richard Byrd and Jorge Nocedal., 1997. URL http://www.eecs.northwestern.edu/~nocedal/lbfgsb.html.

Marcel van Gerven, Botond Cseke, Floris de Lange, and Tom Heskes. Efficient Bayesian multivariate fMRI analysis using a sparsifying spatio-temporal prior. *Neuroimage*, 50:150–161, 2010.

David Wipf and Srikantan Nagarajan. A new view of automatic relevance determination. In *NIPS*, 2008.

Journal of Machine Learning Research 13 (2012) 1705-1744 Submitted 7/11; Revised 3/12; Published 6/12

Manifold Identification in Dual Averaging for Regularized Stochastic Online Learning

Sangkyun Lee SANGKYUN.LEE@TU-DORTMUND.DE
Fakultät für Informatik, LS VIII
Technische Universität Dortmund
44221 Dortmund, Germany

Stephen J. Wright SWRIGHT@CS.WISC.EDU
Computer Sciences Department
University of Wisconsin
1210 W. Dayton Street
Madison, WI 53706, USA

Editor: Léon Bottou

Abstract

Iterative methods that calculate their steps from approximate subgradient directions have proved to be useful for stochastic learning problems over large and streaming data sets. When the objective consists of a loss function plus a nonsmooth regularization term, the solution often lies on a low-dimensional manifold of parameter space along which the regularizer is smooth. (When an ℓ_1 regularizer is used to induce sparsity in the solution, for example, this manifold is defined by the set of nonzero components of the parameter vector.) This paper shows that a regularized dual averaging algorithm can identify this manifold, with high probability, before reaching the solution. This observation motivates an algorithmic strategy in which, once an iterate is suspected of lying on an optimal or near-optimal manifold, we switch to a "local phase" that searches in this manifold, thus converging rapidly to a near-optimal point. Computational results are presented to verify the identification property and to illustrate the effectiveness of this approach.

Keywords: regularization, dual averaging, partly smooth manifold, manifold identification

1. Introduction

Online learning algorithms based on stochastic approximation often are effective for solving large machine learning problems. Each step of these methods evaluates an approximate subgradient of the objective at the current iterate, based on a small subset (perhaps a single item) of the data. The amount of computation and data manipulation required per iteration is therefore small. Although many iterations are needed to converge to high-accuracy solutions, the tradeoffs between optimization errors and the other errors that arise in machine learning problems suggest that solutions of moderate accuracy often suffice. However, most existing stochastic algorithms do not produce approximate solutions that have the desirable structure (such as sparsity) that motivate the use of a regularization term in the objective.

We focus on the regularized dual averaging (RDA) approach developed by Nesterov (2009) and extended by Xiao (2010) to regularized problems. We show that, under appropriate assumptions, iterates generated by this method have a structure similar to the solution (with high probability) after

a sufficiently large number of iterations. This structure is characterized by a manifold, and identification of structure corresponds to identifying the manifold on which the solution lies. We sketch an algorithmic strategy that exploits this property by switching to a "local phase" that searches on the identified manifold, which often has much lower dimension than the full space.

1.1 Problem Setting and Regularized Dual Averaging

In *regularized stochastic learning*, we consider the following problem:

$$\min_{w\in\mathbb{R}^n} \phi(w) := f(w) + \Psi(w), \tag{1}$$

where

$$f(w) := \mathbb{E}_\xi[F(w;\xi)] = \int_\Xi F(w;\xi)dP(\xi), \tag{2}$$

and ξ is a random vector whose probability distribution P is supported on the set $\Xi \subset \mathbb{R}^d$. The regularization function $\Psi : \mathbb{R}^n \to \mathbb{R}\cup\{+\infty\}$ is assumed to be a closed proper convex function whose effective domain (denoted by $\operatorname{dom}\Psi$) is closed, and there is an open neighborhood $\mathcal{O}$ of $\operatorname{dom}\Psi$ that is contained in the domain of $F(\cdot,\xi)$, for all $\xi \in \Xi$. We assume that the expectation in (2) is well defined and finite-valued for all $w \in \mathcal{O}$ and that for every $\xi \in \Xi$, $F(w,\xi)$ is a smooth convex function of $w \in \mathcal{O}$. (An elementary argument shows that f is therefore convex.) We use w^* to denote a minimizer of (1).

The purpose of the regularizer Ψ is to promote certain desirable types of structure in the solution of (1). A common desirable property is *sparsity* (that is, w^* has few nonzero elements), which can be promoted by setting $\Psi(\cdot) = \lambda\|\cdot\|_1$ for some parameter $\lambda > 0$.

One method for finding an (approximate) solution to (1) is to draw random variables ξ_j for all $j \in \mathcal{N}$ independently from the space Ξ, where $\mathcal{N}$ is an index set of finite cardinality, and solve a *sample average approximation* (SAA) problem

$$\min_{w\in\mathbb{R}^n} \tilde{\phi}_{\mathcal{N}}(w) := \tilde{f}_{\mathcal{N}}(w) + \Psi(w) \tag{3}$$

where $\tilde{f}_{\mathcal{N}}(w) := \frac{1}{\operatorname{card}(\mathcal{N})}\sum_{j\in\mathcal{N}} F(w;\xi_j)$. This approach requires *batch* optimization, which does not scale well as $\operatorname{card}(\mathcal{N})$ grows.

Iteration t of a stochastic online learning approach examines a cost function $F(\cdot;\xi_t) : \mathbb{R}^n \to \mathbb{R}$ for some $\xi_t \in \Xi$, drawn randomly according to the distribution P, where $\{\xi_t\}_{t\geq 1}$ forms an i.i.d. sequence of samples. The next iterate w_{t+1} is obtained from information gathered up to the time t, the aim being to generate a sequence $\{w_t\}$ such that

$$\lim_{t\to\infty} \mathbb{E}[F(w_t;\xi)] + \Psi(w_t) = f(w^*) + \Psi(w^*). \tag{4}$$

We focus on objectives that consist of a smooth loss function F in conjunction with a nonsmooth regularizer Ψ. Xiao (2010) recently developed the *regularized dual averaging* (RDA) method, in which the smooth term is approximated by an averaged gradient in the subproblem at each iteration, while the regularization term appears explicitly. Xiao's approach extends the method of Nesterov (2009) in the sense that the regularization term is not handled explicitly in Nesterov's paper. Specifically, the RDA subproblem is

$$w_{t+1} = \arg\min_{w\in\mathbb{R}^n} \left\{ \langle \bar{g}_t, w\rangle + \Psi(w) + \frac{\beta_t}{t} h(w) \right\}, \tag{5}$$

where $\bar{g}_t = \frac{1}{t}\sum_{j=1}^{t} g_j$ and $g_j \in \partial F(w_j;\xi_j)$. The *prox-function* $h(\cdot)$ is a proper strongly convex function whose minimizers are also minimizers of Ψ, and for which the starting point w_1 of the algorithm is a minimizer of h. (The function $h(\cdot) = \|\cdot - w_1\|^2$ is a case of particular interest.) The sequence $\{\beta_t\}_{t\geq 1}$ is nonnegative.

A characteristic of problems with nonsmooth regularizers is that the solution often lies on a manifold of low dimension. In ℓ_1-regularized problems, for instance, the number of nonzero components at the solution is often a small fraction of the dimension of the full space. When a reliable method for identifying an optimal (or near-optimal) manifold is available, we have the possibility of invoking an algorithm that searches just in the low-dimensional space defined by this manifold—possibly a very different algorithm from the one used on the full space. One local-phase algorithm that is well suited to the ℓ_1 regularizer is the *LASSO-patternsearch* (LPS) (Shi et al., 2008; Wright, 2012), a batch optimization method for ℓ_1-regularized logistic regression, which takes gradient steps in the space of nonzero variables, enhanced by Newton-like scaling. In logistic regression, as in other applications, it can be much less expensive to compute first- and second-order information on a restricted subspace than on the full space.

A second motivation for aiming to identify the optimal manifold is that for problems of large dimension, it may be quite expensive even to *store* a single iterate w_t, whereas an iterate whose structure is similar to that of the solution w^* may be stored economically. (To take the case of ℓ_1 regularization again, we would need to store only the nonzero elements of w_t and their locations.)

In this paper, we investigate the ability of the RDA algorithm to identify the optimal manifold. We also describe an enhanced, two-phase version of this algorithm, which we call RDA$^+$. We test this approach on ℓ_1-regularized logistic regression problems, in which an LPS-based algorithm is used to explore the near-optimal manifold identified by RDA.

1.2 Optimal Manifolds and the Identification

Identification of optimal manifolds has been studied in the context of constrained convex optimization (Wright, 1993; Burke and Moré, 1994) and nonsmooth nonconvex optimization (Hare and Lewis, 2004). In constrained optimization, the optimal manifold is typically a face or edge of the feasible set that contains the solution. In nonsmooth optimization, the optimal manifold is a smooth surface passing through the optimum, parameterizable by relatively few variables, such that the restriction of the objective function to the manifold is smooth. In either case, when a certain nondegeneracy condition is satisfied, this manifold may be identified *without knowing the solution*, usually as a by-product of an algorithm for solving the problem. Lewis and Wright (2008) analyze a framework for composite minimization (which uses a subproblem in which f is replaced by an exact linearization around w_t) and prove identification results. Part of the motivation for the current paper is to obtain similar results as in Lewis and Wright (2008) in the stochastic gradient setting.

1.3 Alternative Stochastic Approximation Approaches

Stochastic approximation algorithms have a rich history and are currently the focus of intense research in the machine learning and optimization communities. We mention a few relevant developments here, and discuss their manifold identification properties.

Stochastic approximation methods often solve formulations in which an explicit constraint $w \in \mathcal{W}$ (for a compact convex set $\mathcal{W}$) is present. These can be incorporated into the framework (1) by

defining the regularization function Ψ as follows:

$$\Psi(w) := \delta_{\mathcal{W}}(w) + \psi(w),$$

where $\delta_{\mathcal{W}}(w)$ is the indicator function (zero on $\mathcal{W}$ and $+\infty$ elsewhere) and ψ is a convex function whose domain includes $\mathcal{W}$. (In this setting, $\mathcal{O}$ is taken to be an open neighborhood of $\mathcal{W}$.) The classical approaches to solve such problems can be traced back to Robbins and Monro (1951) and Kiefer and Wolfowitz (1952). The *stochastic gradient descent* (SGD) method generates its iterates by stepping in the direction of a subgradient estimate and then projecting onto $\mathcal{W}$, as follows:

$$w_{t+1} = \Pi_{\mathcal{W}}\left(w_t - \alpha_t(g_t + \kappa_t)\right), \ \ t = 1, 2, \ldots$$

where $g_t \in \partial F(w_t; \xi_t)$ for some sampled random variable ξ_t, $\kappa_t \in \partial\psi(w_t)$, and $\Pi_{\mathcal{W}}(\cdot)$ denotes the Euclidean projection onto the set $\mathcal{W}$. With steplength choice of the form $\alpha_t = \theta/t$ (for a well-chosen constant θ), the SGD method exhibits $O(1/t)$ convergence rate in the quantity (4) for strongly convex functions f (Chung, 1954; Sacks, 1958). For general convex functions, a step length of the form $\alpha_t = \theta'/\sqrt{t}$ (for some $\theta' > 0$) yields an $O(1/\sqrt{t})$ rate of convergence in the function value (Nemirovski and Yudin, 1978; Polyak, 1990; Polyak and Juditsky, 1992). Simplified proofs of these rates are given by Nemirovski et al. (2009). These rates are known to be optimal for subgradient schemes in "black-box" algorithmic models (Nemirovski and Yudin, 1983). Although batch optimization methods based on an approximate objective, such as (3), may provide better convergence rates, SGD has been widely used in machine learning because it scales well to large data sets and provides good generalization performance in practice (Zinkevich, 2003; Bottou, 2004; Zhang, 2004; Shalev-Shwartz et al., 2007).

As discussed by Xiao (2010), the SGD method does not exploit the problem structure that is present in the regularizer ψ, and there is no reason to believe that these algorithms have good manifold identification properties. When $\psi(\cdot) = \|\cdot\|_1$, for instance, there is no particular reason for the iterates w_t to be sparse. Though equal in expectation, g_t and $\nabla f(x_t)$ may be far apart, so that even if a careful choice of κ_t is made at each iteration, the updates may have the cumulative effect of destroying sparsity in the iterates w_t.

Variants of SGD for the general convex case often work with averaged primal iterates, of the form

$$\bar{w}_t := \frac{\sum_{j=1}^{t} \nu_j w_j}{\sum_{j=1}^{t} \nu_j}, \tag{6}$$

where the ν_j are nonnegative weights (see, for example, Nemirovski et al., 2009). Averaging does not improve identification properties. For ℓ_1 regularization, we can still expect the averaged iterates $\bar{w}_t$ to be at least as dense as the "raw" iterates w_t.

Recently, various authors have proposed modifications of SGD that account for the regularization structure. Some representative examples include *composite objective mirror descent* (COMID) (Duchi et al., 2010), *forward-backward splitting* (FOBOS) (Duchi and Singer, 2009), *truncated gradient* (TG) (Langford et al., 2009), and *sparsity-preserving stochastic gradient* (SSG) (Lin et al., 2011). The basic FOBOS subproblem is similar to that of the prox-descent algorithm of Lewis and Wright (2008) and the SpaRSA framework of Wright et al. (2009), which generate their iterates by setting $g_t = \nabla f(w_t)$ and solving

$$w_{t+1} = \arg\min_{w \in \mathbb{R}^n} \left\{ \langle g_t, w\rangle + \Psi(w) + \frac{1}{2\alpha_t}\|w - w_t\|_2^2 \right\}, \tag{7}$$

for some parameter $\alpha_t > 0$ (which plays a step-length-like role). Duchi and Singer (2009) suggest an extension to an online setting, in which g_t is replaced by an approximate gradient. SSG (Lin et al., 2011) also has the subproblem (7) at its core, but embeds it in a strategy for generating *three* sequences of iterates (rather than the single sequence $\{w_t\}$), extending an idea of Nesterov (2004). The TG method (Langford et al., 2009) solves a subproblem like (7) on some iterations; on other iterations it simply steps in the direction g_t of the latest gradient estimate for f, ignoring the regularization term. COMID (Duchi et al., 2010) is also based on a subproblem of the form (7), but with a Bregman divergence replacing the final quadratic term, thus yielding a more general framework.

Since all these methods make explicit use of the regularization term Ψ in the subproblem, they are more likely to generate iteration sequences that share the structure of the solution w^*, that is, to identify a near-optimal manifold. Such behavior is far from guaranteed, however, because g_t may be only a rough approximation to $\nabla f(w_t)$. The inaccuracy of this gradient estimate may cause the iterates to step away from the optimal manifold, even from an iterate w_t that is close to the solution w^*. (In Example 1 at the end of Section 4, we discuss a function satisfying all the assumptions of this paper, in which the subproblem (7) steps away from the optimal point and off the optimal manifold.) In contrast, the dual average $\bar{g}_t$ used by the RDA subproblem stabilizes around $\nabla f(w^*)$ as the iterates converge to w^*, allowing identification results to be derived from analysis like that of Hare and Lewis (2004) and Lewis and Wright (2008), suitably generalized to the stochastic setting.

Averaging of the primal iterates (6) does not improve the identification properties of the methods above, and will usually make them worse. Considering again ℓ_1 regularization, we observe that if component i of *any* iterate w_t is nonzero, then component i of all averaged iterates at subsequent iterations will also be nonzero (unless some fortuitous cancellation occurs). RDA itself, in the version recommended by the analysis of Xiao (2010), has the same deficiency, as the main convergence results in that paper are for averaged iterates (6). In the current paper, we facilitate the use of the raw iterates w_t by RDA by adding two assumptions. First, w^* is assumed to be a *strong* minimizer of the restriction of ϕ to the optimal manifold. Second, a nondegeneracy condition ensures that ϕ increases sharply as we move off the manifold. Together, these conditions ensure that w^* is a strong minimizer of ϕ, so convergence of $\phi(w_t)$ to $\phi(w^*)$ forces convergence of w_t to w^*.

Convergence analysis of stochastic approximation algorithms focuses largely on the *regret*, which is defined as follows, for any instantiation of the random sequence $\{w_t\}_{t\geq 1}$ with respect to any fixed decision $w \in \operatorname{dom}\Psi$:

$$R_t(w) := \sum_{j=1}^{t} [F(w_j;\xi_j) + \Psi(w_j)] - \sum_{j=1}^{t} [F(w;\xi_j) + \Psi(w)]. \tag{8}$$

As we discuss later, the RDA algorithm has $O(\sqrt{t})$ regret bounds when $\beta_t = O(\sqrt{t})$ for general convex cases, and $O(\ln t)$ bounds with $\beta_t = O(\ln t)$ for strongly convex cases. These bounds are comparable to those of the SGD method. For general convex cases, when we use $\alpha_t = O(1/\sqrt{t})$, SGD achieves an $O(\sqrt{t})$ regret bound (see, for example, Zinkevich, 2003; Nemirovski et al., 2009), which cannot be improved in general. For the strongly convex case, SGD has an $O(\ln t)$ regret bound (see, for example, Hazan et al., 2006; Bartlett et al., 2008) with $\alpha_t = O(1/t)$.

1.4 Notation and Terminology

Throughout the paper, we use $\|\cdot\|$ (without a subscript) to denote the Euclidean norm $\|\cdot\|_2$, and $\mathrm{card}(M)$ to denote the cardinality of a finite set M. The distance function $\mathrm{dist}\,(w,C)$ for $w \in \mathbb{R}^n$ and a convex set $C \subset \mathbb{R}^n$ is defined by $\mathrm{dist}\,(w,C) := \inf_{c\in C} \|w-c\|$. The *effective domain* of $\Psi : \mathbb{R}^n \to \mathbb{R} \cup \{+\infty\}$ is defined by $\mathrm{dom}\,\Psi := \{w \in \mathbb{R}^n \,|\, \Psi(w) < +\infty\}$. ri C denotes the *relative interior* of a convex set C, that is, the interior relative to the affine span of C (the smallest affine set which can be expressed as the intersection of hyperplanes containing C).

We call a function $\varphi : \mathbb{R}^n \to \mathbb{R} \cup \{+\infty\}$ *strongly convex* if there exists a constant $\sigma > 0$ such that $\forall w, w' \in \mathrm{dom}\,\varphi$ and $\forall \alpha \in [0,1]$,

$$\varphi(\alpha w + (1-\alpha)w') \le \alpha\varphi(w) + (1-\alpha)\varphi(w') - \frac{\sigma}{2}\alpha(1-\alpha)\|w-w'\|^2.$$

(σ is known as the *modulus of convexity*.) Strong convexity implies that for any $w \in \mathrm{dom}\,\varphi$ and $w' \in \mathrm{ri}\,\mathrm{dom}\,\varphi$, we have

$$\varphi(w) \ge \varphi(w') + \langle s, w-w'\rangle + \frac{\sigma}{2}\|w-w'\|^2, \quad \forall s \in \partial\varphi(w'). \tag{9}$$

We say that a function φ has a *locally strong minimizer* at w^* if there exist positive constants c and $\bar{r}$ such that

$$\varphi(w) - \varphi(w^*) \ge c\|w-w^*\|^2, \text{ for all } w \in \mathcal{O} \text{ with } \|w-w^*\| \le \bar{r}.$$

w^* is a *globally strong minimizer* if this expression is true with $\bar{r} = \infty$.

The algorithm we consider in this paper makes use of an i.i.d. sequence $\{\xi_j\}_{j\ge 1}$ of random variables drawn from Ξ according to the distribution P. We denote the history of random variables up to time t by

$$\xi_{[t]} := \{\xi_1, \xi_2, \ldots, \xi_t\}.$$

The iterate w_t produced by the algorithm depends on $\xi_1, \xi_2, \ldots, \xi_{t-1}$ but not on $\xi_t, \xi_{t+1}, \cdots$; we sometimes emphasize this fact by writing $w_t = w_t(\xi_{[t-1]})$.

2. Assumptions and Basic Results

We summarize here our fundamental assumptions about the problem and its solution, together with some basic observations and results that will be used in later sections.

2.1 Unbiasedness

As in Nemirovski et al. (2009), we assume the following *unbiasedness* property:

$$\nabla f(w) = \nabla_w \mathbb{E}[F(w;\xi)] = \mathbb{E}[\nabla_w F(w;\xi)] \tag{10}$$

for any w independent of ξ. (As the differentiation of F is taken only for its first argument, we omit the subscript "w" in subsequent discussions.) Given that $w_t = w_t(\xi_{[t-1]})$, this implies

$$\mathbb{E}[\nabla F(w_t;\xi_t)] = \mathbb{E}\left[\mathbb{E}[\nabla F(w_t;\xi_t)|\xi_{[t-1]}]\right] = \mathbb{E}\left[\nabla f(w_t)\right].$$

2.2 Uniform Lipschitz Continuity

We assume that each $F(w;\xi)$ is a smooth convex function of $w \in \mathcal{O}$ for every $\xi \in \Xi$, and in particular that $\nabla F(\cdot;\xi)$ is uniformly Lipschitz continuous with respect to its first argument, over all ξ. That is, there exists a constant $L > 0$ such that

$$\|\nabla F(w;\xi) - \nabla F(w';\xi)\| \leq L\|w - w'\|, \ \ \forall w, w' \in \mathcal{O}, \ \ \forall \xi \in \Xi. \tag{11}$$

We further assume that there exists a uniform bound G for which

$$\|\nabla F(w;\xi)\| \leq G, \ \ \forall w \in \mathcal{O}, \ \forall \xi \in \Xi. \tag{12}$$

These assumptions immediately lead to similar properties on ∇f. We prove this claim after noting a simple consequence of Jensen's inequality, whose proof is omitted.

Lemma 1 *For a vector-valued function $v : \Xi \to \mathbb{R}^n$ which is integrable with respect to P, we have*

$$\|\mathbb{E}v\|_2 \leq \mathbb{E}\|v\|_2.$$

Lemma 2 *If $\nabla F(w;\xi)$ satisfies the uniform Lipschitz continuity assumption* (11)*, then $\nabla f(w)$ is also uniformly Lipschitz continuous on $\mathcal{O}$ with the same constant L. If $\nabla F(w;\xi)$ satisfies the uniform bound* (12)*, then ∇f satisfies the same bound, that is, $\|\nabla f(x)\| \leq G$ for all $w \in \mathcal{O}$.*

Proof From unbiasedness, we have for $w, w' \in \mathcal{O}$ independent of ξ that

$$\begin{aligned}
\nabla f(w) &= \nabla \mathbb{E}[F(w;\xi)] = \mathbb{E}[\nabla F(w;\xi)] && \text{from (10)} \\
&= \mathbb{E}[\nabla F(w';\xi) + u_\xi] && \text{for } u_\xi := \nabla F(w;\xi) - \nabla F(w';\xi) \\
&= \nabla f(w') + \mathbb{E}[u_\xi] && \text{from (10) again.}
\end{aligned}$$

Since $\|u_\xi\| \leq L\|w - w'\|$, we have

$$\|\nabla f(w) - \nabla f(w')\| = \|\mathbb{E}u_\xi\| \leq \mathbb{E}\|u_\xi\| \leq L\|w - w'\|,$$

where the first inequality is due to Lemma 1. This proves the first claim.

For the second claim, we have

$$\|\nabla f(w)\| = \|\mathbb{E}[\nabla F(w;\xi)]\| \leq \mathbb{E}\|\nabla F(w;\xi)\| \leq G,$$

as required. ■

2.3 Optimality and Nondegeneracy

We specify several optimality conditions that are assumed to hold throughout the paper. The optimality of w^* for the problem (1) can be characterized as follows:

$$0 \in \nabla f(w^*) + \partial\Psi(w^*).$$

We assume that w^* is a *nondegenerate* solution—one that satisfies the stronger condition

$$0 \in \text{ri } [\nabla f(w^*) + \partial\Psi(w^*)]. \tag{13}$$

The nondegeneracy assumption is common in manifold identification analyses. It can be replaced with weaker assumptions to prove weaker results (see, for instance, Oberlin and Wright, 2006), though the analysis is somewhat more complicated.

2.4 Manifolds and Partial Smoothness

In this section we discuss properties of differential manifolds and partial smoothness by repeating some definitions from Hare and Lewis (2004).

Definition 3 (Manifold) *A set $\mathcal{M} \subset \mathbb{R}^n$ is a* manifold *about $\bar{z} \in \mathcal{M}$ if it can be described locally by a collection of $\mathcal{C}^p$ functions ($p \geq 2$) with linearly independent gradients. That is, there exists a map $H : \mathbb{R}^n \to \mathbb{R}^k$ that is $\mathcal{C}^p$ around $\bar{z}$ with $\nabla H(\bar{z})^T \in \mathbb{R}^{k \times n}$, surjective, such that points z near $\bar{z}$ lie in $\mathcal{M}$ if and only if $H(z) = 0$.*

The *normal space* to $\mathcal{M}$ at $\bar{z}$, denoted by $N_{\mathcal{M}}(\bar{z})$, is the range space of $\nabla H(\bar{z})$, while the *tangent space* to $\mathcal{M}$ at $\bar{z}$ is the null space of $\nabla H(\bar{z})^T$. We assume without loss of generality that $\nabla H(\bar{z})$ has orthonormal columns.

We define *partial smoothness* as follows (Lewis, 2003, Section 2).

Definition 4 (Partial Smoothness) *A function $\varphi : \mathbb{R}^n \to \mathbb{R} \cup \{+\infty\}$ is ($\mathcal{C}^2$-)* partly smooth *at a point $\bar{z} \in \mathbb{R}^n$ relative to a set $\mathcal{M} \subset \mathbb{R}^n$ containing $\bar{z}$ if $\mathcal{M}$ is a manifold about $\bar{z}$ and the following properties hold:*

(i) (Smoothness) *The function φ restricted to $\mathcal{M}$, denoted by $\varphi|_{\mathcal{M}}$, is $\mathcal{C}^2$ near $\bar{z}$,*

(ii) (Regularity) *φ is subdifferentially regular at all points $z \in \mathcal{M}$ near $\bar{z}$, with $\partial\varphi(z) \neq \emptyset$,*

(iii) (Sharpness) *The affine span of $\partial\varphi(\bar{z})$ is a translate of $N_{\mathcal{M}}(\bar{z})$, and*

(iv) (Sub-continuity) *The set-valued mapping $\partial\varphi : \mathcal{M} \rightrightarrows \mathbb{R}^n$ is continuous at $\bar{z}$.*

We refer to $\mathcal{M}$ as the active manifold *at the point $\bar{z}$, and as the* optimal manifold *when $\bar{z} = w^*$, where w^* is a solution of* (1).

The regularity condition (ii) is discussed by Lewis (2003, p. 706); for our purposes it suffices to note that this condition holds for closed convex functions and continuously differentiable functions, and hence for our objective ϕ. Henceforth, we assume that Ψ is partly smooth at w^* relative to the optimal manifold, which implies partial smoothness of ϕ (by an argument like that of Lemma 2).

We discuss the concepts outlined in the definitions above for the specific case of $\Psi(\cdot) = \|\cdot\|_1$. Given $\bar{z} \in \mathbb{R}^n$, we define the following partition of the indices $\{1, 2, \ldots, n\}$:

$$\{1, 2, \ldots, n\} = \mathcal{P} \cup \mathcal{N} \cup \mathcal{Z},$$

where $\bar{z}_i = 0$ for all $i \in \mathcal{Z}$, $\bar{z}_i > 0$ for all $i \in \mathcal{P}$, and $\bar{z}_i < 0$ for all $i \in \mathcal{N}$. A natural definition of the active manifold $\mathcal{M}$ is thus

$$\mathcal{M} = \{z \in \mathbb{R}^n \,|\, z_i = 0 \text{ for all } i \in \mathcal{Z}\}.$$

Note that $\bar{z} \in \mathcal{M}$, and that the mapping H of Definition 3 can be defined as $H(z) = [z_i]_{i \in \mathcal{Z}}$, with $k = \text{card}(\mathcal{Z})$ in that definition. The restriction of Ψ to this manifold thus has the following form for all $z \in \mathcal{M}$ near $\bar{z}$:

$$-\sum_{i \in \mathcal{N}} z_i + \sum_{i \in \mathcal{P}} z_i,$$

so that (i) of Definition 4 is satisfied. For $z \in \mathcal{M}$ near $\bar{z}$, we have

$$[\partial\phi(z)]_i = [\nabla f(z)]_i + \begin{cases} [-1,1] & \text{for } i \in \mathcal{Z}, \\ -1 & \text{for } i \in \mathcal{N}, \\ +1 & \text{for } i \in \mathcal{P}. \end{cases}$$

Clearly, condition (ii) of Definition 4 holds. The affine span of $\partial\phi(\bar{z})$ is

$$\{\nabla f(\bar{z}) + g \,|\, g_i = -1 \text{ for } i \in \mathcal{N},\ g_i = +1 \text{ for } i \in \mathcal{P}, \text{ and } g_i \in \mathbb{R} \text{ for } i \in \mathcal{Z}\},$$

whereas $N_{\mathcal{M}}(\bar{z}) = \{g \,|\, g_i = 0 \text{ for } i \in \mathcal{P} \cup \mathcal{N},\ g_i \in \mathbb{R} \text{ for } i \in \mathcal{Z}\}$. Comparison of the last two expressions shows that (iii) of Definition 4 is satisfied. Finally, the set-valued map $\partial\Psi$ is in fact constant along $\mathcal{M}$ near $\bar{z}$, so because f is smooth, Definition 4 (iv) also holds.

2.5 Strong Minimizer Properties

We assume that w^* is a *locally strong minimizer* of ϕ relative to the optimal manifold $\mathcal{M}$ with modulus $c_{\mathcal{M}}$, that is, there exists $c_{\mathcal{M}} > 0$ and $r_{\mathcal{M}} > 0$ such that $\{w \in \mathbb{R}^n \,|\, \|w - w^*\| \leq r_{\mathcal{M}}\} \subset \mathcal{O}$ and

$$\phi|_{\mathcal{M}}(w) \geq \phi|_{\mathcal{M}}(w^*) + c_{\mathcal{M}}\|w - w^*\|^2, \text{ for all } w \in \mathcal{M} \text{ with } \|w - w^*\| \leq r_{\mathcal{M}}. \tag{14}$$

Together with a nondegeneracy assumption (13), these conditions imply that w^* is a locally strong minimizer of $\phi(w)$ (without restriction to the optimal manifold), as we now prove.

Theorem 5 (Strong Minimizer for General Convex Case) *Suppose that ϕ is partly smooth at w^* relative to the optimal manifold $\mathcal{M}$, that w^* is a locally strong minimizer of $\phi|_{\mathcal{M}}$ with the modulus $c_{\mathcal{M}} > 0$ and radius $r_{\mathcal{M}} > 0$ defined in* (14)*, and that the nondegeneracy condition* (13) *holds at w^*. Then there exist $c \in (0, c_{\mathcal{M}}]$ and $\bar{r} \in (0, r_{\mathcal{M}}]$ such that*

$$\phi(w) - \phi(w^*) \geq c\|w - w^*\|^2, \text{ for all } w \text{ with } \|w - w^*\| \leq \bar{r}, \tag{15}$$

that is, w^* is a locally strong minimizer of ϕ, without restriction to the manifold $\mathcal{M}$.

Proof The proof is a simplification of the proof of Wright (2012, Theorem 2.5), which considers the more general case in which f is prox-regular rather than convex. For completeness, we present the proof in Appendix A. ■

Henceforth, we assume without loss of generality that $\bar{r} \in (0, 1]$.

The condition (15) is similar to the quadratic growth condition proposed by Anitescu (2000) in the context of nonlinear programming. It was shown by Anitescu that this fundamental condition is weaker than many other second-order conditions that are widely used in nonlinear programming.

Two corollaries follow in a straightforward fashion from the theorem above. We state these results here and give their proofs in Appendix A.

Corollary 6 *Suppose that w^* is a locally strong minimizer of* (1) *that satisfies* (15)*. For all $w \in \mathcal{O}$ with $\|w - w^*\| > \bar{r}$, we have*

$$\phi(w) - \phi(w^*) > c\bar{r}\|w - w^*\|.$$

Corollary 7 (Globally Strong Minimizer for Strongly Convex Case) *Suppose that w^* is a locally strong minimizer of* (1) *satisfying* (15)*. If ϕ is strongly convex on* $\operatorname{dom}\Psi$ *with modulus $\sigma > 0$, then w^* is a globally strong minimizer of* (1) *with modulus* $\min(c, \sigma/2)$*, that is,*

$$\phi(w) \geq \phi(w^*) + \min(c, \sigma/2)\|w - w^*\|^2, \text{ for all } w \in \mathcal{O}. \tag{16}$$

2.6 Summary of Assumptions

We summarize here the assumptions introduced in this section, for reference in the remainder of the paper.

The first assumption summarizes basic properties of the functions and the minimizer.

Assumption 1 *The unbiasedness property* (10)*, uniform Lipschitz continuity of* $\nabla F(\cdot;\xi)$ *for all* $\xi \in \Xi$ (11)*, uniform boundedness of* $\|\nabla F(\cdot,\xi)\|$ (12)*, and nondegeneracy at the optimum* w^* (13) *are satisfied.*

The second assumption provides sufficient conditions for w^* to be a locally strong minimizer.

Assumption 2 *The function* ϕ *is partly smooth at its minimizer* w^* *relative to the optimal manifold* $\mathcal{M}$ *and* w^* *is a locally strong minimizer of* $\phi|_{\mathcal{M}}$ *as defined in* (14)*.*

3. Regularized Dual Averaging Algorithm

We start this section by describing regret bounds for the regularized dual averaging (RDA) algorithm of Xiao (2010) (following Nesterov, 2009), focusing on its stochastic variant. We also describe the consequences for the analysis of the condition that the minimum is strong locally (15) or globally (16). We then analyze the properties of the averaged gradient; this analysis forms the basis of the manifold identification result in Section 4.

3.1 The RDA Algorithm

We start by specifying the RDA algorithm from Xiao (2010), noting that our assumptions on the functions F, f, and Ψ from Section 1 are stronger than the corresponding conditions in Xiao (2010), which require only subdifferentiability of $F(w;\xi_t)$ on $\operatorname{dom}\Psi$.

As introduced in (5), the prox-function $h : \mathbb{R}^n \to \mathbb{R} \cup \{+\infty\}$ is proper, strongly convex on $\operatorname{dom}\Psi$, and subdifferentiable on $\operatorname{ri}\operatorname{dom}\Psi$. In addition, we require h to satisfy

$$\arg\min_w h(w) \in \arg\min_w \Psi(w).$$

The *prox-center* w_1 of $\operatorname{dom}\Psi$ with respect to h, which is used as the starting point of the RDA method, is defined as follows:

$$w_1 := \arg\min_{w \in \operatorname{dom}\Psi} h(w).$$

The terms "prox-function" and "prox-center" are borrowed from Nesterov (2009). Following Xiao (2010), we assume without loss of generality that the strong convexity modulus of $h(w)$ is one, and that $\min_w h(w) = \min_w \Psi(w) = 0$. This implies $h(w_1) = 0$ and $\Psi(w_1) = 0$. The most obvious prox-function is $h(\cdot) = \frac{1}{2}\|\cdot - w_1\|^2$, where $w_1 \in \arg\min_w \Psi(w)$.

We now define a constant D that reappears throughout the analysis. For any $D > 0$, we consider a level set of the prox-function h defined as follows:

$$\mathcal{F}_D := \{w \in \operatorname{dom}\Psi \mid h(w) \le D^2\}.$$

We assume in the analysis that points of interest (specifically, w^*) lie in $\mathcal{F}_D$. Because of (9) and our assumptions on h (in particular, $h(w_1) = 0$, $0 \in \partial h(w_1)$, and h has modulus of convexity 1), we have that

$$w \in \mathcal{F}_D \;\Rightarrow\; D^2 \ge h(w) \ge \frac{1}{2}\|w - w_1\|^2 \;\Rightarrow\; \|w - w_1\| \le \sqrt{2}D. \tag{17}$$

Algorithm 1: The RDA Algorithm.

Input:

- a prox-function $h(w)$ that is strongly convex on $\operatorname{dom}\Psi$ and also satisfies

$$\arg\min_{w\in\mathbb{R}^n} h(w) \in \arg\min_{w\in\mathbb{R}^n} \Psi(w).$$

- a nonnegative and nondecreasing sequence $\{\beta_t\}_{t\geq 1}$.

Initialize: set $w_1 = \arg\min h(w)$ and $\bar{g}_0 = 0$

for $t = 1, 2, \ldots$ **do**

Sample ξ_t from Ξ and compute a gradient $g_t = \nabla F(w_t; \xi_t)$;

Update the average gradient:

$$\bar{g}_t = \frac{t-1}{t}\bar{g}_{t-1} + \frac{1}{t}g_t.$$

Compute the next iterate:

$$w_{t+1} = \arg\min_{w\in\mathbb{R}^n} \left\{ \langle \bar{g}_t, w\rangle + \Psi(w) + \frac{\beta_t}{t}h(w) \right\}. \tag{18}$$

end

At iteration t, the stochastic RDA algorithm samples a vector $\xi_t \in \Xi$, according to the distribution P, and evaluates an approximate gradient as follows:

$$g_t := \nabla F(w_t; \xi_t).$$

We assume that the random variables ξ_t are i.i.d. The *dual average*—an averaged approximation to the gradient of f—is defined as follows:

$$\bar{g}_t := \frac{1}{t}\sum_{j=1}^{t} g_j = \frac{1}{t}\sum_{j=1}^{t} \nabla F(w_j; \xi_j). \tag{19}$$

The RDA algorithm is specified in Algorithm 1. As the objective function in the subproblem (18) is strongly convex when $\beta_t > 0$ or when $\Psi(\cdot)$ is strongly convex (at least one of which we assume for the analysis below), w_{t+1} is uniquely defined by (18). Note that w_{t+1} depends on the history of random variables $\xi_{[t]}$ up to iteration t, and is independent of later samples $\xi_{t+1}, \xi_{t+2}, \ldots$.

We consider two choices of parameter sequences $\{\beta_t\}$ in the remainder of the paper, depending on whether the regularization function Ψ is strongly convex or not. The first choice holds for general convex regularizers Ψ:

$$\beta_t = \gamma\sqrt{t}, \quad \forall t \geq 1, \text{ for some constant } \gamma > 0. \tag{20}$$

The second choice, $\beta_t = O(1+\ln t)$, can be used when Ψ is a strongly convex function, with modulus $\sigma > 0$. Among the three choices discussed in Xiao (2010), that is, $\beta_t = \sigma$, $\beta_t = \sigma(1+\ln t)$, and $\beta_t = 0$, we focus on the zero sequence, which gives the simplest bounds:

$$\beta_t = 0, \quad \forall t \geq 1. \tag{21}$$

In this case, the subproblem (18) in Algorithm RDA has no damping term; we rely instead on the damping effect supplied by the strong convexity of Ψ to stabilize the iterates.

3.2 Bounds on Regret and Expected Errors in the Iterations

Our first key result concerns bounds on the regret function defined in (8).

Theorem 8 *Suppose that the unbiasedness and uniform boundedness properties in Assumption 1 are satisfied. When $\{\beta_t\}$ is chosen according to* (20)*, we have for any $w \in \mathcal{F}_D$ that*

$$R_t(w) \leq \left(\gamma D^2 + \frac{G^2}{\gamma}\right)\sqrt{t}, \qquad t \geq 1. \tag{22}$$

Moreover, when $\Psi(w)$ is strongly convex with the modulus $\sigma > 0$, then the choice (21) *for $\{\beta_t\}$ results in the following bound for $w \in \mathcal{F}_D$:*

$$R_t(w) \leq \frac{G^2}{2\sigma}(6 + \ln t), \qquad t \geq 1. \tag{23}$$

Proof See Xiao (2010, Corollary 2) for the general convex case, and Xiao (2010, Theorem 1 and Section 3.2) for the strongly convex case. ■

The next result obtains bounds on the expected errors in the iterates generated by Algorithm 1. For the purpose of this and future results, we define the indicator function $I_{(A)}$ for the event A as follows

$$I_{(A)} = \begin{cases} 1 & \text{if event } A \text{ is true,} \\ 0 & \text{if event } A \text{ is false.} \end{cases}$$

For a random event A, $I_{(A)}$ is a random variable.

Lemma 9 (Expected Error Bounds of Iterates) *Suppose that Assumptions 1 and 2 are satisfied, and that $w^* \in \mathcal{F}_D$. Then for the iterates $w_1, w_2, \ldots, w_t$ generated by the stochastic RDA algorithm with $\{\beta_t\}$ chosen by* (20)*, we have*

$$\frac{1}{t}\sum_{j=1}^{t} \mathbb{E}\left[I_{(\|w_j - w^*\| \leq \bar{r})}\|w_j - w^*\|^2\right] \leq \frac{1}{c}\left(\gamma D^2 + \frac{G^2}{\gamma}\right) t^{-1/2}, \tag{24}$$

$$\frac{1}{t}\sum_{j=1}^{t} \mathbb{E}\left[I_{(\|w_j - w^*\| > \bar{r})}\|w_j - w^*\|\right] \leq \frac{1}{c\bar{r}}\left(\gamma D^2 + \frac{G^2}{\gamma}\right) t^{-1/2}. \tag{25}$$

When $\Psi(w)$ is strongly convex with the modulus $\sigma > 0$, then with $\{\beta_t\}$ chosen by (21) *we have*

$$\frac{1}{t}\sum_{j=1}^{t} \mathbb{E}\left[\|w_j - w^*\|^2\right] \leq \frac{G^2}{2\sigma \min(c, \sigma/2)} \frac{6 + \ln t}{t}. \tag{26}$$

Proof See Appendix B. ■

Note the differences between (24) and (25): the norms in the summation are squared in the first expression but not in the second, which includes an extra factor of $1/\bar{r}$. The next result combines these bounds into a more useful form for the results that follow.

Theorem 10 *Suppose the assumptions of Lemma 9 are satisfied. Then for the iterates $w_1, w_2, \ldots, w_t$ generated by the stochastic RDA algorithm with the choice* (20) *for $\{\beta_t\}$, we have*

$$\frac{1}{t}\sum_{j=1}^{t}\mathbb{E}\|w_j - w^*\| \leq \mu t^{-1/4}, \tag{27}$$

for all $t \geq \hat{t}$, where the constants $\hat{t}$ and μ are defined as follows:

$$\mu := \frac{2}{\sqrt{c\bar{r}}}\left(\gamma D^2 + \frac{G^2}{\gamma}\right)^{1/2}, \qquad \hat{t} := \left\lceil \frac{1}{\bar{r}^2 c^2}\left(\gamma D^2 + \frac{G^2}{\gamma}\right)^2 \right\rceil. \tag{28}$$

When $\Psi(w)$ is strongly convex with the modulus $\sigma > 0$, then with the choice (21) *for $\{\beta_t\}$ we have*

$$\frac{1}{t}\sum_{j=1}^{t}\mathbb{E}\|w_j - w^*\| \leq \mu'\left(\frac{6+\ln t}{t}\right)^{1/2} \tag{29}$$

for the constant μ' defined by

$$\mu' := \frac{G}{\sqrt{2\sigma\min(c,\sigma/2)}}. \tag{30}$$

Proof See Appendix B. ■

3.3 Stochastic Behavior of the Dual Average

We now study the convergence properties of the dual average $\bar{g}_t$, showing that the probability that $\bar{g}_t$ lies outside any given ball around $\nabla f(w^*)$ goes to zero as t increases.

Theorem 11 *Suppose that Assumptions 1 and 2 are satisfied and that $w^* \in \mathcal{F}_D$, and let $\varepsilon > 0$ be any chosen positive constant. When $\{\beta_t\}$ is chosen from* (20)*, we have*

$$\mathbb{P}(\|\bar{g}_t - \nabla f(w^*)\| \geq \varepsilon) \leq 2n\exp\left(-\frac{\varepsilon^2 t}{32n^2G^2}\right) + 2\mu\varepsilon^{-1}Lt^{-1/4}, \quad \forall t \geq \hat{t}, \tag{31}$$

where μ and $\hat{t}$ are defined in (28)*. When Ψ is strongly convex and the choice* (21) *is used for $\{\beta_t\}$, we have*

$$\mathbb{P}(\|\bar{g}_t - \nabla f(w^*)\| \geq \varepsilon) \leq 2n\exp\left(-\frac{\varepsilon^2 t}{32n^2G^2}\right) + 2\mu'\varepsilon^{-1}L\left(\frac{6+\ln t}{t}\right)^{1/2}, \quad \forall t \geq 1, \tag{32}$$

where μ' is defined in (30)*.*

Proof Using the definition (19) of $\bar{g}_t$ and the unbiasedness property (10) that $\mathbb{E}[\nabla F(w_j;\xi_j) \mid \xi_{[j-1]}] = \nabla f(w_j)$, we can write

$$t[\bar{g}_t - \nabla f(w^*)] = \sum_{j=1}^{t}\left\{\nabla F(w_j;\xi_j) - \mathbb{E}[\nabla F(w_j;\xi_j) \mid \xi_{[j-1]}]\right\} + \sum_{j=1}^{t}\left\{\nabla f(w_j) - \nabla f(w^*)\right\}.$$

Considering the norms of the vectors in both sides and using the Lipschitz property of $\nabla f(\cdot)$ in Lemma 2, we get

$$t\|\bar{g}_t - \nabla f(w^*)\| \leq \left\| \sum_{j=1}^{t} z_j \right\| + L \sum_{j=1}^{t} \|w_j - w^*\|, \tag{33}$$

where we define $z_j := \nabla F(w_j;\xi_j) - \mathbb{E}[\nabla F(w_j;\xi_j) \mid \xi_{[j-1]}]$.

For the ith component $[z_j]_i$ of the vector $z_j \in \mathbb{R}^n$, $\sum_{j=1}^{t}[z_j]_i$ forms a martingale with bounded differences since $|[z_j]_i| \leq 2G$ from (12) and $\mathbb{E}[z_j \mid \xi_{[j-1]}] = 0$. Therefore by Hoeffding-Azuma inequality (Azuma, 1967) we obtain for any $\theta > 0$,

$$\mathbb{P}\left(\left| \sum_{j=1}^{t} [z_j]_i \right| \geq \theta \right) \leq 2\exp\left(-\frac{\theta^2}{8tG^2} \right), \ \ i = 1, 2, \ldots, n.$$

Therefore, using the equivalence relation $\|v\|_2 \leq \|v\|_1$ of norms for a vector $v \in \mathbb{R}^n$, we have

$$\begin{aligned} \mathbb{P}\left(\left\| \sum_{j=1}^{t} z_j \right\|_2 \geq \theta \right) &\leq \mathbb{P}\left(\left\| \sum_{j=1}^{t} z_j \right\|_1 \geq \theta \right) = \mathbb{P}\left(\sum_{i=1}^{n} \left| \sum_{j=1}^{t} [z_j]_i \right| \geq \theta \right) \\ &\leq \mathbb{P}\left(\bigcup_{i=1}^{n} \left\{ \left| \sum_{j=1}^{t} [z_j]_i \right| \geq \frac{\theta}{n} \right\} \right) \leq 2n\exp\left(-\frac{\theta^2}{8tn^2G^2} \right). \end{aligned} \tag{34}$$

Here the second inequality uses the implication that at least one $a_i \in \mathbb{R}$ should satisfy $a_i \geq b/n$ when $\sum_{i=1}^{n} a_i \geq b$, and the last inequality is from the union bound of probabilities.

Also, from Markov's inequality and the bound (27) in Theorem 10 we get for any $\theta' > 0$ and $t \geq \hat{t}$ that

$$\mathbb{P}\left(\frac{1}{t} \sum_{j=1}^{t} \|w_j - w^*\| \geq \theta' \right) \leq \frac{\mu t^{-1/4}}{\theta'}. \tag{35}$$

Together with (33), the bounds in (34) and (35) imply that

$$\begin{aligned} \mathbb{P}\left(t\|\bar{g}_t - \nabla f(w^*)\| \geq \delta \right) &\leq \mathbb{P}\left(\left\| \sum_{j=1}^{t} z_i \right\| + L \sum_{j=1}^{t} \|w_j - w^*\| \geq \delta \right) \\ &\leq \mathbb{P}\left(\left\| \sum_{j=1}^{t} z_i \right\| \geq \frac{\delta}{2} \right) + \mathbb{P}\left(\frac{1}{t} \sum_{j=1}^{t} \|w_j - w^*\| \geq \frac{\delta}{2tL} \right) \\ &\leq 2n\exp\left(-\frac{\delta^2}{32tn^2G^2} \right) + \frac{2\mu L t^{3/4}}{\delta}. \end{aligned}$$

The first claim follows when we define $\delta := \varepsilon t$:

$$\mathbb{P}\left(\|\bar{g}_t - \nabla f(w^*)\| \geq \varepsilon \right) \leq 2n\exp\left(-\frac{\varepsilon^2 t}{32n^2G^2} \right) + \frac{2\mu L t^{-1/4}}{\varepsilon}.$$

The claim for the strongly convex case is proved similarly, using the bound (29) in Theorem 10 instead of (27) when we apply Markov inequality in (35). ■

The next result shows formally that for all t sufficiently large, the second term dominates in the bounds (31) and (32).

Corollary 12 *Suppose the assumptions of Theorem 11 hold and let μ and $\hat{t}$ be defined as in* (28). *Then for $\varepsilon \in (0, 4nG/\sqrt{e}]$ and $t \geq \max(\hat{t}, \bar{t})$, with*

$$\bar{t} := \left\lceil 16n^2G^2\varepsilon^{-2} \max\left(-W\left(\frac{-\varepsilon^2}{16n^2G^2}\right), -4\ln\left(\frac{\mu L}{n\varepsilon}\right)\right)\right\rceil, \tag{36}$$

(where $W(\cdot)$ denotes the branch of the Lambert function with $W < -1$), the bound (31) *simplifies to*

$$\mathbb{P}(\|\bar{g}_t - \nabla f(w^*)\| \geq \varepsilon) \leq 4\mu\varepsilon^{-1}Lt^{-1/4}. \tag{37}$$

When Ψ is strongly convex and the choice (21) *is used for $\{\beta_t\}$, and provided that $\varepsilon \in \left(0, 4nG/\sqrt{2/e}\right]$ and $t \geq \bar{t}'$ with*

$$\bar{t}' := \left\lceil 32n^2G^2\varepsilon^{-2} \max\left(-W\left(\frac{-\varepsilon^2}{32n^2G^2}\right), -2\ln\left(\frac{\mu' L\sqrt{6}}{n\varepsilon}\right)\right)\right\rceil, \tag{38}$$

the bound (32) *simplifies to*

$$\mathbb{P}(\|\bar{g}_t - \nabla f(w^*)\| \geq \varepsilon) \leq 4\mu'\varepsilon^{-1}L\left(\frac{6+\ln t}{t}\right)^{1/2}, \qquad t \geq 1. \tag{39}$$

Proof Note first that the ratio $(\ln t)/t$ is decreasing for $t \geq e$, so for t_1 defined by

$$\ln t_1 = \frac{\varepsilon^2}{16n^2G^2}t_1, \tag{40}$$

we have for $t \geq t_1$ and for sufficiently small ε ensuring $\varepsilon^2/(16n^2G^2) \leq 1/e$ (which is guaranteed by our assumption on ε) that

$$\frac{\ln t}{t} \leq \frac{\ln t_1}{t_1} \leq \frac{\varepsilon^2}{16n^2G^2}. \tag{41}$$

Note that (40) has the form $\ln t_1 = \alpha t_1$, for $\alpha = \varepsilon^2/(16n^2G^2)$, for which the solution is $t_1 = -W(-\alpha)/\alpha$. Thus, for any $t \geq \bar{t}$, the condition (41) holds. We continue to assume that $t \geq \bar{t}$, and note that

$$-\frac{\varepsilon^2 t}{32n^2G^2} + \frac{1}{4}\ln t \overset{(41)}{\leq} -\frac{\varepsilon^2 t}{32n^2G^2} + \frac{\varepsilon^2 t}{64n^2G^2} = -\frac{\varepsilon^2 t}{64n^2G^2} \overset{(36)}{\leq} \ln\left(\frac{\mu L}{n\varepsilon}\right).$$

By rearranging this expression and taking the exponential of both sides, we obtain

$$-\frac{\varepsilon^2 t}{32n^2G^2} \leq \ln\left(\frac{\mu L}{n\varepsilon}\right) - \frac{1}{4}\ln t \;\Leftrightarrow\; \exp\left(-\frac{\varepsilon^2 t}{32n^2G^2}\right) \leq \left(\frac{\mu L}{n\varepsilon}\right) t^{-1/4},$$

which implies that the first term on the right-hand side of the bound (31) is dominated by the second. We obtain the result (37) by substituting into (31).

The strongly convex case is similar. By solving $\ln t_2 = \alpha t_2$, for $\alpha = \varepsilon^2/(32n^2G^2)$, we have for $t \geq t_2$ and sufficiently small ε that

$$\frac{\ln t}{t} \leq \frac{\ln t_2}{t_2} \leq \frac{\varepsilon^2}{32n^2G^2}. \tag{42}$$

Thus for $t \geq \bar{t}'$, we have

$$-\frac{\varepsilon^2 t}{32n^2G^2} + \frac{1}{2}\ln t \overset{(42)}{\leq} -\frac{\varepsilon^2 t}{64n^2G^2} \overset{(38)}{\leq} \ln\left(\frac{\mu' L\sqrt{6}}{\varepsilon n}\right) \leq \ln\left(\frac{\mu' L\sqrt{6+\ln t}}{\varepsilon n}\right).$$

By rearranging the outermost expressions in this bound, and taking the exponents of both sides, we obtain

$$\exp\left(-\frac{\varepsilon^2 t}{32n^2G^2}\right) \leq \frac{\mu' L}{\varepsilon n}\left(\frac{6+\ln t}{t}\right)^{1/2},$$

from which the result (39) follows. ■

The max-terms in (36) and (38) grow only slowly with the dimension n. Hence, we can base our estimate of the required size of t on the factor in front of the max terms in (36) and (38), and on the right-hand sides of (37) and (39). In particular, for the general case, we need t to be at least a modest multiple of $(4nG/\varepsilon)^2$ (for $t \geq \bar{t}$) and also $t \geq (4\mu L/\varepsilon)^4$ (for the right-hand side of (37) to be useful).

4. Manifold Identification

In this section we show that most sufficiently advanced iterates of the RDA algorithm lie on the optimal manifold. Our analysis is based upon the properties of the dual average discussed in the previous section and on basic results for manifold identification. Specifically, we make use of a result of Hare and Lewis (2004), which states that when a sequence of points approaches a limit lying on an optimal nondegenerate manifold and the subgradients at these points approach zero, then all sufficiently advanced members of the sequence lie on the manifold. We identify subsequences of the RDA sequence that lie on the optimal manifold with increasing likelihood as the iteration counter grows. We further show that these subsequences form a dense subset of the full sequence. Separate but similar results are proved for the general convex case and the strongly convex case.

4.1 Convergent Sequences

We start with two results that estimate the likelihood of w_j lying within a given radius of w^*. The first of these results is for general convex objectives.

Lemma 13 (Convergent Sequences for General Convex Case) *Suppose that Assumptions 1 and 2 hold, that $w^* \in \mathcal{F}_D$, and that $\{\beta_j\}$ is chosen according to* (20)*. Define the subsequence $\mathcal{S}$ by*

$$\mathcal{S} := \Big\{ j \in \{1,2,\ldots\} \mid \mathbb{E}\left[I_{(\|w_j - w^*\| \leq \bar{r})}\|w_j - w^*\|^2\right] \leq j^{-1/4}, \text{ and} \\ \mathbb{E}\left[I_{(\|w_j - w^*\| > \bar{r})}\|w_j - w^*\|\right] \leq (1/\bar{r})j^{-1/4}\Big\}. \tag{43}$$

For any $\varepsilon > 0$, we then have

$$\mathbb{P}(\|w_j - w^*\| \geq \varepsilon) \leq \frac{1}{\varepsilon}\left(\frac{1}{\varepsilon} + \frac{1}{\bar{r}}\right) j^{-1/4}, \ \ \forall j \in \mathcal{S}. \tag{44}$$

Defining

$$\mathcal{S}_t := \mathcal{S} \cap \{1,2,\ldots,t\},$$

we have

$$\frac{1}{t}\mathrm{card}(\mathcal{S}_t) \geq 1 - \frac{2}{c}\left(\gamma D^2 + \frac{G^2}{\gamma}\right) t^{-1/4}, \tag{45}$$

that is, the density of $\mathcal{S}_t$ *in* $\{1,2,\ldots,t\}$ *is* $1 - O(t^{-1/4})$.

Proof To measure the cardinality of the complement of $\mathcal{S}_t$, that is, $\mathcal{S}_t^c := \{1,2,\ldots,t\} \setminus \mathcal{S}_t$, we first define indicator variables χ_-^j and χ_+^j for $j \geq 1$ as follows:

$$\chi_-^j := \begin{cases} 1 & \text{if } \mathbb{E}\left[I_{(\|w_j - w^*\| \leq \bar{r})} \|w_j - w^*\|^2\right] > j^{-1/4}, \\ 0 & \text{otherwise.} \end{cases}$$

$$\chi_+^j := \begin{cases} 1 & \text{if } \mathbb{E}\left[I_{(\|w_j - w^*\| > \bar{r})} \|w_j - w^*\|\right] > (1/\bar{r}) j^{-1/4}, \\ 0 & \text{otherwise.} \end{cases}$$

As the set $\mathcal{S}_t^c$ contains all indices $j \in \{1,2,\ldots,t\}$ that satisfy $\chi_-^j = 1$ or $\chi_+^j = 1$, the cardinality of $\mathcal{S}_t^c$ is bounded above by $\sum_{j=1}^t (\chi_-^j + \chi_+^j)$. For $\sum_{j=1}^t \chi_-^j$, we note that

$$\begin{aligned} \sum_{j=1}^t \mathbb{E}\left[I_{(\|w_j - w^*\| \leq \bar{r})} \|w_j - w^*\|^2\right] &\geq \sum_{j=1}^t \chi_-^j \mathbb{E}\left[I_{(\|w_j - w^*\| \leq \bar{r})} \|w_j - w^*\|^2\right] \\ &\geq \sum_{j=1}^t \chi_-^j j^{-1/4} \quad \text{(from the definition of } \chi_-^j\text{)} \\ &\geq t^{-1/4} \sum_{j=1}^t \chi_-^j. \end{aligned}$$

Using (24), we deduce that

$$\frac{1}{t}\sum_{j=1}^t \chi_-^j \leq \frac{1}{c}\left(\gamma D^2 + \frac{G^2}{\gamma}\right) t^{-1/4}.$$

Similar arguments for $\sum_{j=1}^t \chi_+^j$ with $\mathbb{E}\left[I_{(\|w_j - w^*\| > \bar{r})} \|w_j - w^*\|\right]$, $j = 1,2,\ldots,t$ and (25) lead to

$$\frac{1}{t}\sum_{j=1}^t \chi_+^j \leq \frac{1}{c}\left(\gamma D^2 + \frac{G^2}{\gamma}\right) t^{-1/4}.$$

Therefore, the fraction of the cardinality of $\mathcal{S}_t$ to $\{1,2,\ldots,t\}$ is

$$\begin{aligned} \frac{1}{t}\mathrm{card}(\mathcal{S}_t) &= 1 - \frac{1}{t}\mathrm{card}(\mathcal{S}_t^c) \\ &\geq 1 - \frac{1}{t}\sum_{j=1}^t (\chi_-^j + \chi_+^j) \\ &\geq 1 - \frac{2}{c}\left(\gamma D^2 + \frac{G^2}{\gamma}\right) t^{-1/4}, \end{aligned}$$

thus proving (45).

To show (44), we first observe that for any $\varepsilon > 0$,

$$\begin{aligned}\mathbb{P}(\|w_j - w^*\| \geq \varepsilon) = \mathbb{P}(\|w_j - w^*\| \geq \varepsilon,\ \|w_j - w^*\| \leq \bar{r}) \\ + \mathbb{P}(\|w_j - w^*\| \geq \varepsilon,\ \|w_j - w^*\| > \bar{r}).\end{aligned} \tag{46}$$

Focusing on the first term, we have for all $j \in \mathcal{S}$ that

$$\begin{aligned}\mathbb{P}(\|w_j - w^*\| \geq \varepsilon,\ \|w_j - w^*\| \leq \bar{r}) &= \mathbb{P}(I_{(\|w_j - w^*\| \leq \bar{r})}\|w_j - w^*\| \geq \varepsilon) \\ &\leq \varepsilon^{-2}\mathbb{E}\left[I_{(\|w_j - w^*\| \leq \bar{r})}\|w_j - w^*\|^2\right] \\ &\leq \varepsilon^{-2} j^{-1/4},\end{aligned} \tag{47}$$

where the first inequality is due to Markov and the second inequality is from the definition of $\mathcal{S}$ in (43). Similarly for the second term in (46), we have for all $j \in \mathcal{S}$ that

$$\begin{aligned}\mathbb{P}(\|w_j - w^*\| \geq \varepsilon,\ \|w_j - w^*\| > \bar{r}) &= \mathbb{P}(I_{(\|w_j - w^*\| > \bar{r})}\|w_j - w^*\| \geq \varepsilon) \\ &\leq \varepsilon^{-1}\mathbb{E}\left[I_{(\|w_j - w^*\| > \bar{r})}\|w_j - w^*\|\right] \\ &\leq \varepsilon^{-1}\bar{r}^{-1} j^{-1/4}.\end{aligned} \tag{48}$$

Applying (47) and (48) to (46) leads to the claim,

$$\mathbb{P}(\|w_j - w^*\| \geq \varepsilon) \leq \varepsilon^{-1}(\varepsilon^{-1} + \bar{r}^{-1}) j^{-1/4}, \quad \forall j \in \mathcal{S}.$$

■

This result implies that for sufficiently large j, the majority of iterates w_j converges to w^*, in probability. Similar results can be derived for the convergence of $\mathbb{E}[\phi(w_j)]$ to $\phi(w^*)$. The next theorem is the corresponding result for the strongly convex case.

Lemma 14 (Convergent Sequences for Strongly Convex Case) *Suppose that Assumptions 1 and 2 hold, that $w^* \in \mathcal{F}_D$, and that the regularizer Ψ is strongly convex with modulus $\sigma > 0$. Suppose that $\{\beta_j\}$ is defined by* (21)*. For any $\varepsilon > 0$, we have*

$$\mathbb{P}\left(\|w_j - w^*\| \geq \varepsilon\right) \leq \frac{G^2}{\varepsilon^2\sigma^2}\left(\frac{6 + \ln j}{j}\right), \qquad j \geq 1.$$

Proof From the proof of Xiao (2010, Corollary 4), we have

$$\mathbb{E}\left[\|w_j - w^*\|^2\right] \leq \frac{G^2}{\sigma^2}\left(\frac{6 + \ln j}{j}\right), \qquad j \geq 1.$$

The claim follows from the Markov inequality. ■

4.2 Identification

In this subsection, we state the main identification results. We start with a result from Hare and Lewis (2004), stating it in a modified form that is useful for our analysis below.

Theorem 15 *Suppose that ϕ is partly smooth at the minimizer w^* relative to the optimal manifold $\mathcal{M}$ and that the nondegeneracy condition* (13) *holds. Then there exists a threshold $\bar{\varepsilon} > 0$ such that for all $w \in \mathcal{O}$ with $\|w - w^*\| < \bar{\varepsilon}$ and $\operatorname{dist}(0, \partial\phi(w)) < \bar{\varepsilon}$, we have $w \in \mathcal{M}$.*

Proof Suppose for contradiction that no such $\bar{\varepsilon}$ exists. Let $\{\varepsilon_j\}_{j\geq 1}$ be any sequence of positive numbers such that $\varepsilon_j \downarrow 0$. Then for each $j \geq 1$ we have w_j such that $\|w_j - w^*\| < \varepsilon_j$, $\operatorname{dist}(0, \partial\phi(w_j)) < \varepsilon_j$ but $w_j \notin \mathcal{M}$. Considering the sequence $\{w_j\}_{j\geq 1}$, we have that $w_j \to w^*$, and $\operatorname{dist}(0, \partial\phi(w_j)) \to 0$. With convexity, these imply $\phi(w_j) \to \phi(w^*)$, since for all $a_j \in \partial\phi(w_j)$ we have $\phi(w_j) - \phi(w^*) \leq a_j^T(w_j - w^*) \leq \|a_j\|\|w_j - w^*\|$. (Because of our assumptions, we can choose a_j such that $\|a_j\| \leq \varepsilon_j$.) Convexity implies prox-regularity, so by applying Theorem 5.3 of Hare and Lewis (2004), we have that $w_j \in \mathcal{M}$ for all j sufficiently large. This contradicts our choice of w_j, so we conclude that $\bar{\varepsilon} > 0$ with the claimed properties exists. ■

The next theorem is our main result, showing that the RDA algorithm identifies the optimal manifold with increasing probability as iterations proceed. This result requires a condition (49) on h that is trivially satisfied by the usual prox-function $h(w) = \frac{1}{2}\|w - w_1\|^2$, with constant $\eta = 1$.

Theorem 16 (Identification for General Convex Case) *Suppose that Assumptions 1 and 2 hold, that $w^* \in \mathcal{F}_D$, that*

$$\sup_{b_j \in \partial h(w_j)} \|b_j\| \leq \eta \|w_j - w_1\|, \quad j = 1, 2, \ldots \tag{49}$$

for some $\eta > 0$, and that $\{\beta_j\}$ is defined as in (20)*. Given the set of indices $\mathcal{S}$ defined in* (43)*, we have*

$$\mathbb{P}(w_j \in \mathcal{M}) \geq 1 - (\zeta_1 + \zeta_2) j^{-1/4}$$

for all $j \in \mathcal{S}$ sufficiently large, where

$$\zeta_1 := \frac{3\max(1, L)}{\bar{\varepsilon}} \left(\frac{3\max(1, L)}{\bar{\varepsilon}} + \frac{1}{\bar{r}} \right), \text{ and } \zeta_2 := \frac{15\mu L}{\bar{\varepsilon}}.$$

Here $\bar{\varepsilon} > 0$ has the value defined in Theorem 15, L is the Lipschitz constant of (11)*, $\bar{r}$ is the radius of strong local minimization from* (15)*, and μ is defined in* (28)*.*

Proof We focus on the iterate w_j and the random events associated with it. First we denote the following event as E_1:

$$E_1: \quad \|w_j - w^*\| \leq \frac{\bar{\varepsilon}}{3\max(L, 1)}.$$

Note that E_1 depends on the history $\xi_{[j-1]}$ of random variables prior to iteration j. If E_1 is true, it trivially implies the condition $\|w_j - w^*\| \leq \bar{\varepsilon}$ of Theorem 15. From Lemma 13, with $\varepsilon = \frac{\bar{\varepsilon}}{3\max(L,1)}$, we have that

$$\mathbb{P}(\|w_j - w^*\| \leq \bar{\varepsilon}) \geq \mathbb{P}(E_1) \geq 1 - \zeta_1 j^{-1/4}, \quad \text{for all } j \in \mathcal{S}. \tag{50}$$

We now examine the other condition in Theorem 15, namely

$$\operatorname{dist}\big(0, \nabla f(w_j) + \partial\Psi(w_j)\big) \le \bar{\varepsilon}.$$

By adding and subtracting terms, we obtain

$$\begin{aligned}\nabla f(w_j) + a_j = (\nabla f(w_j) - \nabla f(w^*)) + (\nabla f(w^*) - \bar{g}_{j-1}) - \frac{\beta_{j-1}}{j-1} b_j \\ + \left(\bar{g}_{j-1} + a_j + \frac{\beta_{j-1}}{j-1} b_j\right). \end{aligned} \tag{51}$$

for any $a_j \in \partial\Psi(w_j)$ and $b_j \in \partial h(w_j)$. We choose the specific a_j and b_j that satisfy the optimality of the subproblem (18), that is,

$$0 = \bar{g}_{j-1} + a_j + \frac{\beta_{j-1}}{j-1} b_j.$$

This choice eliminates the last term in (51). For the other three terms, we have the following observations.

(i) For those w_j satisfying E_1, the Lipschitz property of ∇f (Lemma 2) implies that

$$\|\nabla f(w_j) - \nabla f(w^*)\| \le L\|w_j - w^*\| \le \frac{\bar{\varepsilon} L}{3\max(L,1)} \le \frac{\bar{\varepsilon}}{3}.$$

Hence, E_1 implies the following event:

$$E_2: \quad \|\nabla f(w_j) - \nabla f(w^*)\| \le \bar{\varepsilon}/3.$$

(ii) From Corollary 12, we have by setting $\varepsilon = \bar{\varepsilon}/3$ and $t = j-1$ that

$$\mathbb{P}(\|\nabla f(w^*) - \bar{g}_{j-1}\| \ge \bar{\varepsilon}/3) \le 4\mu L \left(\frac{3}{\bar{\varepsilon}}\right)(j-1)^{-1/4} < \zeta_2 j^{-1/4},$$

for $j - 1 \ge \max(\hat{t}, \bar{t})$, where $\hat{t}$ is defined in (28) and $\bar{t}$, which depends on $\bar{\varepsilon}$, is defined in (36). Hence, denoting by E_3 the event

$$E_3: \quad \|\nabla f(w^*) - \bar{g}_{j-1}\| \le \bar{\varepsilon}/3,$$

we have that

$$\mathbb{P}(\neg E_3) < \zeta_2 j^{-1/4}, \qquad j \ge \max(\hat{t}, \bar{t}) + 1. \tag{52}$$

(iii) Since $\beta_{j-1} = \gamma(j-1)^{1/2}$, we have for w_j satisfying E_1 that

$$\begin{aligned}\frac{\beta_{j-1}}{j-1}\|b_j\| &= \gamma(j-1)^{-1/2}\|b_j\| \\ &\le \gamma\eta(j-1)^{-1/2}\|w_j - w_1\| && \text{from (49)} \\ &\le \gamma\eta(j-1)^{-1/2}(\|w_j - w^*\| + \|w_1 - w^*\|) \\ &\le \gamma\eta(j-1)^{-1/2}\left(\frac{\bar{\varepsilon}}{3\max(L,1)} + \sqrt{2}D\right) && \text{from } w^* \in \mathcal{F}_D \text{ and (17)}.\end{aligned}$$

Therefore, E_1 implies the event

$$E_4: \quad \frac{\beta_{j-1}}{j-1}\|b_j\| \leq \frac{\bar{\varepsilon}}{3}, \qquad j \geq t_0+1,$$

where we define t_0 by

$$t_0 := \left\lceil \frac{9\gamma^2\eta^2}{\bar{\varepsilon}^2}\left(\frac{\bar{\varepsilon}}{3\max(L,1)}+\sqrt{2}D\right)^2 \right\rceil.$$

Therefore for $j \in \mathcal{S}$ with $j \geq \max\{\hat{t},\bar{t},t_0\}+1$, by definition of the events E_1, E_2, E_3, and E_4 above, the probability that the conditions of Theorem 15 hold is bounded as follows:

$$\begin{aligned}
&\mathbb{P}\Big(\|w_j - w^*\| \leq \bar{\varepsilon} \wedge \operatorname{dist}(0,\partial\phi(w_j)) \leq \bar{\varepsilon}\Big) \\
&\geq \quad \mathbb{P}\Big(E_1 \wedge E_2 \wedge E_3 \wedge E_4\Big) = \mathbb{P}(E_1 \wedge E_3) \\
&\geq \quad 1 - \mathbb{P}(\neg E_1) - \mathbb{P}(\neg E_3) \geq 1 - (\zeta_1+\zeta_2) j^{-1/4},
\end{aligned}$$

where the last inequality is due to (50) and (52). Our claim follows. ■

Theorem 17 (Identification for Strongly Convex Case) *Suppose that Assumptions 1 and 2 hold, that Ψ is strongly convex with modulus $\sigma > 0$, that $w^* \in \mathcal{F}_D$, that $h(\cdot)$ satisfies* (49)*, and that $\beta_j = 0$ for all $j \geq 1$, as defined in* (21)*. Then we have*

$$\mathbb{P}(w_j \in \mathcal{M}) \geq 1 - (\zeta_1' + \zeta_2')\left(\frac{6+\ln j}{j}\right)^{1/2}.$$

for all j sufficiently large, where

$$\zeta_1' := \frac{G^2}{\sigma^2}\left(\frac{3\max(1,L)}{\bar{\varepsilon}}\right)^2, \text{ and } \zeta_2' := \frac{17\mu' L}{\bar{\varepsilon}}.$$

Here $\bar{\varepsilon} > 0$ has the value defined in Theorem 15, L is the Lipschitz constant of (11)*, G is the uniform bound on gradient norms in* (12)*, and μ' is defined in* (30)*.*

Proof This proof is almost identical to that of Theorem 16; here we briefly mention the required changes for the strongly convex case. Consider $\bar{\varepsilon} > 0$ and the event E_1 defined in the proof of Theorem 16. From Lemma 14 with $\varepsilon = \frac{\bar{\varepsilon}}{3\max(L,1)}$, we have

$$\mathbb{P}(\|w_j - w^*\| \leq \bar{\varepsilon}) \geq \mathbb{P}(E_1) \geq 1 - \zeta_1'(6+\ln j)/j, \qquad j \geq 1.$$

Instead of (ii) and (iii) in the proof of Theorem 16, we use the following:

(ii') From Corollary 12, we have by setting $\varepsilon = \bar{\varepsilon}/3$ and $t = j-1$ that

$$\mathbb{P}(\|\nabla f(w^*) - \bar{g}_{j-1}\| \geq \bar{\varepsilon}/3) \leq 4\mu'\frac{3}{\bar{\varepsilon}}L\left(\frac{6+\ln(j-1)}{j-1}\right)^{1/2} < \frac{17\mu' L}{\bar{\varepsilon}}\left(\frac{6+\ln j}{j}\right)^{1/2},$$

for all $j > \max(\bar{t}'+1, 2)$, where $\bar{t}'$ is defined in (38). Hence, denoting by E_3 the event $\|\nabla f(w^*) - \bar{g}_{j-1}\| \leq \bar{\varepsilon}/3$, we have that

$$\mathbb{P}(\neg E_3) < \zeta_2' \left(\frac{6+\ln j}{j}\right)^{1/2},$$

for all j sufficiently large.

(iii') With $\beta_{j-1} = 0$ and the given conditions, the event E_4 holds for all $j \geq 2$.

Using the modified probability bounds for E_1 and E_3, we have

$$\begin{aligned} P\Big(\|w_j - w^*\| \leq \bar{\varepsilon} \wedge \operatorname{dist}(0, \partial\phi(w_j)) \leq \bar{\varepsilon}\Big) &\geq P\Big(E_1 \wedge E_3\Big) \\ &\geq 1 - \zeta_1' \left(\frac{6+\ln j}{j}\right) - \zeta_2' \left(\frac{6+\ln j}{j}\right)^{1/2} \\ &\geq 1 - (\zeta_1' + \zeta_2') \left(\frac{6+\ln j}{j}\right)^{1/2}, \end{aligned}$$

for all $j \geq \max(\bar{t}'+1, 9)$, using the fact that $(6+\ln j)/j \leq 1$ for $j \geq 9$. Our claim follows. ■

Lemma 13 tells us that the sequence $\mathcal{S}$ is "dense" in $\{1, 2, \ldots\}$, while Theorem 16 states that for all sufficiently large $j \in \mathcal{S}$, w_j lies on the optimal manifold with probability approaching one as j increases. When the regularizer Ψ is strongly convex, Theorem 17 tells that similar results hold earlier in the sequence $\{w_j\}$.

We conclude this section with a simple example to show that algorithms based on subproblem (7) that were discussed in Section 1.3 do not have reliable identification properties. The major reason is that each iteration uses a "raw" sampled gradient g_t of f, rather than the averaged (and thus smoothed) approximate gradient $\bar{g}_t$ of RDA. Thus, as we see now, even when the current iterate w_t is optimal, the subproblem may step away from this point, and away from the optimal manifold.

Example 1 Consider the following definitions for the problem (1):

- $n = 1$ (a scalar problem)
- $\Xi = [-2, 2]$ with ξ uniformly distributed on this interval.
- $F(w;\xi) = \xi w$. Thus $\nabla F(w;\xi) = \xi$ and $f(w) \equiv 0$.
- $\Psi(w) = |w|$.

With these definitions, the solution is $w^* = 0$ and the optimal manifold is zero-dimensional: $\mathcal{M} = \{0\}$. Thus Assumption 2 is trivially satisfied. Regarding Assumption 1, the nondegeneracy condition is satisfied, since $\partial\Psi(0) = [-1, 1]$ while $\nabla f(0) = 0$. It is easy to verify too that F satisfies the unbiasedness, uniform Lipschitz, and uniform boundedness properties of Assumption 1.

Setting $w_t = w^* = 0$, the subproblem (7) is

$$w_{t+1} = \arg\min_w \xi_t w + |w| + \frac{1}{2\alpha_t} w^2,$$

where ξ_t is selected uniformly at random from $[-2,2]$. For $\xi_t \in [-1,1]$, we have $w_{t+1} = 0$, so the next iterate remains at the optimal point. However, if $\xi_t \in [-2,-1)$, we have $w_{t+1} = -\alpha_t(\xi_t + 1) > 0$, while if $\xi_t \in (1,2]$, we have $w_{t+1} = -\alpha_t(\xi_t - 1) < 0$. In both cases, the next iterate steps away from the solution $w^* = 0$ and from the optimal manifold. Because ξ_t is uniformly distributed in $[-2,2]$, this event happens with probability $1/2$ in this example. (The probability of this behavior can be made arbitrarily close to 1 by suitable extension of the interval Ξ.)

5. Enhancing the Regularized Dual Averaging Algorithm

Here we present a simple optimization strategy motivated by our analysis above, in which the RDA method gives way to a local phase after a near-optimal manifold is identified.

Algorithm 2 summarizes our algorithm RDA$^+$. This algorithm starts with RDA steps until it identifies a near-optimal manifold, then searches this manifold using a different optimization strategy, possibly better suited to lower-dimensional spaces and possibly with better local convergence properties. If an explicit parametrization of $\mathcal{M}$ is available, the "local phase" could take the form of a Newton-like method applied to the composition of the objective with this parametrization. In the important special case of $\Psi(\cdot) = \lambda\|\cdot\|_1$, $\mathcal{M}$ can be represented simply by its nonzero variables, and LPS (Shi et al., 2008; Wright, 2012) can be applied on the space of just these variables.

To decide when to make the switch to the local phase, we use a simple heuristic inspired by Theorem 16 and 17 that if the past τ consecutive iterates have been on the same manifold $\mathcal{M}$, we take $\mathcal{M}$ to be approximately optimal. However, we "safeguard" by expanding $\mathcal{M}$ to incorporate additional dimensions that may yet contain the minimizer. This simple approach will work provided that the $\mathcal{M}$ so constructed is a *superset* of the optimal manifold, since our implementation of the local phase is able to move to more restricted submanifolds of $\mathcal{M}$ but does not expand its search to include dimensions not originally included in the manifold. When sufficient progress is not attained in the local phase, we can resume the paused dual-averaging phase.

We describe the details of Algorithm 2 for ℓ_1 regularization, where $\Psi(w) = \lambda\|w\|_1$ for some $\lambda > 0$. (Thus, the starting point will be $w_1 = 0$.) The optimal manifold corresponds (near w^*) to the points in $\mathbb{R}^n$ that have the same nonzero patterns as w^*. We use the simple quadratic prox-function $h(w) = \frac{1}{2}\|w - w_1\|^2$. Since Ψ is not strongly convex, we use the sequence $\{\beta_t\}$ defined in (20).

We now describe various specifics of the implementation of Algorithm 2 for this case.

5.1 Computation of w_{j+1}

The closed-form solution for the subproblem (18) with $t = j$ is

$$[w_{j+1}]_i = \frac{\sqrt{j}}{\gamma}\text{soft}(-[\bar{g}_j]_i, \lambda), \;\; i = 1,2,\ldots,n,$$

where $\text{soft}(u,a) := \text{sgn}(u)\max\{|u| - a, 0\}$ is the well-known soft-threshold function.

5.2 Surrogate Objective

To apply the batch optimization method LPS in the local phase of Algorithm 2, we use an empirical estimate $\tilde{\phi}_{\mathcal{N}}$ in (3) as a surrogate objective function (where ξ_j, $j \in \mathcal{N}$ is a sample of random variables from the space Ξ according to the distribution P), and then solve

$$\min_{w \in \mathcal{M}} \;\; \tilde{\phi}_{\mathcal{N}}(w) = \tilde{f}_{\mathcal{N}}(w) + \lambda\|w\|_1. \tag{53}$$

Algorithm 2: RDA$^+$ Algorithm.

Input:

- a prox-function $h(w)$ that is strongly convex on $\operatorname{dom}\Psi$ and also satisfies

$$\arg\min_{w\in\mathbb{R}^n} h(w) \in \arg\min_{w\in\mathbb{R}^n} \Psi(w),$$
$$\sup_{b\in\partial h(w)} \|b\| \le \eta\|w-w_1\|, \ \forall w \in \operatorname{dom}\Psi, \quad \text{where } w_1 \in \arg\min_w \Psi(w).$$

- a nonnegative and nondecreasing sequence $\{\beta_j\}$, $j \ge 1$.

- a positive integer τ.

Initialize: Set $\bar{g}_0 = 0$;
for $j = 1,2,\ldots$ **do** (Dual Averaging)
  Choose a random vector $\xi_j \in \Xi$;
  Compute a gradient $g_j = \nabla F(w_j;\xi_j)$;
  Update the average gradient:

$$\bar{g}_j = \frac{j-1}{j}\bar{g}_{j-1} + \frac{1}{j}g_j.$$

  Compute the next iterate by solving the subproblem (18), which is

$$w_{j+1} = \arg\min_{w\in\mathbb{R}^n} \left\{ \langle \bar{g}_j, w\rangle + \Psi(w) + \frac{\beta_j}{j}h(w) \right\}.$$

  if *there is* $\mathcal{M}$ *such that* $w_{j+2-i} \in \mathcal{M}$ *for* $i = 1,2,\ldots,\tau$ **then** (Local Phase)
    Safeguard $\mathcal{M}$ by adding dimensionality as appropriate to encompass w^*;
    Use a technique for low-dimensional optimization to search for a solution on manifold $\mathcal{M}$, starting at w_{j+1};
  end
end

LPS calculates first- and second-order information for $\tilde{\phi}_{\mathcal{N}}$ on the subset of components defined by $\mathcal{M}$. Since the intrinsic dimension of $\mathcal{M}$ is usually much smaller than the dimension n of the full space, these restricted gradients and Hessians are much cheaper to compute than their full-space counterparts.

5.3 Local Phase: LPS

We give further information on the LPS approach, following Wright (2012) but specializing the description to the problem (53). Many details are omitted; we refer the reader to Wright (2012) for a complete description and analysis of the approach, and to Shi et al. (2008) for an earlier version together with an application to ℓ_1-regularized logistic regression.

Algorithm 3: LPS Approach in Local Phase

Input: $\nu_{\max} > \nu_{\min} > 0$, $K > 1$, $s > 1$, w_{j+1};
Initialize: $z_0 \leftarrow w_{j+1}$;
for $k = 0, 1, 2, \ldots$ **do**
  Choose $\mathcal{M}_k \subset \mathcal{M}$ such that each nonzero component in $\mathcal{M}$ appears in at least one of the manifolds $\mathcal{M}_k, \mathcal{M}_{k-1}, \ldots, \mathcal{M}_{k-K+1}$, for $k \geq K$;
  Choose $\nu_k \in [\nu_{\min}, \nu_{\max}]$;
  Solve (54) for d_k;
  while $\tilde{\phi}_{\mathcal{N}}(z_k + d_k) > \tilde{\phi}_{\mathcal{N}}(z_k) - |d_k|^3$ **do**
    Set $\nu_k \leftarrow s\nu_k$;
    Solve (54) for d_k;
  end
RN Find $\tilde{d}_k$ with $\tilde{\phi}_{\mathcal{N}}(z_k + \tilde{d}_k) \leq \tilde{\phi}_{\mathcal{N}}(z_k + d_k)$ and $\tilde{\phi}_{\mathcal{N}}(z_k + \tilde{d}_k) \leq \tilde{\phi}_{\mathcal{N}}(z_k) - .01|\tilde{d}_k|^3$;
  Set $z_{k+1} \leftarrow z_k + \tilde{d}_k$;
end

The scheme is outlined as Algorithm 3. We use $\mathcal{M}_k$ to denote a subset of the restricted manifold $\mathcal{M}$, again defined by the indices of the components in which we consider a move at iteration k. We require that each nonzero component in $\mathcal{M}$ be considered for a possible step at least once every K iterations, where K is a defined parameter. The basic step at each iteration, given an LPS iterate z_k, is obtained by solving the following subproblem:

$$\min_{d \in \mathcal{M}_k} \nabla \tilde{f}_{\mathcal{N}}(z_k)^T d + \frac{\nu_k}{2} d^T d + \lambda \|z_k + d\|_1, \tag{54}$$

where ν_k is manipulated by the algorithm to produce a decrease in the objective at each iteration. Formulation of this subproblem requires evaluation only of those elements of the gradient $\nabla \tilde{f}_{\mathcal{N}}$ corresponding to the nonzero set $\mathcal{M}_k$. Since it is separable in the components of d, it can be solved in closed form in a number of operations proportional to the number of nonzero components in $\mathcal{M}_k$.

The enhancement in the line marked as RN of Algorithm 3 can be carried out by a reduced Newton-type method, applied to the current set of nonzero components of $z_k + d_k$. Here, we obtain an estimate of the restriction of the Hessian $\nabla^2 \tilde{f}_{\mathcal{N}}$ to the nonzero set, possibly by taking a random sub-batch of $\mathcal{N}$.

5.4 Checking Optimality

From the optimality condition for (3), we define the optimality measure $\delta(w_j)$ as follows,

$$\delta(w_j) := \frac{1}{\sqrt{n}} \inf_{a_j \in \partial \|w_j\|_1} \|\nabla \tilde{f}_{\mathcal{N}}(w_j) + \lambda a_j\|. \tag{55}$$

Since $\delta(w^*) \approx 0$ for a sufficiently large sample set $\mathcal{N}$ because of the law of large numbers, it makes sense to terminate when $\delta(w_j)$ drops below a certain threshold.

5.5 Safeguarding

At the start of the local phase, we augment $\mathcal{M}$ by adding components i for which $[w_{j+1}]_i = 0$ but $[\bar{g}_j]_i$ is close to one of the endpoints of its allowable range; that is,

$$[w_{j+1}]_i = 0 \text{ and } |[\bar{g}_j]_i| > \rho\lambda \tag{56}$$

for some ρ between 0 and 1 (but closer to 1). This conservative strategy allows for the possibility that $|[\bar{g}_j]_i|$ will exceed λ on a later iteration, causing $[w_{j+1}]_i$ to move away from zero.

6. Computational Experiments

We report here on computational experiments involving binary classification tasks via ℓ_1-regularized logistic regression. Given a set of m training examples, we select one at time t—indexed by ξ_t—and use its feature vector $x_{\xi_t} \in \mathbb{R}^{n-1}$ and label $y_{\xi_t} \in \{-1,1\}$ to define the corresponding loss function for $\tilde{w} \in \mathbb{R}^{n-1}$, $b \in \mathbb{R}$ and $w = (\tilde{w}, b)$:

$$F(w;\xi_t) = \ln\left(1 + \exp\left(-y_{\xi_t}(\tilde{w}^T x_{\xi_t} + b)\right)\right).$$

We choose $\Psi(w) = \lambda\|\tilde{w}\|_1$ as the regularizer for some $\lambda > 0$, and set $w_1 = 0$, as required in Algorithm 2.

For the second phase of Algorithm 2, we set $\mathcal{N} = \{1,2,\ldots,m\}$ to obtain the empirical estimate $\tilde{\phi}_{\mathcal{N}}$ from the full training set.

6.1 Manifold Identification

To investigate the identification behavior of the RDA algorithm in practical circumstances, we use five data sets from the UCI Machine Learning Repository, which have the sizes / dimensions shown in Table 1. We apply the original LPS to acquire the reference solution $w^*_{\mathcal{N}}$ of (3), with the tight optimality threshold of 10^{-6}. We then tabulate how many iterations of RDA are required before it generates a point in the optimal manifold $\mathcal{M}$ containing $w^*_{\mathcal{N}}$. We also check when the iterates of RDA reach a modest superset of the optimal manifold—a "2×" superset composed of the points in $\mathbb{R}^n$ having the same sign pattern for the active components in $\mathcal{M}$, and up to twice as many nonzeros as the points in $\mathcal{M}$. For each data set we use three values of λ equally spaced in the log-scale range of $[0.3, 0.9]\lambda_{\max}$, where $\lambda_{\max}$, computed accordingly to Koh et al. (2007), is the value beyond which the solution $w^*_{\mathcal{N}}$ has all zero components, except for the intercept term.

Table 1 shows performance of the RDA algorithm, over 100 repeated runs for each data set (using random permutations of training data for each run and for each sweep through the data), as measured by the number of iterations required for the algorithm to identify the optimal manifold and its 2× superset. Since the empirical distributions of the iteration counts are skewed, we show the median (rather than the mean) and the standard deviation. The table also shows the values of the optimality measure δ defined in (55) for the iterate at the moment we identify the optimal manifold. These results demonstrate that a huge number of iterations may be required to identify the optimal manifold, whereas identifying the superset is often much easier. In cases in which only a few components of $w^*_{\mathcal{N}}$ are nonzero, just a small fraction of the training examples usually suffice to identify the 2× superset. We note too that even when optimal identification is achieved, the iterate is still far from being optimal, by the criterion (55), suggesting the need for a local phase to achieve tighter optimality.

Data set	λ	$2\times$ Superset		Optimal $\mathcal{M}$		Optimality δ	NNZs $w^*_{\mathcal{N}}$
Glass ($m=214, n=10$)	0.29	14	(25)	20	(27)	0.068	1
	0.17	13	(10)	116	(428)	0.063	2
	0.10	13	(11)	28392	(6907)	0.016	3
Iono ($m=351, n=35$)	0.22	38	(84)	122	(95)	0.015	2
	0.13	44	(28)	30812	(15575)	0.008	3
	0.07	86	(41)	404	(150)	0.019	5
Arrhythmia ($m=452, n=280$)	0.15	192	(110)	304	(141)	0.001	2
	0.09	272	(88)	2036	(1076)	0.002	8
	0.05	447	(195)	27750	(4590)	0.001	13
Spambase ($m=4601, n=58$)	0.17	137	(219)	357	(325)	0.006	1
	0.10	722	(2495)	4340	(3097)	0.004	8
	0.06	812	(1247)	4680	(2209)	0.004	17
Pageblock ($m=5473, n=11$)	0.11	26	(326)	58	(395)	0.063	1
	0.07	182	(941)	524	(1233)	0.038	3
	0.04	103	(913)	461	(1232)	0.040	4

Table 1: Manifold identification properties of the RDA algorithm over 100 runs for each data set. The median number of iterations required to identify the optimal manifold $\mathcal{M}$, and the number required to identify a $2\times$ superset, are presented, along with the standard deviations over the 100 tests (in parentheses). δ represents the optimality measure at the moment of identifying $\mathcal{M}$, while the last column shows the number of nonzeros (excluding intercepts) in the reference solution obtained by LPS.

6.2 Performance on the MNIST Data Set

We now focus on the effects of the local phase on the performance of RDA$^+$. For this purpose, we use the MNIST data set which consists of gray-scale images of digits represented by 28×28 pixels. We choose the binary classification problem of distinguishing between the digits 6 and 7, for which the data set has 12183 training and 1986 test examples. Although the "6 vs 7" task is relatively easy, we choose this setting so that we can compare our results to those reported by Xiao (2010) for the original RDA algorithm.

We compare RDA$^+$ to several other algorithms: SGD, TG, RDA, and LPS. The SGD method (see, for instance, Nemirovski et al., 2009) for ℓ_1 regularization consists of the iterations

$$[w_{t+1}]_i = [w_t]_i - \alpha_t \left([g_t]_i + \lambda \mathrm{sgn}\left([w_t]_i\right) \right), \; i = 1, 2, \ldots, n,$$

where g_t is a sampled approximation to the gradient of f at w_t, obtained from a single training example. The TG method (Langford et al., 2009) truncates the iterates obtained by the standard SGD at every Kth step (where K is a user-defined constant). That is,

$$[w_{t+1}]_i = \begin{cases} \mathrm{trnc}\left([w_t]_i - \alpha_t [g_t]_i, \lambda_t^{\mathrm{TG}}, \theta\right) & \text{if } \mathrm{mod}(t,K) = 0, \\ [w_t]_i - \alpha_t [g_t]_i & \text{otherwise,} \end{cases}$$

where $\lambda_t^{\mathrm{TG}} := \alpha_t \lambda K$, $\mathrm{mod}(t,K)$ is the remainder on division of t by K, θ is a user-defined constant, and

$$\mathrm{trnc}(\omega, \lambda_t^{\mathrm{TG}}, \theta) = \begin{cases} 0 & \text{if } |\omega| \leq \lambda_t^{\mathrm{TG}}, \\ \omega - \lambda_t^{\mathrm{TG}} \mathrm{sgn}(\omega) & \text{if } \lambda_t^{\mathrm{TG}} < |\omega| \leq \theta, \\ \omega & \text{otherwise.} \end{cases}$$

We follow Xiao (2010) in using $\theta = \infty$ and $K = 10$.

For the stepsize α_t in SGD and TG, we adopt a variable stepsize scheme (Zinkevich, 2003; Nemirovski et al., 2009), choosing α_t to be a multiple of $1/\sqrt{t}$ so that the methods can achieve regret bounds of $O(\sqrt{t})$ similar to that of RDA.

In our implementations of RDA$^+$ and RDA, we set $\gamma = 5000$ in (20). (This value is determined by cross validation with RDA, using a single scan through the data set.) For LPS and the local phase of RDA$^+$, we compute a reduced Newton step on the current active manifold only when the number of nonzeros falls below 200. We also use the full set of training examples to compute the reduced gradient and reduced Hessian of the surrogate function $f_{\mathcal{N}}$.

For SGD, TG, and RDA, we keep track not only of the primal iteration sequence $\{w_t\}$, but also the primal averages $\bar{w}_T := \frac{1}{T}\sum_{t=1}^{T} w_t$, where T for each algorithm denotes the iteration number where the algorithm is stopped. We include these in the comparison because the convergence of the stochastic subgradient algorithms is often described in terms of the primal averages. Note that RDA$^+$ and LPS do not make use of primal averages.

We first run the RDA$^+$ algorithm with random permutations of the training samples, stopping when $\tau = 100$ consecutive iterates have the same sparsity pattern, after seeing all samples at least once. (All repeated runs required at most 19327 iterations to stop, which is less than two complete sweeps through the data set.) In the safeguarding test (56), we use $\rho = 0.85$. We run the local phase of RDA$^+$ until the optimality measure in (55) falls below 10^{-4}. We record the total runtime of the RDA$^+$ algorithm, then run other algorithms SGD, TG, RDA, and LPS up to the runtime of RDA$^+$, stopping them earlier if they achieve the desired optimality before that point.

6.2.1 PROGRESS IN TIME

We compare the convergence of the algorithms in terms of the optimality measure and the number of nonzero components. Figure 1 presents the plots for the iterates without averaging, for three different values of λ: 10, 1, and 0.1. The optimality plots (on the left) show that RDA$^+$ achieves the target optimality much faster than other algorithms, including LPS. RDA behaves better than SGD and TG, but still does not come close to the target value of optimality. There is only a modest decrease in the optimality measure for SGD, TG, and RDA over the time frame of this experiment.

The plots on the right of Figure 1 show the number of nonzeros (excluding intercepts hereafter) in the iterates. RDA tends to produce much sparser iterates with less fluctuation than SGD and TG, but it fails to reduce the number of nonzeros to the smallest number identified by RDA$^+$ in the given time, for the values $\lambda = 1.0$ and $\lambda = 0.1$.

In its local phase, RDA$^+$ behaves very similarly to LPS, sharing the typical behavior of non-monotonic decrease in the optimality measure (55). However, the local phase often converges faster than LPS, because it starts with the reduced space chosen by the dual-averaging phase of RDA$^+$. The number of nonzeros often increases at the point of switching between phases, as the safeguarding strategy adds more elements to the nonzero set.

Figure 2 shows similar plots but for the primal averaged iterates $\bar{w}_t$. We duplicate the plots of RDA$^+$ and LPS (which do not use iterate averaging) from Figure 1 for easy comparison. The number of nonzero components is clearly higher for averaged iterates.

6.2.2 QUALITY OF SOLUTIONS

In Figure 3, we compare the quality of the solutions in terms of optimality, the number of nonzeros, and test error rate. We present the results for the iterates without averaging in the three plots on the left, and those for the primal-averaged iterates (for algorithms SGD, TG, and RDA) in the plots on the right. (The plots of RDA$^+$ and LPS on the left are duplicated in the right-hand plots for easy comparison.) We run the algorithms with the same setting used in the previous experiments, except for LPS, which we run to optimality (10^{-4}, without time limit) to provide a baseline for comparison. (The runtime of LPS was about four times longer than that of RDA$^+$ on average.) The experiments are repeated for 100 runs of the data (using random permutations of training data for each run and for each sweep through the data), for each of seven λ values in the interval $[.01, 10]$. (The value of $\lambda_{\max}$ for this data set is 45.8.)

In Figure 3, only the solutions from RDA$^+$ achieve the desired optimality and the smallest number of nonzeros, with almost identical quality to the solutions from LPS. The solutions (both with and without averaging) from SGD, TG, and RDA are suboptimal, leaving much scope for zeroing out many more components of the iterates. RDA achieves a similar number of nonzeros to RDA$^+$ for large λ values, but more nonzeros for smaller values of λ. In terms of the test error rate, RDA$^+$ produces slightly better solutions than SGD, TG, and RDA overall. Although the improvement is marginal, we note that high accuracy is difficult to achieve solely with the stochastic online learning algorithms in limited time. The averaged iterates of SGD and TG show smaller test error for $\lambda \geq 1$ than others, but they require a large number of nonzero components, despite the strong regularization imposed.

In Figure 4, we show typical solutions obtained from the various algorithms for different values of λ. The first three rows present the solutions acquired without averaging, and the last three rows present those obtained with primal averaging. The solutions from RDA$^+$ reveal almost identical sparsity pattern to those from the baseline algorithm LPS, achieving smallest nonzero patterns. RDA produces solutions of similar sparsity to RDA$^+$ for large λ values, but much denser solutions for smaller λ values. Again, we see that primal-averaged solutions are denser than those without averaging.

7. Conclusion

We have shown that under assumptions of nondegeneracy and strong local minimization at the optimum, the (non-averaged) iterates generated by the RDA algorithm identify the optimal manifold with probability approaching one as iterations proceed. This observation enables us to develop a new algorithmic framework that enjoys the low computational footprint of stochastic gradient methods as well as the rapid local convergence of Newton-type optimization techniques, which can be used to search for solutions on near-optimal manifolds that often have much lower intrinsic dimension than the full space.

We believe that our analysis can be extended in several directions. First, the use of ℓ_p norms in definition of the prox-function h can lead to faster convergence of RDA, but the interaction with the strong minimizer assumption and the manifold identification results that we use here is

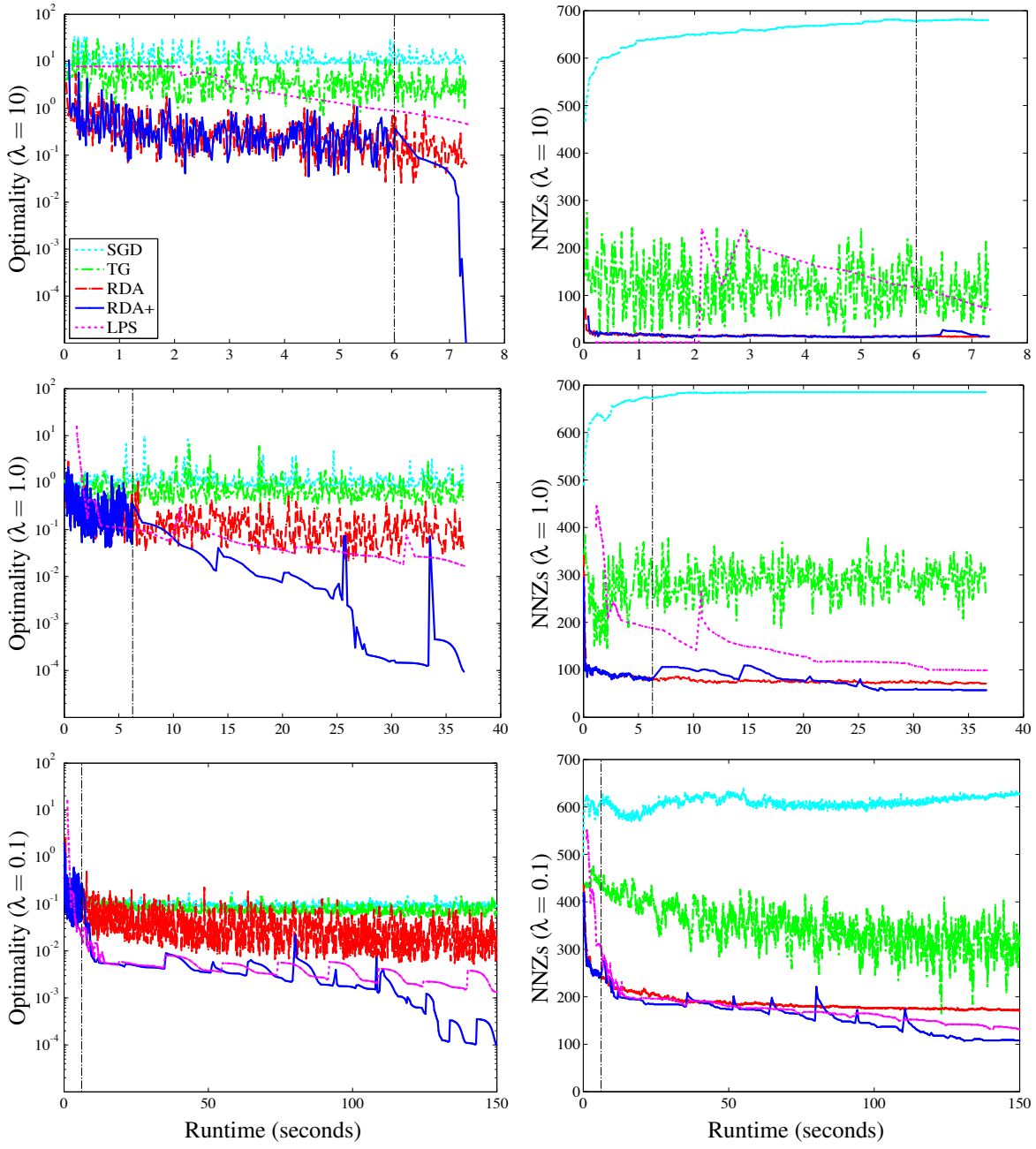


Figure 1: Convergence of iterates for various algorithms applied to an ℓ_1-regularized logistic regression function constructed from the digits 6 and 7 in the MNIST data set. Convergence is measured in terms of the optimality measure (left) and the number of nonzero components (excluding intercepts) in the iterates (right). SGD, TG, RDA, and LPS are run up to the time taken for RDA$^+$ to achieve 10^{-4} optimality value. The vertical lines indicate the event of phase switching in RDA$^+$. The vertical axes on the left are in logarithmic scale.

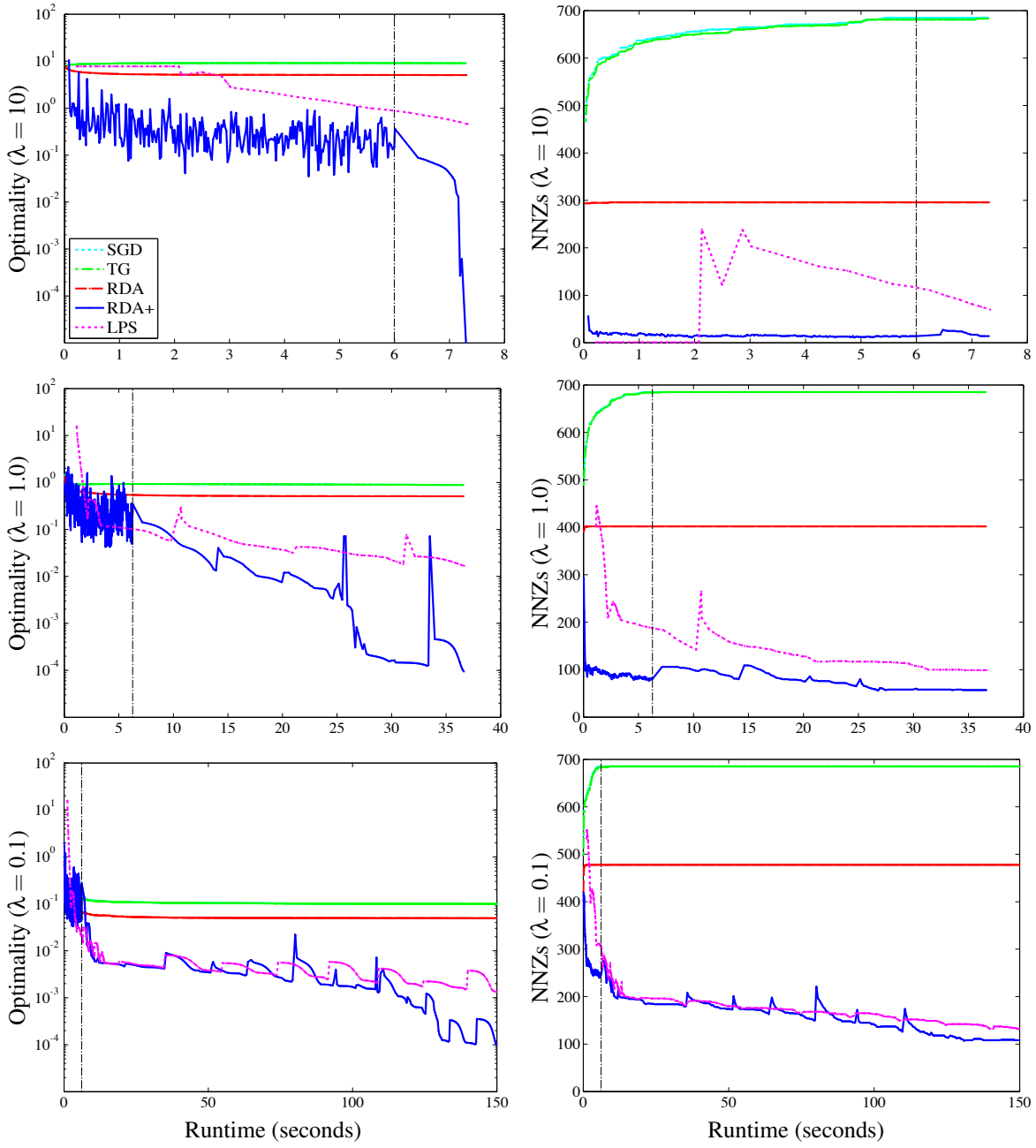


Figure 2: Convergence of averaged iterates (for SGD, TG, and RDA) and original iterates (for RDA$^+$ and LPS) on the ℓ_1-regularized logistic regression function constructed from the digits 6 and 7 in the MNIST data set. Convergence is measured in terms of the optimality measure (left) and the number of nonzero components (excluding intercepts) in the iterates (right). SGD, TG, RDA, and LPS are run up to the time taken for RDA$^+$ to achieve 10^{-4} optimality value. The vertical lines indicate the event of phase switching in RDA$^+$. The vertical axes on the left are in logarithmic scale.

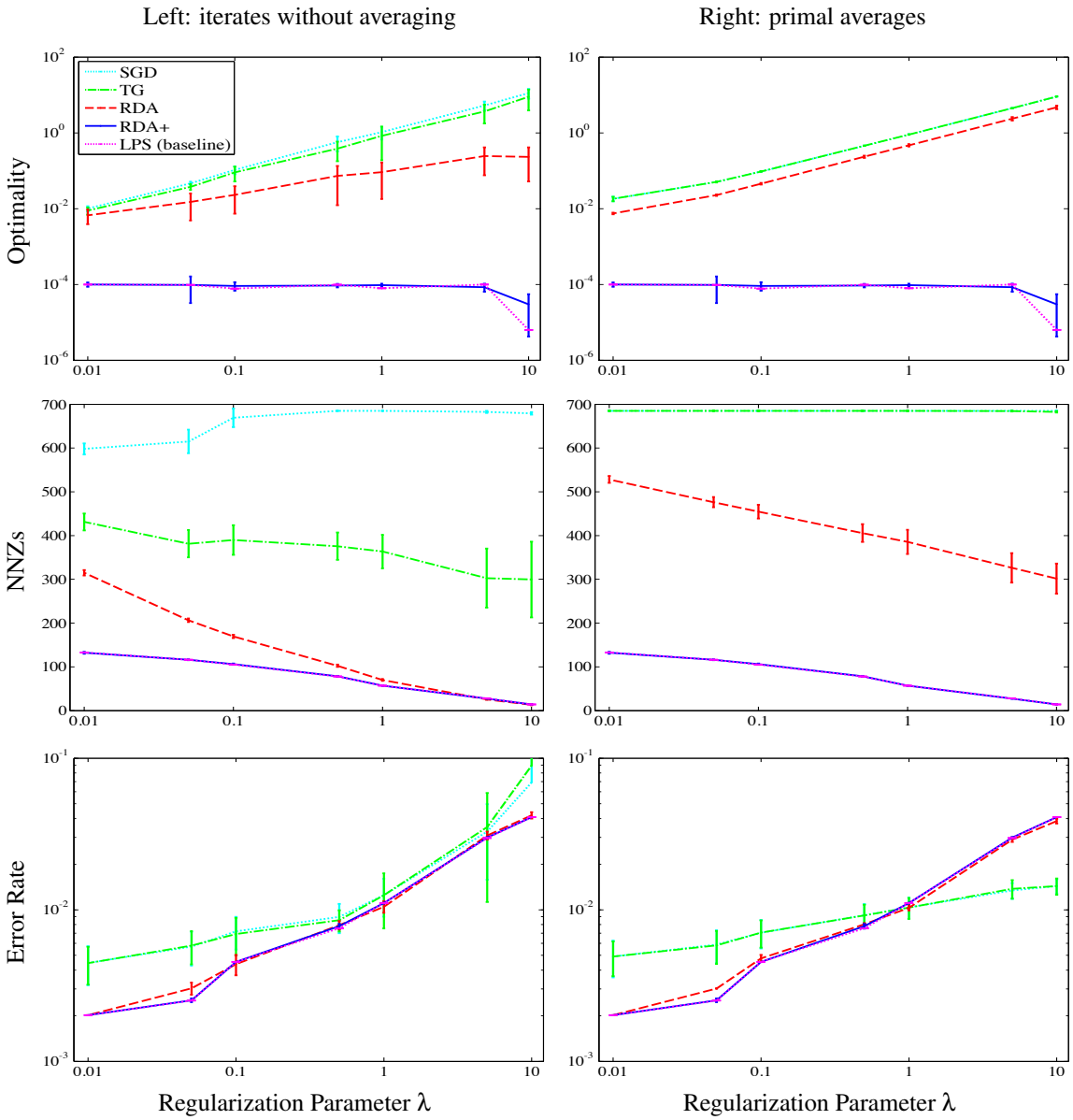


Figure 3: Quality of solutions (MNIST 6 vs. 7) in terms of the optimality, the number of nonzero components (without intercepts), and the test error rate, measured for 100 different random permutations of the training set. The plots on the left show the results for the iterates without averaging, and those on the right show averaged primal iterates for algorithms SGD, TG, and RDA, and non-averaged iterates for RDA$^+$ and LPS. The SGD, TG, and RDA algorithms are run up to the time taken for RDA$^+$ to achieve a 10^{-4} threshold in the measure (55), whereas LPS is run to convergence. Axes in the first and third rows are in logarithmic scale.

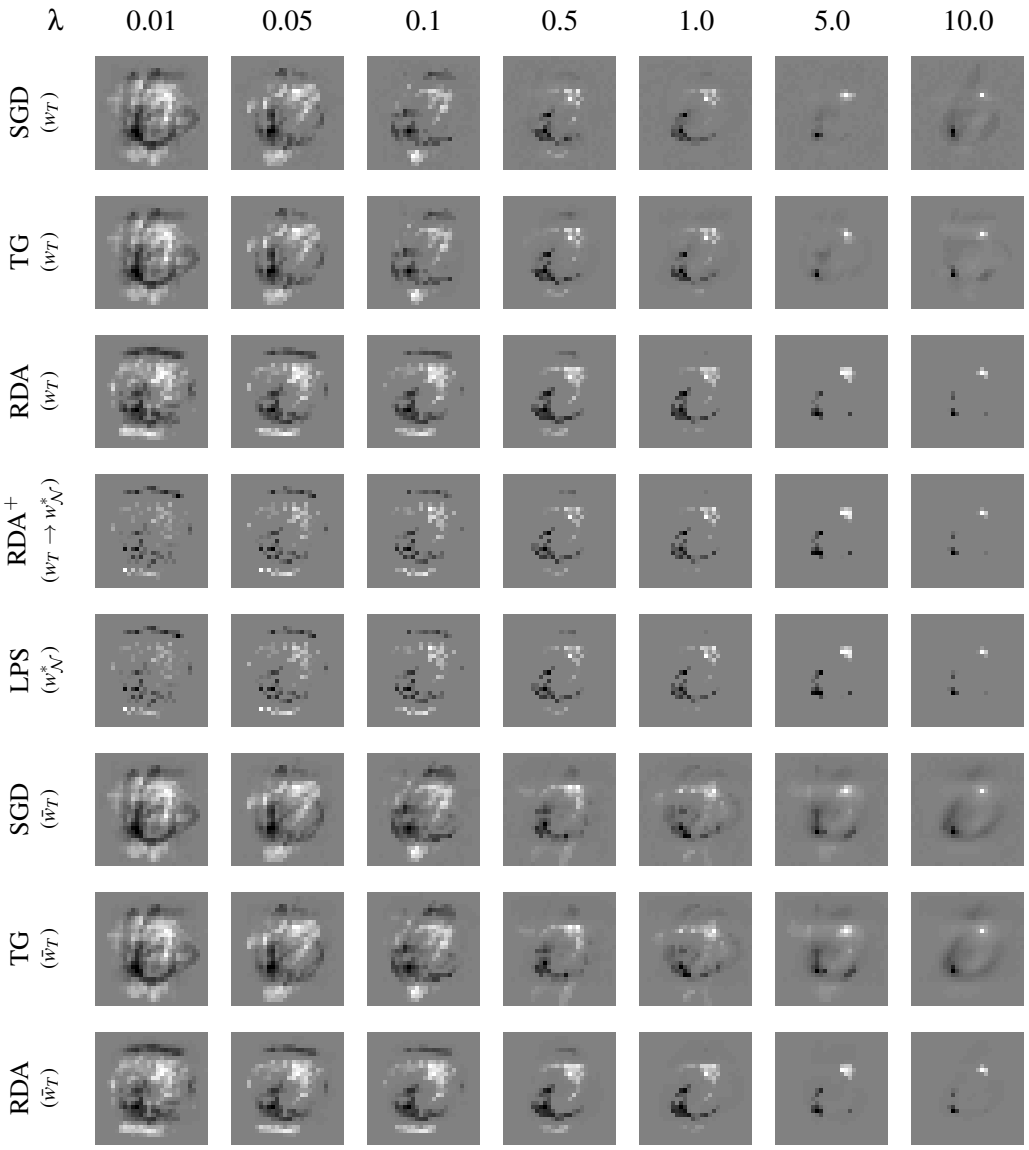


Figure 4: Sparsity patterns of the solutions for classification of the digits 6 and 7 in the MNIST data set. The regularization parameter λ is varied in the range of $[0.01, 10]$. The spots represent the positive (bright) and negative (dark) values, whereas the gray background represents zero. The top three rows show the solutions acquired without averaging, and the bottom three rows show those obtained with primal averaging. The two rows in the middle presents the solutions from RDA$^+$ and LPS. The algorithms SGD, TG, and RDA are run up to the time taken for RDA$^+$ to achieve a solution with 10^{-4} optimality value; the batch algorithm LPS is run without time limit. Note that for each value of λ, the sparsest solutions are obtained by RDA$^+$ and LPS.

complicated. Second, multiple samples could be used as in Dekel et al. (2012) to construct an approximate subgradient with reduced variance, which may thus lead to faster identification. Third, it is likely that the nondegeneracy assumption can be weakened, in which case a "super-manifold" of the optimal manifold would be identifiable. We leave such investigations to future work.

Acknowledgments

We thank our colleague Ben Recht for helpful comments and discussions. We are also grateful to the referees and the editor for their perceptive technical comments and their patient handling of the manuscript. We are particularly indebted to one of the anonymous referees, whose extensive, detailed comments and pointers led us to strengthen and simplify our analysis.

This research was supported in part by National Science Foundation Grant DMS-0914524, and in part by Deutsche Forschungsgemeinschaft (DFG) within the Collaborative Research Center SFB 876 "Providing Information by Resource-Constrained Analysis", project C1.

Appendix A. Strong Minimizer Property

In this section we prove Theorem 5 and its corollaries stated in Section 2.5, based on results from manifold analysis and other elementary arguments. Our proof is similar to that of Wright (2012, Theorem 2.5), but simpler.

We first state an elementary result on manifold characterization, which is proved by Vaisman (1984, Sections 1.4-1.5) and Wright (2012, Appendix A).

Lemma 18 *Let the manifold $\mathcal{M} \subset \mathbb{R}^n$ containing $\bar{z}$ be characterized by a $\mathcal{C}^p$ ($p \geq 2$) function $H : \mathbb{R}^n \to \mathbb{R}^k$. Then there is $\bar{y} \in \mathbb{R}^{n-k}$ and a $\mathcal{C}^p$ function G mapping some neighborhood of $\bar{y}$ to $\mathbb{R}^n$ such that $G(y) \in \mathcal{M}$ for all y near $\bar{y}$. Moreover, $G(y) - \bar{z} = Y(y-\bar{y}) + O(\|y-\bar{y}\|^2)$, where $Y \in \mathbb{R}^{n\times(n-k)}$ is an orthonormal matrix whose columns span the tangent space to $\mathcal{M}$ at $\bar{z}$.*

The next result, from Wright (2012, Lemma 2.2), shows how perturbations from a point at which the objective function is partly smooth can be decomposed according to the manifold characterization above.

Lemma 19 *Let the manifold $\mathcal{M} \subset \mathbb{R}^n$ be characterized in a neighborhood of $\bar{z} \in \mathcal{M}$ by $\mathcal{C}^p$ mappings $H : \mathbb{R}^n \to \mathbb{R}^k$ and $G : \mathbb{R}^{n-k} \to \mathbb{R}^n$ and the point $\bar{y}$ described in Lemma 18. Then for all z near $\bar{z}$, there are unique vectors $y \in \mathbb{R}^{n-k}$ and $v \in \mathbb{R}^k$ with $\|(y^T - \bar{y}^T, v^T)\| = O(\|z-\bar{z}\|)$ such that $z = G(y) + \nabla H(\bar{z})v$.*

We also make use of a result from Wright (2012, Lemma A.1).

Lemma 20 *Consider a function $\varphi : \mathbb{R}^n \to \mathbb{R} \cup \{+\infty\}$, a point $\bar{z} \in \mathbb{R}^n$, and a manifold $\mathcal{M}$ containing $\bar{z}$ such that φ is partly smooth at $\bar{z}$ with respect to $\mathcal{M}$. If the nondegeneracy condition $0 \in \operatorname{ri} \partial\varphi(\bar{z})$ holds, then there exists $\varepsilon > 0$ such that*

$$\sup_{g \in \partial\varphi(\bar{z})} \langle g, d\rangle \geq \varepsilon\|d\|, \ \ \forall d \in N_{\mathcal{M}}(\bar{z}).$$

Proof (Theorem 5) We now proceed with the proof of the main result of Section 2.5. Recall the following assumptions:

(i) ϕ is partly smooth at w^* relative to the optimal manifold $\mathcal{M}$.

(ii) w^* is a locally strong minimizer of $\phi|_{\mathcal{M}}$ with modulus $c_{\mathcal{M}} > 0$ and radius $r_{\mathcal{M}} > 0$, and

(iii) the nondegeneracy condition (13) holds at w^*.

For the minimizer w^* of (1) and the optimal manifold $\mathcal{M}$ containing w^*, we consider the mappings H and G, the matrix Y, and the point $\bar{y} \in \mathbb{R}^{n-k}$ satisfying Lemma 18 and Lemma 19, associated with $\bar{z} = w^* \in \mathbb{R}^n$. From Lemma 19, for all w satisfying $\|w - w^*\| \leq \bar{r} \leq r_{\mathcal{M}}$ with small enough $\bar{r} > 0$, we can find unique vectors $y \in \mathbb{R}^{n-k}$ and $v \in \mathbb{R}^k$ with $\|(y^T - \bar{y}^T, v^T)\| = O(\|w - w^*\|)$ such that $w = G(y) + \nabla H(w^*)v$. Therefore we have

$$\phi(w) - \phi(w^*) = [\phi(G(y) + \nabla H(w^*)v) - \phi(G(y))] + [\phi(G(y)) - \phi(w^*)]. \tag{57}$$

From the locally strong minimizer property relative to $\mathcal{M}$ and the facts that $w^* \in \mathcal{M}$ and $G(y) \in \mathcal{M}$ for all y near $\bar{y}$, we have for the second bracketed term that

$$\phi(G(y)) - \phi(w^*) = \phi|_{\mathcal{M}}(G(y)) - \phi|_{\mathcal{M}}(w^*) \geq c_{\mathcal{M}} \|G(y) - w^*\|^2 \tag{58}$$

for all y near $\bar{y}$. Consider next the first bracketed term of (57). From Lemma 20, we have $\varepsilon > 0$ such that $\sup_{g \in \partial\phi(w^*)} \langle g, d \rangle \geq \varepsilon \|d\|$ for all $d \in N_{\mathcal{M}}(w^*)$. From the subcontinuity property (iv) of $\partial\phi(w^*)$ in Definition 4, we can choose a neighborhood of w^* sufficiently small that for all $G(y)$ in this neighborhood and $g \in \partial\phi(w^*)$, there exists $\hat{g} \in \partial\phi(G(y))$ such that $\|\hat{g} - g\| \leq \varepsilon/2$. These facts, together with convexity of ϕ, imply that for all y near $\bar{y}$ and v near 0 we have

$$\begin{aligned}
\phi(G(y) + \nabla H(w^*)v) - \phi(G(y)) &\geq \sup_{\hat{g} \in \partial\phi(G(y))} \langle \hat{g}, \nabla H(w^*)v \rangle \\
&\geq \sup_{g \in \partial\phi(w^*)} \langle g, \nabla H(w^*)v \rangle - (\varepsilon/2)\|\nabla H(w^*)v\| \\
&\geq (\varepsilon/2)\|\nabla H(w^*)v\|.
\end{aligned}$$

By substituting this inequality and (58) into (57), we obtain

$$\phi(w) - \phi(w^*) \geq (\varepsilon/2)\|\nabla H(w^*)v\| + c_{\mathcal{M}} \|G(y) - w^*\|^2.$$

By further reducing $\bar{r}$ if necessary, we can choose the neighborhood of w^* small enough to ensure that $\|\nabla H(w^*)v\| \leq 1$, and therefore

$$\begin{aligned}
\phi(w) - \phi(w^*) &\geq (\varepsilon/2)\|\nabla H(w^*)v\|^2 + c_{\mathcal{M}} \|G(y) - w^*\|^2 \\
&\geq \min(\varepsilon/2, c_{\mathcal{M}}) \left[\|\nabla H(w^*)v\|^2 + \|G(y) - w^*\|^2 \right] \\
&\geq \frac{1}{2} \min(\varepsilon/2, c_{\mathcal{M}}) \Big[\|\nabla H(w^*)v\| + \|G(y) - w^*\| \Big]^2 \\
&\geq \frac{1}{2} \min(\varepsilon/2, c_{\mathcal{M}}) \|w - w^*\|^2.
\end{aligned}$$

(The third inequality follows from the elementary bound $(a^2 + b^2) \geq \frac{1}{2}(a+b)^2$, for any scalars a and b.) We have thus shown that w^* indeed is a local strong minimizer of ϕ, without the restriction to the manifold $\mathcal{M}$, with modulus $c := \frac{1}{2}\min(\varepsilon/2, c_{\mathcal{M}})$ and radius $\bar{r}$. ■

We follow with the proofs of the remaining results of Section 2.5.

Proof (Corollary 6) Given $w \in \mathcal{O}$ with $\|w - w^*\| > \bar{r}$, we have from the convexity of ϕ that

$$\phi\left(w^* + \bar{r}\frac{w-w^*}{\|w-w^*\|}\right) \leq \phi(w^*) + \frac{\bar{r}}{\|w-w^*\|}(\phi(w) - \phi(w^*)).$$

From the locally strong minimizer property (Theorem 5), we also have

$$\phi\left(w^* + \bar{r}\frac{w-w^*}{\|w-w^*\|}\right) - \phi(w^*) \geq c\left\|\left(w^* + \bar{r}\frac{w-w^*}{\|w-w^*\|}\right) - w^*\right\|^2 = c\bar{r}^2.$$

Collecting the above two inequalities leads to the claim. ■

Proof (Corollary 7) Given $w \in \mathcal{O}$, if $\|w-w^*\| \leq \bar{r}$, then the claim follows from (15). If $\|w-w^*\| > \bar{r}$, then we have from strong convexity of ϕ that

$$\begin{aligned}\phi\left(w^* + \bar{r}\frac{w-w^*}{\|w-w^*\|}\right) \leq \phi(w^*) + \frac{\bar{r}}{\|w-w^*\|}(\phi(w) - \phi(w^*)) \\ - \frac{\sigma}{2}\frac{\bar{r}}{\|w-w^*\|}\left(1 - \frac{\bar{r}}{\|w-w^*\|}\right)\|w-w^*\|^2.\end{aligned}$$

From the locally strong minimizer property (Theorem 5), we also have

$$\phi\left(w^* + \bar{r}\frac{w-w^*}{\|w-w^*\|}\right) - \phi(w^*) \geq c\bar{r}^2.$$

Combining the above two inequalities results in

$$\phi(w) - \phi(w^*) \geq \left[\sigma/2 + \frac{\bar{r}}{\|w-w^*\|}(c - \sigma/2)\right]\|w-w^*\|^2 \geq \min(c, \sigma/2)\|w-w^*\|^2.$$

■

Appendix B. Expected Error Bounds for Iterates of RDA

In this section we provide the background for the results of Section 3, regarding the iterates generated by the RDA algorithm.

Proof (Lemma 9) For the general convex case, with $\{\beta_t\}$ chosen by (20), we consider the expected regret up to time t with respect to w^*, and obtain

$$\begin{aligned}
\mathbb{E}[R_t(w^*)] &= \mathbb{E}\left[\sum_{j=1}^{t}(F(w_j;\xi_j)+\Psi(w_j)) - \sum_{j=1}^{t}(F(w^*;\xi_j)+\Psi(w^*))\right] \\
&= \sum_{j=1}^{t}\mathbb{E}\left[\mathbb{E}\left\{(F(w_j;\xi_j)+\Psi(w_j)-F(w^*;\xi_j)-\Psi(w^*))\,|\,\xi_{[j-1]}\right\}\right] \\
&= \sum_{j=1}^{t}\mathbb{E}\left[f(w_j)+\Psi(w_j)-f(w^*)-\Psi(w^*)\right] \\
&= \sum_{j=1}^{t}\mathbb{E}\left[\phi(w_j)-\phi(w^*)\right]. \qquad (59)
\end{aligned}$$

Under Assumptions 1 and 2, there exists $c>0$ and $\bar{r}>0$ that satisfy (15), which is

$$\phi(w)-\phi(w^*) \geq c\|w-w^*\|^2, \text{ for all } w \text{ with } \|w-w^*\| \leq \bar{r},$$

as proved in Theorem 5. Noting that $I_{(\|w_j-w^*\|\leq\bar{r})}+I_{(\|w_j-w^*|>\bar{r})}=1$, we can split the right-hand side of (59) into two sums and obtain

$$\mathbb{E}[R_t(w^*)] = \sum_{j=1}^{t}\mathbb{E}\left[I_{(\|w_j-w^*\|\leq\bar{r})}\{\phi(w_j)-\phi(w^*)\}\right] + \sum_{j=1}^{t}\mathbb{E}\left[I_{(\|w_j-w^*\|>\bar{r})}\{\phi(w_j)-\phi(w^*)\}\right]. \quad (60)$$

Note that both terms on the right-hand side of (60) are nonnegative. For the first term, we have by using the regret bound (22) and the locally strong minimizer property (15) that

$$\begin{aligned}
\left(\gamma D^2+\frac{G^2}{\gamma}\right)t^{1/2} &\geq \sum_{j=1}^{t}\mathbb{E}\left[I_{(\|w_j-w^*\|\leq\bar{r})}\{\phi(w_j)-\phi(w^*)\}\right] \\
&\geq c\sum_{j=1}^{t}\mathbb{E}\left[I_{(\|w_j-w^*\|\leq\bar{r})}\|w_j-w^*\|^2\right],
\end{aligned}$$

proving the first inequality (24). For the second inequality, we have from (60), the regret bound (22), and Corollary 6 that

$$\begin{aligned}
\left(\gamma D^2+\frac{G^2}{\gamma}\right)t^{1/2} &\geq \sum_{j=1}^{t}\mathbb{E}\left[I_{(\|w_j-w^*\|>\bar{r})}\{\phi(w_j)-\phi(w^*)\}\right] \\
&\geq c\bar{r}\sum_{j=1}^{t}\mathbb{E}\left[I_{(\|w_j-w^*\|>\bar{r})}\|w_j-w^*\|\right],
\end{aligned}$$

thus proving (25).

When Ψ is strongly convex with the modulus $\sigma>0$ and $\{\beta_t\}$ chosen by (21), we apply the other regret bound (23) to (59), resulting in

$$\frac{G^2}{2\sigma}(6+\ln t) \geq \mathbb{E}R_t(w^*) \geq \sum_{j=1}^{t}\mathbb{E}\{\phi(w_j)-\phi(w^*)\} \geq \min(c,\sigma/2)\sum_{j=1}^{t}\mathbb{E}\left[\|w_j-w^*\|^2\right],$$

where for the last inequality we use the fact that w^* is a (global) strong minimizer with the modulus $\min(c, \sigma/2)$, as shown in Corollary 7. This proves (26).

■

Proof (Theorem 10) We start with the general convex case. From the Cauchy-Schwartz inequality $\|z\|_1 \leq \sqrt{m}\|z\|_2$ for a vector $z \in \mathbb{R}^m$ and Jensen's inequality, we have

$$\begin{aligned}\frac{1}{t}\sum_{j=1}^{t}\mathbb{E}\left[I_{(\|w_j-w^*\|\leq\bar{r})}\|w_j-w^*\|\right] &\leq \frac{\sqrt{t}}{t}\left[\sum_{j=1}^{t}\left\{\mathbb{E}\left[I_{(\|w_j-w^*\|\leq\bar{r})}\|w_j-w^*\|\right]\right\}^2\right]^{1/2}\\ &\leq \left[\frac{1}{t}\sum_{j=1}^{t}\mathbb{E}\left[I_{(\|w_j-w^*\|\leq\bar{r})}\|w_j-w^*\|^2\right]\right]^{1/2}\\ &\leq \frac{1}{\sqrt{c}}\left(\gamma D^2+\frac{G^2}{\gamma}\right)^{1/2}t^{-1/4},\end{aligned}$$

where the last inequality is from (24). By combining this result with (25) and the definition of $\hat{t}$ in (28), and using our standing assumption that $\bar{r} \in (0,1]$, we have for $t \geq \hat{t}$ that

$$\begin{aligned}\frac{1}{t}\sum_{j=1}^{t}\mathbb{E}\|w_j-w^*\| &= \frac{1}{t}\sum_{j=1}^{t}\mathbb{E}\left[I_{(\|w_j-w^*\|\leq\bar{r})}\|w_j-w^*\|\right]+\frac{1}{t}\sum_{j=1}^{t}\mathbb{E}\left[I_{(\|w_j-w^*\|>\bar{r})}\|w_j-w^*\|\right]\\ &\leq \frac{1}{\sqrt{c}}\left(\gamma D^2+\frac{G^2}{\gamma}\right)^{1/2}t^{-1/4}+\frac{1}{\bar{r}c}\left(\gamma D^2+\frac{G^2}{\gamma}\right)t^{-1/2}\\ &\leq \frac{1}{\sqrt{c}}\left(\gamma D^2+\frac{G^2}{\gamma}\right)^{1/2}t^{-1/4}+\frac{1}{\sqrt{\bar{r}c}}\left(\gamma D^2+\frac{G^2}{\gamma}\right)^{1/2}t^{-1/4}\\ &\leq \mu t^{-1/4}.\end{aligned}$$

for μ defined in (28).

For the strongly convex case, we have from Cauchy-Schwarz and Jensen's inequalities that

$$\frac{1}{t}\sum_{j=1}^{t}\mathbb{E}\|w_j-w^*\| \leq \left[\frac{1}{t}\sum_{j=1}^{t}\left\{\mathbb{E}\|w_j-w^*\|\right\}^2\right]^{1/2} \leq \left[\frac{1}{t}\sum_{j=1}^{t}\mathbb{E}\|w_j-w^*\|^2\right]^{1/2}.$$

Applying the bound in (26) to the last line leads to (29). ■

References

M. Anitescu. Degenerate nonlinear programming with a quadratic growth condition. *SIAM Journal on Optimization*, 10(4):1116–1135, 2000.

K. Azuma. Weighted sums of certain dependent random variables. *Tohoku Mathematical Journal*, 19(3):357–367, 1967.

P. L. Bartlett, E. Hazan, and A. Rakhlin. Adaptive online gradient descent. In *Advances in Neural Information Processing Systems 20*, pages 65–72, 2008.

L. Bottou. Stochastic learning. In *Advanced Lectures on Machine Learning*, volume 3176 of *Lecture Notes in Artificial Intelligence*, pages 146–168. Springer Verlag, 2004.

J. V. Burke and J. J. Moré. Exposing constraints. *SIAM Journal on Optimization*, 4(3):573–595, 1994.

K. L. Chung. On a stochastic approximation method. *Annals of Mathematical Statistics*, 25(3): 463–483, 1954.

O. Dekel, R. Gilad-Bachrach, O. Shamir, and L. Xiao. Optimal distributed online prediction using mini-batches. *Journal of Machine Learning Research*, 13:165–202, 2012.

J. Duchi and Y. Singer. Efficient online and batch learning using forward backward splitting. *Journal of Machine Learning Research*, 10:2899–2934, 2009.

J. Duchi, S. Shalev-Shwartz, Y. Singer, and A. Tewari. Composite objective mirror descent. In *Proceedings of the 23rd Annual Conference on Learning Theory*, 2010.

W. L. Hare and A. S. Lewis. Identifying active constraints via partial smoothness and prox-regularity. *Journal of Convex Analysis*, 11(2):251–266, 2004.

E. Hazan, A. Kalai, S. Kale, and A. Agarwal. Logarithmic regret algorithms for online convex optimization. In *Proceedings of the 19th Annual Conference on Learning Theory*, pages 499–513, 2006.

J. Kiefer and J. Wolfowitz. Stochastic estimation of the maximum of a regression function. *Annals of Mathematical Statistics*, 23(3):462–466, 1952.

K. Koh, S.-J. Kim, and S. Boyd. An interior-point method for large-scale ℓ_1-regularized logistic regression. *Journal of Machine Learning Research*, 8:1519–1555, 2007.

J. Langford, L. Li, and T. Zhang. Sparse online learning via truncated gradient. *Journal of Machine Learning Research*, 10:777–801, 2009.

A. S. Lewis. Active sets, nonsmoothness, and sensitivity. *SIAM Journal on Optimization*, 13: 702–725, 2003.

A. S. Lewis and S. J. Wright. A proximal method for composite minimization. Technical report, University of Wisconsin-Madison, August 2008.

Q. Lin, X. Chen, and J. Peña. A sparsity preserving stochastic gradient method for composite optimization. Working paper, Tepper School of Business, Carnegie Mellon University, March 2011. Revised May 2012.

A. Nemirovski and D. Yudin. On Cezari's convergence of the steepest descent method for approximating saddle point of convex-concave functions. *Soviet Mathematics-Doklady*, 19, 1978.

A. Nemirovski and D. Yudin. *Problem Complexity and Method Efficiency in Optimization*. John Wiley, 1983.

A. Nemirovski, A. Juditsky, G. Lan, and A. Shapiro. Robust stochastic approximation approach to stochastic programming. *SIAM Journal on Optimization*, 19(4):1574–1609, 2009.

Y. Nesterov. *Introductory Lectures on Convex Optimization: A Basic Course*. Kluwer Academic Publishers, 2004.

Y. Nesterov. Primal-dual subgradient methods for convex problems. *Mathematical Programming*, 120:221–259, 2009.

C. Oberlin and S. J. Wright. Active set identification in nonlinear programming. *SIAM Journal on Optimization*, 17(2):577–605, 2006.

B. T. Polyak. New stochastic approximation type procedures. *Avtomatica i Telemekhanika*, 7: 98–107, 1990.

B. T. Polyak and A. B. Juditsky. Acceleration of stochastic approximation by averaging. *SIAM Journal on Control and Optimization*, 30(4):838–855, 1992.

H. Robbins and S. Monro. A stochastic approximation method. *Annals of Mathematical Statistics*, 22(3):400–407, 1951.

J. Sacks. Asymptotic distribution of stochastic approximation procedures. *Annals of Mathematical Statistics*, 29(2):373–405, 1958.

S. Shalev-Shwartz, Y. Singer, and N. Srebro. Pegasos: Primal Estimated sub-GrAdient SOlver for SVM. In *Proceedings of the 24th International Conference on Machine Learning*, pages 807–814, 2007.

W. Shi, G. Wahba, S. J. Wright, K. Lee, R. Klein, and B. Klein. LASSO-Patternsearch algorithm with application to opthalmology data. *Statistics and its Interface*, 1:137–153, 2008.

I. Vaisman. *A First Course in Differential Geometry*. Monographs and Textbooks in Pure and Applied Mathematics. Marcel Dekker, 1984.

S. J. Wright. Identifiable surfaces in constrained optimization. *SIAM Journal on Control and Optimization*, 31(4):1063–1079, 1993.

S. J. Wright. Accelerated block-coordinate relaxation for regularized optimization. *SIAM Journal on Optimization*, 22(1):159–186, 2012.

S. J. Wright, R. D. Nowak, and M. A. T. Figueiredo. Sparse reconstruction by separable approximation. *IEEE Transactions on Signal Processing*, 57:2479–2493, 2009.

L. Xiao. Dual averaging methods for regularized stochastic learning and online optimization. *Journal of Machine Learning Research*, 11:2543–2596, 2010.

T. Zhang. Solving large scale linear prediction problems using stochastic gradient descent algorithms. In *Proceedings of the 21st International Conference on Machine Learning*, 2004.

M. Zinkevich. Online convex programming and generalized infinitesimal gradient ascent. In *Proceedings of the 20th International Conference on Machine Learning*, pages 928–936, 2003.

Journal of Machine Learning Research 13 (2012) 1745-1808 Submitted 1/12; Published 6/12

Variational Multinomial Logit Gaussian Process

Kian Ming A. Chai CKIANMIN@DSO.ORG.SG
DSO National Laboratories
20 Science Park Drive
Singapore 118230

Editor: Manfred Opper

Abstract

Gaussian process prior with an appropriate likelihood function is a flexible non-parametric model for a variety of learning tasks. One important and standard task is multi-class classification, which is the categorization of an item into one of several fixed classes. A usual likelihood function for this is the multinomial logistic likelihood function. However, exact inference with this model has proved to be difficult because high-dimensional integrations are required. In this paper, we propose a variational approximation to this model, and we describe the optimization of the variational parameters. Experiments have shown our approximation to be tight. In addition, we provide data-independent bounds on the marginal likelihood of the model, one of which is shown to be much tighter than the existing variational mean-field bound in the experiments. We also derive a proper lower bound on the predictive likelihood that involves the Kullback-Leibler divergence between the approximating and the true posterior. We combine our approach with a recently proposed sparse approximation to give a variational sparse approximation to the Gaussian process multi-class model. We also derive criteria which can be used to select the inducing set, and we show the effectiveness of these criteria over random selection in an experiment.

Keywords: Gaussian process, probabilistic classification, multinomial logistic, variational approximation, sparse approximation

1. Introduction

Gaussian process (GP, Rasmussen and Williams, 2006) is attractive for non-parametric probabilistic inference because knowledge can be specified directly in the prior distribution through the mean and covariance function of the process. Inference can be achieved in closed form for regression under Gaussian noise, but approximation is necessary under other likelihoods. For binary classification with logistic and probit likelihoods, a number of approximations have been proposed and compared (Nickisch and Rasmussen, 2008). These are either Gaussian or factorial approximations to the posterior of the latent function values at the observed inputs. Compared to the binary case, progress is slight for multi-class classification. The main hurdle is the need for—and yet the lack of—accurate approximation to the multi-dimensional integration of the likelihood or the log-likelihood against Gaussians (Seeger and Jordan, 2004).

For multi-class classification with latent Gaussian process, different likelihood functions may be used: the multinomial logistic function (Williams and Barber, 1998; Gibbs, 1997; Seeger and Jordan, 2004), also called the soft-max (Bridle, 1989); the multinomial probit function (Girolami and Rogers, 2006); and the uniform noise model (Kim and Ghahramani, 2006). For inference, the exact posterior is usually approximated with a Gaussian or a factorial distribution, similar to

the binary case. Different principles may be used to fit the approximation: Laplace approximation (Williams and Barber, 1998); assumed density filtering (Seeger and Jordan, 2004) and expectation propagation (Kim and Ghahramani, 2006); and variational approximation (Gibbs, 1997; Girolami and Rogers, 2006).

This paper addresses the variational approximation of the multinomial logit Gaussian process model, where the likelihood function is the multinomial logistic. In contrast with the variational mean-field approach of Girolami and Rogers (2006), where a factorial approximation is assumed from the onset, we use a full Gaussian approximation on the posterior of the latent function values. The approximation is fitted by minimizing the Kullback-Leibler divergence to the true posterior, which is known to be the same as maximizing a variational lower bound on the marginal likelihood. This procedure requires the expectation of the log-likelihood under the approximating distribution. This is intractable in general, so we introduce a bound on the expected log-likelihood and optimize this bound instead. This contrasts with the proposal by Gibbs (1997) to bound the multinomial logistic likelihood directly. Our bound on the expected log-likelihood is derived using a novel variational method that results in the multinomial logistic being associated with a mixture of Gaussians. Monte-Carlo simulations indicate that this bound is very tight in practice.

Our approach gives a lower bound on the marginal likelihood of the model. By fixing some variational parameters, we arrive at data-independent bounds on the marginal likelihood. These bounds depend only on the number of classes and kernel Gram matrix of the data, but not on the classifications in the data. On four UCI data sets, the one bound we evaluated is tighter than the variational mean-field bound (Girolami and Rogers, 2006).

Although the variational approximation provides a lower bound on the marginal likelihood, approximate prediction in the usual straightforward manner does not necessarily give a lower bound on the predictive likelihood. We show that a proper lower bound on the predictive likelihood can be obtained when we take into account the Kullback-Leibler divergence between the approximating and the true posterior. This perspective supports the minimization of the divergence as a criterion for approximate inference.

To address large data sets, we give a sparse approximation to the multinomial logit Gaussian process model. In a natural manner, this sparse approximation combines our proposed variational approximation with the variational sparse approximation that has been introduced for regression (Titsias, 2009a). The result maintains a variational lower bound on the marginal likelihood, which can be used to guide model learning. We also introduce scoring criteria for the selection of the inducing variables in the sparse approximation. Experiments indicate that the criteria are effective.

1.1 Overview

In Section 2, we describe the latent Gaussian process model with the multinomial logistic likelihood, and we give the variational lower bound on the marginal likelihood for approximate inference. The data-independent bounds on the marginal likelihood are developed in this section and so are the bounds for the predictive likelihood. In Section 3, we provide the necessary updates to optimize the variational bound. Sparse approximation is presented in Section 4. Section 5 looks at the sum-to-zero property that exists in our variational inference for certain covariance functions. This is the property that has been used in motivating several single-machine multi-class support vector machines (SVMs). Section 6 addresses model learning for the multinomial logit Gaussian process model. It also looks at the active selection of the inducing set for sparse approximation. Section 7

outlines the computational complexity of our approach. Related work is discussed in Section 8. Section 9 describes several experiments and gives the results. Among others, we compare the tightness of our variational approximation to the variational mean-field approximation (Girolami and Rogers, 2006), and the errors of our classification results with those given by four single-machine multi-class SVMs. Section 10 concludes and provides further discussions.

1.2 Notation

Vectors are represented by lower-case bold-faced letters, and matrices are represented by upper-case normal-faced letters. The transpose of matrix X is denoted by X^{T}. An asterisk $*$ in the superscript is used for the optimized value of a quantity or function. Sometimes it is used twice when optimized with respect to two variables. For example, if $h(x,y)$ is a function, $h^*(y)$ is $h(x,y)$ optimized over x, and h^{**} is $h(x,y)$ optimized over x and y. The dependency of a function on its variables is frequently suppressed when the context is clear: we write h instead of $h(x,y)$ and h^* instead of $h^*(y)$. In optimizing a function $h(x)$ over x, x^{fx} and x^{NR} refers to fixed-point update and Newton-Raphson update respectively, while x^{cc} refers to an update using the convex combination $x^{\mathrm{cc}} = (1-\eta)x_1 + \eta x_2$, where $\eta \in [0,1]$ is to be determined, and x_1 and x_2 are in the domain of optimization.

We use $\mathbf{x}_i$ for an input that has to be classified into one of C classes. The class of $\mathbf{x}_i$ is denoted by $\mathbf{y}_i$ using the one-of-C encoding. Hence, $\mathbf{y}_i$ is in the canonical basis of $\mathbb{R}^C$, which is the set $\{\mathbf{e}^c\}_{c=1}^C$, where $\mathbf{e}^c$ has one at the cth entry and zero everywhere else. Class index c is used as superscript, while datum index i is used as subscript. The cth entry in $\mathbf{y}_i$ is denoted by y_i^c, which is in $\{0,1\}$, and $\mathbf{x}_i$ belongs to the cth class if $y_i^c = 1$.

Both $\mathbf{x}_i$ and $\mathbf{y}_i$ are observed variables. Associated with each y_i^c is a latent random function response f_i^c. For sparse approximation, we introduce another layer of latent variables, which we denote by $\mathbf{z}$ collectively. These are called the inducing variables. Other variables and functions associated with the sparse approximation are given a tilde $\sim$ accent. The asterisk subscript is used on $\mathbf{x}$, $\mathbf{y}$, $\mathbf{f}$ and $\mathbf{z}$ for two different purposes depending on the context: it is used to indicate a test input for predictive inference, and it is also used for a site under consideration for inclusion to the inducing set for sparse approximation.

We use p to represent the probability density determined by the model and the data, including the case where the model involves sparsity. Any variational approximation to p is denoted by q.

2. Model and Variational Inference

We recall the multinomial logit Gaussian process model (Williams and Barber, 1998) in Section 2.1. We add a simple generalization of the model to include the prior covariance between the latent functions. Bayesian inference with this model is outlined in Section 2.2; this is intractable. We provide variational bounds and approximate inference for the model in Section 2.3.

2.1 Model

For classifying or categorizing the ith input $\mathbf{x}_i$ into one of C classes, we use a vector of C indicator variables $\mathbf{y}_i \in \{\mathbf{e}^c\}$, wherein the cth entry, y_i^c, is one if $\mathbf{x}_i$ is in class c and zero otherwise. We introduce C latent functions, $f^1, \ldots, f^C$, on which we place a zero mean Gaussian process prior

$$\left\langle f^c(\mathbf{x}) f^{c'}(\mathbf{x}') \right\rangle = K^{\mathrm{c}}_{cc'} k^{\mathrm{x}}(\mathbf{x}, \mathbf{x}'), \tag{1}$$

where $K^{\mathrm{c}}_{cc'}$ is the (c,c')th entry of a C-by-C positive semi-definite matrix K^{c} for modeling inter-function covariances, and k^{x} is a covariance function on the inputs. Let $f_i^c \stackrel{\text{def}}{=} f^c(\mathbf{x}_i)$. Given the vector of function values $\mathbf{f}_i \stackrel{\text{def}}{=} (f_i^1, \ldots, f_i^C)^{\mathrm{T}}$ at $\mathbf{x}_i$, the likelihood for the class label is the multinomial logistic

$$p(y_i^c = 1|\mathbf{f}_i) \stackrel{\text{def}}{=} \frac{\exp f_i^c}{\sum_{c'=1}^C \exp f_i^{c'}}. \tag{2}$$

This can also be written as

$$p(\mathbf{y}_i|\mathbf{f}_i) = \frac{\exp \mathbf{f}_i^{\mathrm{T}} \mathbf{y}_i}{\sum_{c=1}^C \exp \mathbf{f}_i^{\mathrm{T}} \mathbf{e}^c}.$$

These two expressions for the likelihood function will be used interchangeably. We use the first expression when the interest on the class c and the second when the interest is on $\mathbf{f}_i$.

The above model for the latent functions f^cs has been used previously for multi-task learning (Bonilla et al., 2008), where f^c is the latent function for the cth task. Most prior works on multi-class Gaussian process (Williams and Barber, 1998; Seeger and Jordan, 2004; Kim and Ghahramani, 2006; Girolami and Rogers, 2006) have chosen K^{c} to be the C-by-C identity matrix, so their latent functions are identical and independent. Williams and Barber (1998) have made this choice because the inter-function correlations are usually difficult to specify, although they have acknowledged that such correlations can be included in general. We agree with them on the difficulty, but we choose to address it by estimating K^{c} from observed data, as has been done for multi-task learning (Bonilla et al., 2008). If K^{c} is the identity matrix, then the block structure of the covariance matrix between the latent function values can be exploited to reduce computation (Seeger and Jordan, 2004).

The model in Equation 1 is known as the *separable model* for covariance. It is perhaps the simplest manner to involve inter-function correlations. One can also consider more involved models, such as those using convolution (Ver Hoef and Barry, 1998) and transformation (Lázaro-Gredilla and Figueiras-Vidal, 2009). Our presentation will mostly be general and applicable to these as well.

2.2 Exact Inference

Given a set of n observations $\{(\mathbf{x}_i, \mathbf{y}_i)\}_{i=1}^n$, we have an nC-vector $\mathbf{y}$ (resp. $\mathbf{f}$) of indicator variables (resp. latent function values) by stacking the $\mathbf{y}_i$s (resp. $\mathbf{f}_i$s). Let X collects $\mathbf{x}_1, \ldots, \mathbf{x}_n$. Dependencies on the inputs X are suppressed henceforth unless necessary.

By Bayes' rule, the posterior over the latent function values is $p(\mathbf{f}|\mathbf{y}) = p(\mathbf{y}|\mathbf{f})p(\mathbf{f})/p(\mathbf{y})$, where $p(\mathbf{y}|\mathbf{f}) = \prod_i p(\mathbf{y}_i|\mathbf{f}_i)$ and $p(\mathbf{y}) = \int p(\mathbf{y}|\mathbf{f})p(\mathbf{f})\mathrm{d}\mathbf{f}$. Inference for a test input $\mathbf{x}_*$ is performed in two steps. First we compute the distribution of latent function values at $\mathbf{x}_*$: $p(\mathbf{f}_*|\mathbf{y}) = \int p(\mathbf{f}_*|\mathbf{f})\, p(\mathbf{f}|\mathbf{y})\mathrm{d}\mathbf{f}$. Then we compute the posterior predictive probability of $\mathbf{x}_*$ being in class c, which is given by $p(y_*^c = 1|\mathbf{y}) = \int p(y_*^c = 1|\mathbf{f}_*)\, p(\mathbf{f}_*|\mathbf{y})\mathrm{d}\mathbf{f}_*$.

2.3 Variational Inference

The integrals needed in the exact inference steps are intractable due to the non-Gaussian likelihood $p(\mathbf{y}|\mathbf{f})$. To progress, we employ variational inference in the following manner. The posterior $p(\mathbf{f}|\mathbf{y})$ is approximated by the variational posterior $q(\mathbf{f}|\mathbf{y})$ by minimizing the Kullback-Leibler (KL) divergence

$$\mathrm{KL}\left(q(\mathbf{f}|\mathbf{y}) \,\|\, p(\mathbf{f}|\mathbf{y})\right) = \int q(\mathbf{f}|\mathbf{y}) \log \frac{q(\mathbf{f}|\mathbf{y})}{p(\mathbf{f}|\mathbf{y})}\mathrm{d}\mathbf{f}.$$

This is the difference between the log marginal likelihood $\log p(\mathbf{y})$ and a variational lower bound

$$\log Z_B = -\mathrm{KL}\left(q(\mathbf{f}|\mathbf{y}) \,\|\, p(\mathbf{f})\right) + \sum_{i=1}^{n} \ell_i(\mathbf{y}_i; q), \tag{3}$$

where

$$\ell_i(\mathbf{y}_i; q) \stackrel{\text{def}}{=} \int q(\mathbf{f}_i|\mathbf{y}) \log p(\mathbf{y}_i|\mathbf{f}_i)\, \mathrm{d}\mathbf{f}_i \tag{4}$$

is the expected log-likelihood of the ith datum under distribution q, and

$$q(\mathbf{f}_i|\mathbf{y}) = \int q(\mathbf{f}|\mathbf{y}) \prod_{j \neq i} \mathrm{d}\mathbf{f}_j \tag{5}$$

is the variational marginal distribution of $\mathbf{f}_i$; see Appendix B.1 for details. The Kullback-Leibler divergence component of $\log Z_B$ can be interpreted as the regularizing factor for the approximate posterior $q(\mathbf{f}|\mathbf{y})$, while the expected log-likelihood can be interpreted as the data fit component. The inequality $\log p(\mathbf{y}) \geq \log Z_B$ with Z_B expressed as in Equation 3 has been given previously in the same context of variational inference for Gaussian latent models (Challis and Barber, 2011). It has also been used in the online learning setting (see Banerjee, 2006, and references therein).

For approximate inference on a test input $\mathbf{x}_*$, first we obtain the approximate posterior, which is $q(\mathbf{f}_*|\mathbf{y}) \stackrel{\text{def}}{=} \int p(\mathbf{f}_*|\mathbf{f})\, q(\mathbf{f}|\mathbf{y}) \mathrm{d}\mathbf{f}$. Then we obtain a lower bound to the approximate predictive probability for class c:

$$\begin{aligned} \log q(y_*^c = 1|\mathbf{y}) &\stackrel{\text{def}}{=} \log \int p(y_*^c = 1|\mathbf{f}_*)\, q(\mathbf{f}_*|\mathbf{y})\, \mathrm{d}\mathbf{f}_* \\ &\geq \int q(\mathbf{f}_*|\mathbf{y}) \log p(y_*^c = 1|\mathbf{f}_*)\, \mathrm{d}\mathbf{f}_* \\ &= \ell_*(y_*^c = 1; q), \end{aligned} \tag{6}$$

where the inequality is due to Jensen's inequality. The corresponding upper bound is obtained using the property of mutual exclusivity:

$$q(y_*^c = 1|\mathbf{y}) = 1 - \sum_{c' \neq c} q(y_*^{c'} = 1|\mathbf{y}) \leq 1 - \sum_{c' \neq c} \exp \ell_*(y_*^{c'} = 1; q). \tag{7}$$

The Bayes classification decision based on the upper bound is consistent with that based on the lower bound, since

$$\arg\max_c \left(1 - \sum_{c' \neq c} \exp \ell_*^{c'}\right) = \arg\max_c \left(1 - \sum_{c'=1}^{C} \exp \ell_*^{c'} + \exp \ell_*^c\right) = \arg\max_c \left(\exp \ell_*^c\right),$$

where we have written ℓ_*^c for $\ell_*(y_*^c = 1; q)$.

The variational inference procedure outlined here depends on the ability to compute expressions (a) $\mathrm{KL}\left(q(\mathbf{f}|\mathbf{y}) \,\|\, p(\mathbf{f})\right)$, (b) $q(\mathbf{f}_*|\mathbf{y})$ and (c) $\ell_i(\mathbf{y}_i; q)$. Expressions (a) and (b) can be made tractable by constraining $q(\mathbf{f}|\mathbf{y})$ to be a Gaussian density with mean $\mathbf{m}$ and covariance V, which are the variational parameters. For (c), we compute its lower bound instead, as detailed in the next section.

Remark 1 *Approximate prediction using the approximate posterior as outlined above is the more common approach (see, for example, Rasmussen and Williams, 2006, §3.5). An alternative is to use $p(\mathbf{y}_*|\mathbf{y}) = p(\mathbf{y}_*, \mathbf{y})/p(\mathbf{y})$ directly. Lower bounds to the marginal likelihoods $p(\mathbf{y}_*, \mathbf{y})$ and $p(\mathbf{y})$ may replace the exact values if they are tight. However, this procedure is more expensive in general since an (approximate) marginal likelihood has to be computed for the training data together with the test data point for every test point.*

2.3.1 VARIATIONAL BOUNDS FOR EXPECTED LOG-LIKELIHOOD

Equations 3 to 7 require the computation of the expected log-likelihood under $q(\mathbf{f}|\mathbf{y})$:

$$\ell(\mathbf{y};q) \stackrel{\text{def}}{=} \int q(\mathbf{f}|\mathbf{y}) \log p(\mathbf{y}|\mathbf{f})\, \mathrm{d}\mathbf{f}, \tag{8}$$

where we have suppressed the datum indices i and $*$ here and henceforth for this section. In our setting, $q(\mathbf{f}|\mathbf{y})$ is a Gaussian density with mean $\mathbf{m}$ and covariance V, and we regard these parameters to be constant throughout this section. The subject of this section is lower bounds on $\ell(\mathbf{y};q)$. Two trivial lower bounds can be obtained by expanding $p(\mathbf{y}|\mathbf{f})$ and using the Jensen's inequality:

$$\ell(\mathbf{y};q) \geq \mathbf{m}^{\mathrm{T}}\mathbf{y} - \log \sum_{c=1}^{C} \exp\left[\mathbf{m}^{\mathrm{T}}\mathbf{e}^c + \frac{1}{2}(\mathbf{y}-\mathbf{e}^c)^{\mathrm{T}} V (\mathbf{y}-\mathbf{e}^c)\right], \tag{9}$$

$$\ell(\mathbf{y};q) \geq \mathbf{m}^{\mathrm{T}}\mathbf{y} - \log \sum_{c=1}^{C} \exp\left[\mathbf{m}^{\mathrm{T}}\mathbf{e}^c + \frac{1}{2}(\mathbf{e}^c)^{\mathrm{T}} V \mathbf{e}^c\right]. \tag{10}$$

These bounds can be very loose. In this section, we give a variational lower bound, and we have found this bound to be quite tight when the variational parameters are optimized. This bound exploits that if a prior $r(\mathbf{f})$ is a mixture of C Gaussians with a particular set of parameters, then the corresponding posterior under the multinomial logistic likelihood is a C-variate Gaussian. We introduce this bound in terms of probability distributions and then express it in terms of variational parameters.

Lemma 2 *Let $r(\mathbf{f}|\mathbf{y})$ be a C-variate Gaussian density with mean $\mathbf{a}$ and precision W, and let $\mathbf{a}^c$ be such that $W\mathbf{a}^c = W\mathbf{a} + \mathbf{e}^c - \mathbf{y}$. If $r(\mathbf{f}) = \sum_{c=1}^{C} \gamma^c r^c(\mathbf{f})$ is the mixture of C Gaussians model on $\mathbf{f}$ with mixture proportions and components*

$$\gamma^c \stackrel{\text{def}}{=} \frac{\exp\left[\frac{1}{2}(\mathbf{a}^c)^{\mathrm{T}} W \mathbf{a}^c\right]}{\sum_{c'} \exp\left[\frac{1}{2}(\mathbf{a}^{c'})^{\mathrm{T}} W \mathbf{a}^{c'}\right]}, \qquad r^c(\mathbf{f}) \stackrel{\text{def}}{=} \frac{|W|^{1/2}}{(2\pi)^{C/2}} \exp\left[-\frac{1}{2}(\mathbf{f}-\mathbf{a}^c)^{\mathrm{T}} W (\mathbf{f}-\mathbf{a}^c)\right],$$

and if

$$r(\mathbf{y}) = \frac{\exp\left[\frac{1}{2}\mathbf{a}^{\mathrm{T}} W \mathbf{a}\right]}{\sum_{c=1}^{C} \exp\left[\frac{1}{2}(\mathbf{a}^c)^{\mathrm{T}} W \mathbf{a}^c\right]}, \tag{11}$$

then

$$\ell(\mathbf{y};q) \geq h(\mathbf{y};q,r) \stackrel{\text{def}}{=} \int q(\mathbf{f}|\mathbf{y}) \log r(\mathbf{f}|\mathbf{y}) \mathrm{d}\mathbf{f} + \log r(\mathbf{y}) - \log \sum_{c=1}^{C} \gamma^c \int q(\mathbf{f}|\mathbf{y}) r^c(\mathbf{f}) \mathrm{d}\mathbf{f}. \tag{12}$$

Proof The choice of notation used in the lemma will be clear from its proof. We begin with a variational posterior distribution $r(\mathbf{f}|\mathbf{y})$. Denote by $r(\mathbf{f})$ the corresponding prior distribution that gives this posterior when combined with the exact data likelihood $p(\mathbf{y}|\mathbf{f})$; that is

$$r(\mathbf{f}|\mathbf{y}) = p(\mathbf{y}|\mathbf{f})\, r(\mathbf{f}) / r(\mathbf{y}), \qquad \text{where} \qquad r(\mathbf{y}) \stackrel{\text{def}}{=} \int p(\mathbf{y}|\mathbf{f})\, r(\mathbf{f}) \mathrm{d}\mathbf{f}.$$

Rearranging for $p(\mathbf{y}|\mathbf{f})$ and putting back into $\ell(\mathbf{y};q)$ defined by (8) gives

$$\ell(\mathbf{y};q) = \int q(\mathbf{f}|\mathbf{y}) \log r(\mathbf{f}|\mathbf{y}) \mathrm{d}\mathbf{f} + \log r(\mathbf{y}) - \int q(\mathbf{f}|\mathbf{y}) \log r(\mathbf{f}) \mathrm{d}\mathbf{f}. \tag{13}$$

This is valid for any choice of distribution $r(\mathbf{f}|\mathbf{y})$, but let us choose it to be a C-variate Gaussian density with mean $\mathbf{a}$ and precision W. After some algebraic manipulation detailed in Appendix B.2, we obtain the expressions for $r(\mathbf{f})$ and $r(\mathbf{y})$ given in the lemma. We proceed with Jensen's inequality to move the logarithm outside the integral for the last term on the right of (13). This leads to the lower bound (12). ■

Remark 3 *The first two terms in the expression for the expected log-likelihood $\ell(\mathbf{y};q)$ given by* (13) *are computable, since $r(\mathbf{f}|\mathbf{y})$ is Gaussian by definition, and $r(\mathbf{y})$ is given in* (11)*; however, the third term remains intractable since $r(\mathbf{f})$ is a mixture of Gaussians. Hence the additional step of using the Jensen's inequality is required to obtain the lower bound $h(\mathbf{y};q,r)$ in* (12) *that is computable.*

Remark 4 *Lemma 2 depends only on the multinomial logistic likelihood function. It does not depend on the distribution $q(\mathbf{f}|\mathbf{y})$. In particular, $q(\mathbf{f}|\mathbf{y})$ can be non-Gaussian.*

Lemma 5 *Let W be a C-by-C positive semi-definite matrix, and let $\mathbf{a} \in \mathbb{R}^C$. Define $S \stackrel{\text{def}}{=} V^{-1} + W$, $\mathbf{b} \stackrel{\text{def}}{=} W(\mathbf{m} - \mathbf{a}) + \mathbf{y}$, and*

$$g^c(\mathbf{y};q,\mathbf{a},W) \stackrel{\text{def}}{=} \exp\left[\mathbf{m}^{\mathrm{T}}\mathbf{e}^c + \frac{1}{2}(\mathbf{b}-\mathbf{e}^c)^{\mathrm{T}}S^{-1}(\mathbf{b}-\mathbf{e}^c)\right]. \tag{14}$$

Then

$$\ell(\mathbf{y};q) \geq h(\mathbf{y};q,\mathbf{a},W) = \frac{C}{2} + \frac{1}{2}\log|SV| - \frac{1}{2}\operatorname{tr} SV + \mathbf{m}^{\mathrm{T}}\mathbf{y} - \log\sum_{c=1}^{C} g^c(\mathbf{y};q,\mathbf{a},W). \tag{15}$$

Proof This follows from Lemma 2 by expressing $h(\mathbf{y};q,r)$ in terms of parameters W and $\mathbf{a}$; the derivation is in Appendix B.3. Matrix W is allowed to be singular because our derivation does not involve the inversion of W; and the determinants of W taken in $r(\mathbf{f}|\mathbf{y})$ and $r^c(\mathbf{f})$ directly cancel out by subtraction, so continuity arguments can be applied. ■

We can view h given in (15) as parameterized either by W and $\mathbf{a}$ or by S and $\mathbf{b}$. For the latter view, the definitions of S and $\mathbf{b}$ constrain their values. Therefore, the following seem necessary from the onset in order for the bound to be valid.

- $S \succeq V^{-1}$ so that W is well-defined.
- If W is rank-deficient, then $\mathbf{b}$ lies on the hyperplane passing through $\mathbf{y}$ and in the column space of W.

However, further analysis will show these constraints to be unnecessary for h to be a lower bound. Consequently, we can view h as a function of the pair $(\mathbf{b},S)$, regardless of there being a pair $(\mathbf{a},W)$ mapping to $(\mathbf{b},S)$. Before proceeding to the formal theorem, a few notations are necessary. Let

$$g^c(q,\mathbf{b},S) \stackrel{\text{def}}{=} \exp\left[\mathbf{m}^{\mathrm{T}}\mathbf{e}^c + \frac{1}{2}(\mathbf{b}-\mathbf{e}^c)^{\mathrm{T}}S^{-1}(\mathbf{b}-\mathbf{e}^c)\right] \tag{16}$$

be a function of the mean $\mathbf{m}$ of distribution q and of $\mathbf{b}$ and $S \succ 0$. When the context is clear, we will suppress the parameters of g^c for conciseness. Let

$$\bar{g}^c \stackrel{\text{def}}{=} g^c / \textstyle\sum_{c'=1}^{C} g^{c'}, \qquad\qquad \bar{\mathbf{g}} \stackrel{\text{def}}{=} (\bar{g}^1, \ldots, \bar{g}^C)^{\mathrm{T}}, \tag{17}$$

and let $\bar{G}$ be the diagonal matrix with $\bar{\mathbf{g}}$ along its diagonal. We further define

$$A \stackrel{\text{def}}{=} \textstyle\sum_{c=1}^{C} \bar{g}^c (\mathbf{b} - \mathbf{e}^c)(\mathbf{b} - \mathbf{e}^c)^{\mathrm{T}} = \mathbf{b}\mathbf{b}^{\mathrm{T}} - \mathbf{b}\bar{\mathbf{g}}^{\mathrm{T}} - \bar{\mathbf{g}}\mathbf{b}^{\mathrm{T}} + \bar{G} \succeq 0. \tag{18}$$

Matrix A given above is a convex combination of C positive semi-definite matrices of ranks one, so A is positive semi-definite. Furthermore, $A \neq 0$. We will also suppress the dependency of A on $\mathbf{m}$, $\mathbf{b}$ and S for conciseness.

The lemmas necessary for the proof of the following theorem are in Appendix B.4.

Theorem 6 *Let S be a C-by-C positive definite matrix, and let $\mathbf{b} \in \mathbb{R}^C$. Let*

$$h(\mathbf{y}; q, \mathbf{b}, S) \stackrel{\text{def}}{=} \frac{C}{2} + \frac{1}{2}\log|SV| - \frac{1}{2}\operatorname{tr} SV + \mathbf{m}^{\mathrm{T}}\mathbf{y} - \log \sum_{c=1}^{C} g^c(q, \mathbf{b}, S) \tag{19}$$

be a function of $\mathbf{b}$ and S, where $g^c(q, \mathbf{b}, S)$ is given by (16)*. Then $\ell(\mathbf{y}; q) \geq h(\mathbf{y}; q, \mathbf{b}, S)$.*

Proof Let $(\mathbf{b}^*, \mathbf{S}^*) \stackrel{\text{def}}{=} \arg\max_{(\mathbf{b},S)} h(\mathbf{y}; q, \mathbf{b}, S)$. The joint concavity of h in $\mathbf{b}$ and S (Lemma 25) implies $h(\mathbf{y}; q, \mathbf{b}^*, S^*) > h(\mathbf{y}; q, \mathbf{b}, S)$ for any $\mathbf{b} \neq \mathbf{b}^*$ and $S \neq S^*$. Thus we only need to prove $\ell(\mathbf{y}; q) \geq h(\mathbf{y}; q, \mathbf{b}^*, S^*)$. Now, if there exists a pair $(\mathbf{a}^*, W^*)$ with $W^* \succeq 0$ such that $S^* = V^{-1} + W^*$ and $\mathbf{b}^* = W^*(\mathbf{m} - \mathbf{a}^*) + \mathbf{y}$, then the application of Lemma 5 completes the proof. To find such a pair, we first set S^* and W^* to the S^{fx} and the W^{fx} given by Lemma 28, then we show below that there exists an $\mathbf{a}^*$ under this setting.

Let $\bar{\mathbf{g}}^* \stackrel{\text{def}}{=} \bar{\mathbf{g}}(q, \mathbf{b}^*, S^*)$ and $\bar{G}^*$ be the diagonal matrix with $\bar{\mathbf{g}}^*$ along its diagonal. By Lemma 26, $\mathbf{b}^* = \bar{\mathbf{g}}^*$, so matrix A simplifies to A^* given by $A^* \stackrel{\text{def}}{=} \bar{G}^* - \bar{\mathbf{g}}^*(\bar{\mathbf{g}}^*)^{\mathrm{T}}$. Since $\bar{\mathbf{g}}^*$ is a probability vector, matrix A^* is the covariance matrix of a multinomial distribution. The entries in $\bar{\mathbf{g}}^*$ are non-zero, so matrix A^* is of rank $(C-1)$, and an eigenpair of A^* is $(0, \mathbf{1})$ (see Watson, 1996). In other words, $\text{null}(A^{**}) = \{\eta\mathbf{1} \mid \eta \in \mathbb{R}\}$. Using Lemma 28, we also have $\text{null}(W^*) = \{\eta\mathbf{1} \mid \eta \in \mathbb{R}\}$. Since $(\mathbf{b}^* - \mathbf{y})^{\mathrm{T}}\mathbf{1} = 1 - 1 = 0$, we have $(\mathbf{b}^* - \mathbf{y}) \notin \text{null}(W^*)$, unless $(\mathbf{b}^* - \mathbf{y}) = \mathbf{0}$. Equivalently, $(\mathbf{b}^* - \mathbf{y})$ is in the row space of W^*. Hence, there exists a vector $\mathbf{v}$ such that $W^*\mathbf{v} = \mathbf{b}^* - \mathbf{y}$. We let $\mathbf{a}^* \stackrel{\text{def}}{=} \mathbf{m} - \mathbf{v}$ to complete the proof. ■

There are two properties that W^* obeys: $\text{null}(W^*) = \{\eta\mathbf{1} \mid \eta \in \mathbb{R}\}$ and $W^* \succeq 0$. One parametrization of W that always satisfies these properties is

$$W \stackrel{\text{def}}{=} M - M\mathbf{1}\mathbf{1}^{\mathrm{T}}M / \mathbf{1}^{\mathrm{T}}M\mathbf{1}, \qquad\qquad \text{where } M \succ 0. \tag{20}$$

The proof for the null space is straightforward, while the proof for positive definiteness is an application of Theorem 7.7.7(a) by Horn and Johnson (1985). If M is a diagonal positive definite matrix, then the parametrization proposed by Seeger and Jordan (2004) is obtained. Further constraining the diagonal to sum to one gives the parametrization resultant from the Laplace approximation (Williams and Barber, 1998; Rasmussen and Williams, 2006). A diagonal M is appealing because it entails that W is the covariance of the multinomial or the Dirichlet distribution, which matches

the likelihood. However, our experience has shown that a diagonal M is far from optimum for our bounds. Therefore, we shall let W vary freely but be subjected directly to the two properties stated at the beginning of this paragraph. There are two reasons for the non-optimality. First, the variational prior $r(\mathbf{f})$ in Lemma 2 is a mixture of Gaussian distributions and not a Dirichlet distribution. Second, the use of Jensen's inequality in Lemma 5 weaken the interpretation of W as the covariance of the variational posterior $r(\mathbf{f}|\mathbf{y})$. Nonetheless, since the null space of W^* is the line $\{\eta\mathbf{1} \mid \eta \in \mathbb{R}\}$, the optimized variational posterior satisfies the invariance $r(\mathbf{f}|\mathbf{y}) = r(\mathbf{f}+\eta\mathbf{1}|\mathbf{y})$, $\eta \in \mathbb{R}$. This is a pleasant property because the likelihood satisfies the same invariance: $p(\mathbf{y}|\mathbf{f}) = p(\mathbf{y}|\mathbf{f}+\eta\mathbf{1})$.

The significance of Theorem 6 over Lemma 5 is in the practical aspects of variational inference:

1. Maximizing h with respect to V does not involve the function g^c.
2. A block coordinate approach to optimization can be used, since we can optimize with respect to V and to S alternately, without ensuring $S \succeq V^{-1}$ when optimizing for V.
3. The vector $\mathbf{y}$ of observed classifications does not appear in the definition of g^c given by Equation 16, in contrast to Equation 14.

Let us emphasis the second point listed above. In place of definitions (16) and (19) for functions g^c and h, suppose we had used

$$g^c(\mathbf{b}', S) \stackrel{\text{def}}{=} \exp\frac{1}{2}(\mathbf{b}' - \mathbf{e}^c)^{\mathrm{T}} S^{-1}(\mathbf{b}' - \mathbf{e}^c),$$

$$h(\mathbf{y}; q, \mathbf{b}, S) \stackrel{\text{def}}{=} \frac{C}{2} + \frac{1}{2}\log|SV| - \frac{1}{2}\operatorname{tr} S(V + \mathbf{m}\mathbf{m}^{\mathrm{T}}) + \mathbf{m}^{\mathrm{T}}(\mathbf{y} - \mathbf{b}') - \log\sum_{c=1}^{C} g^c(q, \mathbf{b}', S)$$

as functions of $\mathbf{b}'$ and $S \succ 0$. This is obtained from Lemma 2 by substituting in $S \stackrel{\text{def}}{=} V^{-1} + W$ and $\mathbf{b}' \stackrel{\text{def}}{=} -V^{-1}\mathbf{m} - W\mathbf{a} + \mathbf{y}$. This formulation of h is jointly concave in $\mathbf{b}'$ and S, so there should be no computation difficulties in optimization. Unfortunately, this formulation does not guarantee $S \succeq V^{-1}$ when the optimization is done without constraints. This is in contrast with the formulation in Theorem 6, for which validity is guaranteed by Lemma 28.

The bound h as defined in Theorem 6 is maximized by finding the stationary points with respect to variational parameters $\mathbf{b}$ and S. Computation can be reduced when the bound is relaxed through fixing or constraining these parameters. Two choices for S are convenient: I and V^{-1}. Fixing S to V^{-1} is expected to be a better choice since its optimal value is between V^{-1} and $V^{-1} + A$ (Lemma 27). This gives the relaxed bound

$$h(\mathbf{y}; q, \mathbf{b}, V^{-1}) = \mathbf{m}^{\mathrm{T}}\mathbf{y} - \log\sum_{c=1}^{C} \exp\left[\mathbf{m}^{\mathrm{T}}\mathbf{e}^c + \frac{1}{2}(\mathbf{b} - \mathbf{e}^c)^{\mathrm{T}} V (\mathbf{b} - \mathbf{e}^c)\right].$$

For the case where q is non-correlated Gaussians, that is, where V is a diagonal matrix, we obtain the bound that has been proposed for variational message passing (Knowles and Minka, 2011, Equation 12). We can also choose to fix $\mathbf{b}$ to $\mathbf{y}$, giving

$$h(\mathbf{y}; q, \mathbf{y}, V^{-1}) = \mathbf{m}^{\mathrm{T}}\mathbf{y} - \log\sum_{c=1}^{C} \exp\left[\mathbf{m}^{\mathrm{T}}\mathbf{e}^c + \frac{1}{2}(\mathbf{y} - \mathbf{e}^c)^{\mathrm{T}} V (\mathbf{y} - \mathbf{e}^c)\right].$$

This is the bound (9) obtained using Jensen's inequality directly. Setting $\mathbf{b}$ to $\mathbf{0}$ instead of $\mathbf{y}$ gives

$$h(\mathbf{y}; q, \mathbf{0}, V^{-1}) = \mathbf{m}^{\mathrm{T}}\mathbf{y} - \log\sum_{c=1}^{C} \exp\left[\mathbf{m}^{\mathrm{T}}\mathbf{e}^c + \frac{1}{2}(\mathbf{e}^c)^{\mathrm{T}} V \mathbf{e}^c\right],$$

which is the bound (10) also obtained using Jensen's inequality directly. Therefore, the bound $\max_{(\mathbf{b},S)} h(\mathbf{y};q,\mathbf{b},S)$ is provably at least as tight as the Jensen's inequality bounds. Other choices for S and $\mathbf{b}$ give different lower bounds on $\max_{(\mathbf{b},S)} h(\mathbf{y};q,\mathbf{b},S)$.

Thus far we have delved into lower bounds for $\ell(\mathbf{y};q)$ defined by Equation 8. Of independent interest is the following upper bound that is proved in Appendix B.5:

Lemma 7 $\ell(\mathbf{y};q) \leq \log p(\mathbf{y}|\mathbf{m}) \stackrel{\text{def}}{=} \mathbf{m}^{\mathrm{T}}\mathbf{y} - \log \sum_{c=1}^{C} \exp \mathbf{m}^{\mathrm{T}}\mathbf{e}^{c}$.

2.3.2 VARIATIONAL BOUNDS FOR MARGINAL LIKELIHOOD

To consolidate, the log marginal likelihood is lower bounded via the sequence

$$\log p(\mathbf{y}) \geq \log Z_B \geq \log Z_h \stackrel{\text{def}}{=} -\mathrm{KL}(q(\mathbf{f}|\mathbf{y}) \,\|\, p(\mathbf{f})) + \sum_{i=1}^{n} h(\mathbf{y}_i;q_i,\mathbf{b}_i,S_i),$$

where the datum subscript i is reintroduced. The aim is to optimize the last lower bound. Recall that $\mathbf{m}$ and V are the mean and covariance of the variational posterior $q(\mathbf{f}|\mathbf{y})$. Also recall that the prior distribution on $\mathbf{f}$ is given by the Gaussian process prior stated in Section 2.1, so $\mathbf{f}$ has zero mean and covariance $K \stackrel{\text{def}}{=} K^{\mathrm{x}} \otimes K^{\mathrm{c}}$, where K^{x} is the n-by-n matrix of covariances between the inputs $\mathbf{x}_1,\ldots,\mathbf{x}_n$. Using arguments similar to those used in proving Lemma 25, one can show that $\log Z_h$ is jointly concave in $\mathbf{m}$, V, $\{\mathbf{b}_i\}$ and $\{S_i\}$. We highlight this with the following proposition, where $\log Z_h$ is expressed explicitly in the variational parameters.

Proposition 8 *Let V be an nC-by-nC positive definite matrix and let $\mathbf{m} \in \mathbb{R}^{nC}$. For $i = 1,\ldots n$, let S_i be a C-by-C positive definite matrix and let $\mathbf{b}_i \in \mathbb{R}^{C}$. Let*

$$\begin{aligned}\log Z_h = nC + \frac{1}{2}\log|K^{-1}V| - \frac{1}{2}\operatorname{tr} K^{-1}V - \frac{1}{2}\mathbf{m}^{\mathrm{T}}K^{-1}\mathbf{m} + \mathbf{m}^{\mathrm{T}}\mathbf{y} \\ + \frac{1}{2}\sum_{i=1}^{n}\left(\log|S_iV_i| - \operatorname{tr} S_iV_i\right) - \sum_{i=1}^{n}\log\sum_{c=1}^{C}\exp\left[\mathbf{m}_i^{\mathrm{T}}\mathbf{e}^{c} + \frac{1}{2}(\mathbf{b}_i - \mathbf{e}^{c})^{\mathrm{T}}S_i^{-1}(\mathbf{b}_i - \mathbf{e}^{c})\right], \quad (21)\end{aligned}$$

where V_i is the ith C-by-C diagonal block of V, and $\mathbf{m}_i$ is the ith C-vector of $\mathbf{m}$. Then $\log Z_h$ is jointly concave in $\mathbf{m}$, V, $\{\mathbf{b}_i\}$ and $\{S_i\}$, and $\log p(\mathbf{y}) \geq \log Z_h$.

Suitable choices of the variational parameters leads to the following two theorems that are proved in Appendix B.6.

Theorem 9 *For a multinomial logit Gaussian process model where the latent process has zero mean and the covariance function induces the Gram matrix K, the average log-marginal-likelihood satisfies*

$$\begin{aligned}\frac{1}{n}\log p(\mathbf{y}) \geq\ & \frac{C}{2} + \frac{C}{2}\log\sigma_{\mathrm{v}}^2 - \frac{1}{2n}\log|K| - \frac{\sigma_{\mathrm{v}}^2}{2n}\operatorname{tr} K^{-1} \\ & - \frac{C-1}{2}\left[2\sqrt{\frac{\sigma_{\mathrm{v}}^2}{C} + \frac{1}{4}} - \log\left(\sqrt{\frac{\sigma_{\mathrm{v}}^2}{C} + \frac{1}{4}} + \frac{1}{2}\right) - 1\right] - \log C \\ >\ & \frac{C}{2} + \frac{C}{2}\log\sigma_{\mathrm{v}}^2 - \frac{1}{2n}\log|K| - \frac{\sigma_{\mathrm{v}}^2}{2n}\operatorname{tr} K^{-1} - \frac{\sigma_{\mathrm{v}}^2}{2} - \log C\end{aligned}$$

for every $\sigma_{\mathrm{v}}^2 > 0$.

Theorem 10 *For a multinomial logit Gaussian process model where the latent process has zero mean, the covariance function is $k((\mathbf{x},c),(\mathbf{x}',c')) = \sigma^2\delta(c,c')k^{\mathrm{x}}(\mathbf{x},\mathbf{x}')$ and k^{x} is a correlation function, that is, $k^{\mathrm{x}}(\mathbf{x},\mathbf{x}) = 1$, the average log-marginal-likelihood satisfies*

$$\begin{aligned}\frac{1}{n}\log p(\mathbf{y}) &\geq -\frac{C-1}{2}\left[2\sqrt{\frac{\sigma^2}{C}+\frac{1}{4}} - \log\left(\sqrt{\frac{\sigma^2}{C}+\frac{1}{4}}+\frac{1}{2}\right) - 1\right] - \log C \\ &> -\sigma^2/2 - \log C.\end{aligned}$$

The bounds in the theorems do not dependent on the observed classes $\mathbf{y}$ because they have been "zeroed-out" by setting $\mathbf{m} = \mathbf{0}$. For the setting in Theorem 10, the lower bound in Theorem 10 is always tighter than that in Theorem 9 because the first four terms within the latter is the negative of a Kullback-Leibler divergence, which is always less than zero. One may imagine that this bound is rather loose. However, we will show in experiments in Section 9.1 that even this is better than the optimized variational mean-field lower bound (Girolami and Rogers, 2006).

Remark 11 *Theorem 10 is consistent with and generalizes the calculations previously obtained for binary classification and in certain limits of the length-scales of the model (Nickisch and Rasmussen, 2008, Appendix B). Our result is also more general because it includes the latent scale σ^2 of the model.*

2.3.3 PREDICTIVE DENSITY: APPROXIMATION AND BOUNDS

According to the Gaussian process prior model specified in Section 2.1, the C latent function values $\mathbf{f}_*$ of a test input $\mathbf{x}_*$ and the latent function values of the n observed data have prior

$$\begin{pmatrix}\mathbf{f}\\ \mathbf{f}_*\end{pmatrix} \sim \mathcal{N}\left(\mathbf{0}, \begin{pmatrix}K & K_* \\ K_*^{\mathrm{T}} & K_{**}\end{pmatrix}\right),$$

where $K_* \stackrel{\text{def}}{=} \mathbf{k}_*^{\mathrm{x}} \otimes K^{\mathrm{c}}$, $K_{**} \stackrel{\text{def}}{=} k^{\mathrm{x}}(\mathbf{x}_*,\mathbf{x}_*)K^{\mathrm{c}}$, and $\mathbf{k}_*^{\mathrm{x}}$ is the vector of covariances between the observed inputs X and the test input $\mathbf{x}_*$. After the variational posterior $q(\mathbf{f}|\mathbf{y}) = \mathcal{N}(\mathbf{f}|\mathbf{m},V)$ has been obtained by maximizing the lower bound $\log Z_h$ in Proposition 8, we can obtain the approximate posterior at the test input $\mathbf{x}_*$:

$$q(\mathbf{f}_*|\mathbf{y}) \stackrel{\text{def}}{=} \int p(\mathbf{f}_*|\mathbf{f})q(\mathbf{f}|\mathbf{y})\mathrm{d}\mathbf{f} = \mathcal{N}(\mathbf{f}_*|\mathbf{m}_*,V_*),$$

where $\mathbf{m}_* \stackrel{\text{def}}{=} K_*^{\mathrm{T}}K^{-1}\mathbf{m}$ and $V_* \stackrel{\text{def}}{=} K_{**} - K_*^{\mathrm{T}}K^{-1}K_* + K_*^{\mathrm{T}}K^{-1}VK^{-1}K_*$.

The approximation to the posterior predictive density of $\mathbf{y}_*$ at $\mathbf{x}_*$ is

$$\begin{aligned}\log p(y_*^c = 1|\mathbf{y}) \approx \log q(y_*^c = 1|\mathbf{y}) &\stackrel{\text{def}}{=} \log \int p(y_*^c = 1|\mathbf{f}_*)\, q(\mathbf{f}_*|\mathbf{y})\, \mathrm{d}\mathbf{f}_* && (22)\\ &\geq \ell_*(y_*^c = 1; q) \\ &\geq \max_{\mathbf{b}_*,S_*} h(\mathbf{e}^c; q_*, \mathbf{b}_*, S_*), && (23)\end{aligned}$$

where $\ell_*(y_*^c = 1; q) = \int q(\mathbf{f}_*|\mathbf{y}) \log p(y_*^c = 1|\mathbf{f}_*)\, \mathrm{d}\mathbf{f}_*$, and q_* in the last expression refers to $q(\mathbf{f}_*|\mathbf{y})$. Expanding h using definitions (16) and (19) gives

$$\log p(y_*^c = 1|\mathbf{y}) \gtrsim \frac{C}{2} + \mathbf{m}_*^{\mathrm{T}}\mathbf{e}^c + \max_{\mathbf{b}_*,S_*}\left(\frac{1}{2}\log|S_*V_*| - \frac{1}{2}\operatorname{tr}S_*V_* - \log\sum_{c'=1}^{C} g^{c'}(q_*,\mathbf{b}_*,S_*)\right), \quad (24)$$

where the probed class $\mathbf{e}^c$ is used only outside the max operator. Hence maximization needs to be only done once instead of C times. Moreover, if one is interested only in the classification decision, then one may simply compare the *re-normalized probabilities*

$$\tilde{p}(y_*^c = 1|\mathbf{y}) = p(y_*^c = 1|\mathbf{m}_*) \stackrel{\text{def}}{=} \frac{\exp(m_*^c)}{\sum_{c'=1}^{C} \exp(m_*^{c'})}. \tag{25}$$

In this case, no maximization is required, and class prediction is faster. The faster prediction is possible because we have used the lower bound (23) for making classification decisions. These classification decisions do not match those given by $q(y_*^c|\mathbf{y})$ in general (Rasmussen and Williams, 2006, Section 3.5 and Exercise 3.10.3). In addition to the normalization across the C classes, the predictive probability $\tilde{p}(y_*^c = 1|\mathbf{y})$ is also an upper bound on $\exp \ell_*(y_*^c = 1; q)$ because of Lemma 7.

The relation in Equation 24 is an approximate inequality ($\gtrsim$) instead of a proper inequality ($\geq$) due to the approximation to $\log q(y_*^c = 1|\mathbf{y})$ in Equation 22. As far as we are aware, this approximation is currently used throughout the literature for Gaussian process classification (Rasmussen and Williams 2006, Equations 3.25, 3.40 & 3.41 and 3.62; Nickisch and Rasmussen 2008, Equation 16). In order to obtain a proper inequality, we will show that the Kullback-Leibler divergence from the approximate posterior to the true posterior has to be accounted for.

First, we generalize and consider a set of n_* test inputs $X_* \stackrel{\text{def}}{=} \{\mathbf{x}_{*1}, \ldots, \mathbf{x}_{*n_*}\}$. The following theorem, which give proper lower bounds, is proved in Appendix B.7.

Theorem 12 *The log joint predictive probability for $\mathbf{x}_{*j}$ to be in class c_j ($j = 1 \ldots n_*$) has lower bounds*

$$\begin{aligned}
\log p(\{y_{*j}^{c_j} = 1\}_{j=1}^{n_*}|\mathbf{y}) &\geq \sum_{j=1}^{n_*} \int q(\mathbf{f}_{*j}|\mathbf{y}) \log p(y_{*j}^{c_j} = 1|\mathbf{f}_{*j}) \, \mathrm{d}\mathbf{f}_{*j} - \mathrm{KL}(q(\mathbf{f}|\mathbf{y}) \| p(\mathbf{f}|\mathbf{y})) \\
&\geq \sum_{j=1}^{n_*} \max_{\mathbf{b}_{*j}, S_{*j}} h(\mathbf{e}^{c_j}; q_{*j}, \mathbf{b}_{*j}, S_{*j}) + \log Z_B - \log p(\mathbf{y}) \\
&\geq \sum_{j=1}^{n_*} \max_{\mathbf{b}_{*j}, S_{*j}} h(\mathbf{e}^{c_j}; q_{*j}, \mathbf{b}_{*j}, S_{*j}) + \log Z_h - \log p(\mathbf{y}).
\end{aligned}$$

In the first bound, the computation of the Kullback-Leibler divergence is intractable, but it is precisely this quantity that we have sought to minimize in the beginning, in Section 2.3. This implies that this divergence is a correct quantity to minimize in order to tighten the lower bound on the predictive probabilities. For one test input $\mathbf{x}_*$,

$$\log p(y_*^c = 1|\mathbf{y}) \geq \max_{\mathbf{b}_*, S_*} h(\mathbf{e}^c; q_*, \mathbf{b}_*, S_*) + \log Z_h - \log p(\mathbf{y}).$$

Because $\log Z_B$, $\log Z_h$ and $\log p(\mathbf{y})$ are independent of the probed class $\mathbf{e}^c$ at $\mathbf{x}_*$, the classification decision and the re-normalized probabilities (25) are also based on a true lower bound to the predictive probability.

Dividing the last bound in Theorem 12 by n_* gives

$$\frac{1}{n_*} \log p(\{y_{*j}^{c_j} = 1\}_{j=1}^{n_*}|\mathbf{y}) \geq \frac{1}{n_*} \sum_{j=1}^{n_*} \max_{\mathbf{b}_{*j}, S_{*j}} h(\mathbf{e}^{c_j}; q_{*j}, \mathbf{b}_{*j}, S_{*j}) + \frac{1}{n_*} \Big[\log Z_h - \log p(\mathbf{y}) \Big].$$

The term $\log Z_h - \log p(\mathbf{y})$ is a constant independent of n_*, so the last term diminishes when n_* is large. In contrast the other two terms on either side of the inequality remain significant because each of them is the sum of n_* summands. Hence, for large n_*, the last term can practically be ignored to give a computable lower bound on the average log predictive probability.

3. Variational Bound Optimization

To optimize the lower bound $\log Z_h$ during learning, we choose a block coordinate approach, where we optimize with respect to the variational parameters $\{\mathbf{b}_i\}$, $\{S_i\}$, $\mathbf{m}$ and V in turn. For prediction, we only need to optimize h with respect to the variational parameters $\mathbf{b}_*$ and S_* for the test input $\mathbf{x}_*$.

3.1 Parameter $\mathbf{b}_i$

Parameters $\mathbf{b}_i$ and S_i are contained within $h(\mathbf{y}_i; q_i, \mathbf{b}_i, S_i)$, so we only need to consider this function. For clarity, we suppress the datum subscript i and the parameters for h and g^c. The partial gradient with respect to $\mathbf{b}$ is $-S^{-1}(\mathbf{b} - \bar{\mathbf{g}})$, where $\bar{\mathbf{g}}$ is defined by Equation 17. Setting the gradient to zero gives the fixed-point update $\mathbf{b}^{\mathrm{fx}} = \bar{\mathbf{g}}$, where $\bar{\mathbf{g}}$ is evaluated at the previous value of $\mathbf{b}$. This says that the optimal value $\mathbf{b}^*$ lies on the C-simplex, so a sensible initialization for $\mathbf{b}$ is a point therein. When the fixed-point update does not improve the lower bound h, we use to the Newton-Raphson update, which incorporates the Hessian

$$\frac{\partial^2 h}{\partial \mathbf{b}\, \partial \mathbf{b}^{\mathrm{T}}} = -S^{-1} - S^{-1}\left(\bar{G} - \bar{\mathbf{g}}\bar{\mathbf{g}}^{\mathrm{T}}\right) S^{-1},$$

where $\bar{G}$ is the diagonal matrix with $\bar{\mathbf{g}}$ along its diagonal. The Hessian is negative semi-definite, which is another proof that h is a concave function of $\mathbf{b}$; see Lemma 25. The update is

$$\mathbf{b}^{\mathrm{NR}} = \mathbf{b} - \eta \left(\frac{\partial^2 h}{\partial \mathbf{b}\, \partial \mathbf{b}^{\mathrm{T}}}\right)^{-1} \frac{\partial h}{\partial \mathbf{b}} = \mathbf{b} - \eta \left[I + \left(\bar{G} - \bar{\mathbf{g}}\bar{\mathbf{g}}^{\mathrm{T}}\right) S^{-1}\right]^{-1} (\mathbf{b} - \bar{\mathbf{g}}),$$

where $\eta = 1$. This update may fail due to numerical errors in areas of high curvatures. In such a case, we search for an optimal $\eta \in [0,1]$ using the false position method.

3.2 Parameter S_i

Similar to $\mathbf{b}_i$, only $h(\mathbf{y}_i; q_i, \mathbf{b}_i, S_i)$ needs to be considered for S_i, and the datum subscript i is suppressed here. The partial gradient with respect to S is given by Equation 59, from which we obtain the implicit Equation 60. Let V factorizes to LL^{T}, where L is non-singular since $V \succ 0$. Using A given by (18) at the current value of S, a fixed-point update for S is

$$S^{\mathrm{fx}} = L^{-\mathrm{T}} P \tilde{\Lambda} P^{\mathrm{T}} L^{-1}, \qquad \tilde{\Lambda} \stackrel{\mathrm{def}}{=} (\Lambda + I/4)^{1/2} + I/2,$$

where $P\Lambda P^{\mathrm{T}}$ is the eigen-decomposition of $L^{\mathrm{T}} A L$; see the proof of Lemma 28 in Appendix B.4.

The fixed-point update S^{fx} may fail to improve the bound. We may fall-back on the Newton-Raphson update for S that uses gradient (59) and a C^2-by-C^2 Hessian matrix. However, this can be rather involved since it needs to ensure that S stays positive definite. An alternative, which we prefer, is to perform line-search in a direction that guarantees positive definiteness. To this end, let $S = S^{\mathrm{cc}} \stackrel{\mathrm{def}}{=} (1-\eta)S + \eta S^{\mathrm{fx}}$, and we search for a $\eta \in [0,1]$ that optimizes the bound using the false position method. Appendix C.1 gives the details.

3.3 Parameter m, and Joint Optimization with b

We now optimize the bound $\log Z_h$ with respect to $\mathbf{m}$. Here, the datum subscript i is reintroduced. Let $\mathbf{y}$ (resp. $\bar{\mathbf{g}}$) be the nC-vector obtained by stacking the $\mathbf{y}_i$s (resp. $\bar{\mathbf{g}}_i$s). Let $\bar{G}$ be the nC-by-nC diagonal matrix with $\bar{\mathbf{g}}$ along its diagonal, and let $\tilde{G}$ be the nC-by-nC block diagonal matrix with $\bar{\mathbf{g}}_i\bar{\mathbf{g}}_i^{\mathrm{T}}$ as the ith block. The gradient and Hessian with respect to $\mathbf{m}$ are

$$\frac{\partial \log Z_h}{\partial \mathbf{m}} = -K^{-1}\mathbf{m} + \mathbf{y} - \bar{\mathbf{g}}, \qquad \frac{\partial^2 \log Z_h}{\partial \mathbf{m} \partial \mathbf{m}^{\mathrm{T}}} = -K^{-1} - \left(\bar{G} - \tilde{G}\right). \tag{26}$$

The Hessian is negative semi-definite; this is another proof that $\log Z_h$ is concave in $\mathbf{m}$. The fixed-point update $\mathbf{m}^{\mathrm{fx}} = K(\mathbf{y} - \bar{\mathbf{g}})$ can be obtained by setting the gradient to zero. This update may fail to give a better bound. One remedy is to use the Newton-Raphson update. Alternatively the concavity in $\mathbf{m}$ can be exploited to optimize with respect to $\eta \in [0,1]$ in $\mathbf{m}^{\mathrm{cc}} \stackrel{\mathrm{def}}{=} (1-\eta)\mathbf{m} + \eta\mathbf{m}^{\mathrm{fx}}$, such as is done for the parameters S_is in the previous section. Here we will give a combined update for $\mathbf{m}$ and the $\mathbf{b}_i$s that can be used during variational learning. This update avoids inverting K, which can be ill-conditioned.

The gradient in (26) implies the self-consistent equation $\mathbf{m}^* = K(\mathbf{y} - \bar{\mathbf{g}}^*)$ at the maximum, where $\bar{\mathbf{g}}^*$ is $\bar{\mathbf{g}}$ evaluated the the optimum parameters. From Lemma 26, another self-consistent equation is $\mathbf{b}^* = \bar{\mathbf{g}}^*$, where the nC-vector $\mathbf{b}^*$ is obtained by stacking all the $\mathbf{b}_i$s. Combining these two equations gives $\mathbf{m}^* = K(\mathbf{y} - \mathbf{b}^*)$, which is a bijection between $\mathbf{b}^*$ and $\mathbf{m}^*$ if K has full rank. For the sparse approximation that will be introduced later, K will be replaced by the "fat" matrix K_{f}, which is column-rank deficient. There, the mapping from $\mathbf{b}^*$ to $\mathbf{m}^*$ becomes many-to-one. With this in mind, instead of letting $\mathbf{m}$ be a variational parameter, we fix it to be a function of $\mathbf{b}$, that is,

$$\mathbf{m} = K(\mathbf{y} - \mathbf{b}), \tag{27}$$

and we optimize over $\mathbf{b}$ instead. The details are in Appendix C.2.

This joint update for $\mathbf{m}$ and the $\mathbf{b}_i$s can be used for variational learning. This, however, does not make the update in Section 3.1 redundant: that is still required during approximate prediction, where the $\mathbf{b}_*$ for the test input $\mathbf{x}_*$ still needs to be optimized over even though $\mathbf{m}$ is fixed after learning.

3.4 Parameter V

For the gradient with respect to V we have

$$\frac{\partial h_i}{\partial V_i} = \frac{1}{2}V_i^{-\mathrm{T}} - \frac{1}{2}S_i^{\mathrm{T}} = -\frac{1}{2}W_i^{\mathrm{T}}, \qquad \frac{\partial \log Z_h}{\partial V} = \frac{1}{2}V^{-1} - \frac{1}{2}K^{-1} - \frac{1}{2}W^{-1},$$

where $W_i \stackrel{\mathrm{def}}{=} S_i - V_i^{-1}$, and W is the block diagonal matrix of the W_is. Here, function h_i is regarded as parameterized by S_i (as in Theorem 6) rather than by W_i (as in Lemma 25). Using gradient $\partial \log Z_h / \partial V$ directly as a search direction to update V is undesirable for two reasons. First, it may not preserve the positive-definiteness of V. Second, it requires K to be inverted, and this can cause numerical issues for some covariance functions such as the squared exponential covariance function, which has exponentially vanishing eigenvalues.

We propose to let V follow the trajectory along a modified gradient, where W is regarded fixed instead of depending on V. To explain, we recall that $\log Z_h \stackrel{\mathrm{def}}{=} -\mathrm{KL}(q(\mathbf{f}|\mathbf{y}) \| p(\mathbf{f})) + h$, where

$h \stackrel{\text{def}}{=} \sum_{i=1}^{n} h_i$ is the sum of functions each concave in V. The modified gradient holds the gradient contribution from h constant at the value at the initial V while the gradient contribution from the Kullback-Leibler divergence varies along the trajectory. We follow the trajectory until the modified gradient is zero. Let this point be V^{fx}. Then

$$\frac{1}{2}(V^{\text{fx}})^{-1} - \frac{1}{2}K^{-1} - \frac{1}{2}W^{-1} = 0, \qquad \text{or} \qquad V^{\text{fx}} = \left(K^{-1} + W\right)^{-1}. \tag{28}$$

The equation on the right can be used as a naïve fix-point update.

The trajectory following this modified gradient will diverge from the trajectory following the exact gradient, so there is no guarantee that V^{fx} gives an improvement over V. To remedy, we follow the strategy used for updating S: we use $V^{\text{cc}} \stackrel{\text{def}}{=} (1-\eta)V + \eta V^{\text{fx}}$ and optimize with respect to $\eta \in [0,1]$. Matrix V^{cc} is guaranteed to be positive definite, since it is a convex combination of two positive definite matrices. Details are in Appendix C.3.

4. Sparse Approximation

The variational approach for learning multinomial logit Gaussian processes discussed in the previous sections has transformed an intractable integral problem into a tractable optimization problem. However, the variational approach is still expensive for large data sets because the computational complexity of the matrix operations is $O(C^3n^3)$, where n is the size of the observed set and C is the number of classes. One popular approach to reduce the complexity is to use sparse approximations: only $s \ll n$ data inputs or sites are chosen to be used within a complete but smaller Gaussian process model, and information for the rest of the observations are induced via these s sites. Each of the s data sites is called an *inducing site*, and the associated random variables $\mathbf{z}$ are called the *inducing variables*. We use the term *inducing set* to mean either the inducing sites or the inducing variables or both. The selection of the inducing set is seen as a model selection problem (Snelson and Ghahramani, 2006; Titsias, 2009a) and will be addressed in Section 6.1.

We seek a sparse approximation will lead to a lower bound on the true marginal likelihood. This approach has been proposed for Gaussian process regression (Titsias, 2009a), and it will facilitate the search for the inducing set later. Recall that the inducing variables at the s inducing sites are denoted by $\mathbf{z} \in \mathbb{R}^s$. We retain $\mathbf{f}$ for the nC latent function values associated with the n observed data $(X, \mathbf{y})$. In general, the inducing variables $\mathbf{z}$ need not be chosen from the latent function values $\mathbf{f}$, so our presentation will treat them as distinct.

The Gaussian prior over the latent values $\mathbf{f}$ is extended to the inducing variables $\mathbf{z}$ to give a Gaussian joint prior $p(\mathbf{f}, \mathbf{z})$. Let $p(\mathbf{f}, \mathbf{z}|\mathbf{y})$ be the true joint posterior of the latent and inducing variables is given the observed data. This posterior is non-Gaussian because of the multinomial logistic likelihood function, and it is intractable to calculate this posterior as is in the non-sparse case. The approximation $q(\mathbf{f}, \mathbf{z}|\mathbf{y})$ to the exact posterior is performed in two steps. In the first step, we let $q(\mathbf{f}, \mathbf{z}|\mathbf{y})$ be a Gaussian distribution. This is a natural choice which follows from the non-sparse case. In the second step, we use the factorization

$$q(\mathbf{f}, \mathbf{z}|\mathbf{y}) \stackrel{\text{def}}{=} p(\mathbf{f}|\mathbf{z})\, q(\mathbf{z}|\mathbf{y}), \tag{29}$$

where $p(\mathbf{f}|\mathbf{z})$ is the marginal of $\mathbf{f}$ from the prior $p(\mathbf{f}, \mathbf{z})$. The same approximation has been used in the sparse approximation for regression (Titsias, 2009a, paragraph before Equation 7). This

approximation makes clear the role of inducing variables $\mathbf{z}$ as the conduit of information from $\mathbf{y}$ to $\mathbf{f}$. Under this approximate posterior, we have the bound

$$\log p(\mathbf{y}) \geq \log \tilde{Z}_B = -\mathrm{KL}(q(\mathbf{z}|\mathbf{y}) \| p(\mathbf{z})) + \sum_{i=1}^{n} \ell_i(\mathbf{y}_i; q),$$

where $\ell_i(\mathbf{y}_i; q) \stackrel{\text{def}}{=} \int q(\mathbf{f}_i|\mathbf{y}) \log p(\mathbf{y}_i|\mathbf{f}_i)\, \mathrm{d}\mathbf{f}_i$, and $q(\mathbf{f}_i|\mathbf{y})$ is the marginal distribution of $\mathbf{f}_i$ from the joint distribution $q(\mathbf{f}, \mathbf{z}|\mathbf{y})$; see Appendix B.1. The reader may wish to compare with Equations 3, 4 and 5 for the non-sparse variational approximation.

Similar to the dissection of $\log Z_B$ after Equation 3, the Kullback-Leibler divergence component of $\log \tilde{Z}_B$ can be interpreted as the regularizing factor for the approximate posterior $q(\mathbf{z}|\mathbf{y})$, while the expected log-likelihood can be interpreted as the data fit component. This dissection provides three insights into the sparse formulation. First, the specification of $p(\mathbf{z})$ is part of the model and not part of the approximation—the approximation step is in the factorization (29). Second, the Kullback-Leibler divergence term involves only the inducing variables $\mathbf{z}$ and *not* the latent variables $\mathbf{f}$. Hence, the regularizing is on the approximate posterior of $\mathbf{z}$ and not on that of $\mathbf{f}$. Third, the involvement of $\mathbf{f}$ is confined to the data fit component in a two step process: generating $\mathbf{f}$ from $\mathbf{z}$ and then generating $\mathbf{y}$ from $\mathbf{f}$.

Applying Theorem 6 on the ℓ_is gives

$$\log \tilde{Z}_B \geq \log \tilde{Z}_h \stackrel{\text{def}}{=} -\mathrm{KL}(q(\mathbf{z}|\mathbf{y}) \| p(\mathbf{z})) + \sum_{i=1}^{n} h(\mathbf{y}_i; q_i, \mathbf{b}_i, S_i), \tag{30}$$

where h is defined by Equation 19, and the q_i within h is the marginal distribution $q(\mathbf{f}_i|\mathbf{y})$.

We now examine $\log \tilde{Z}_h$ using the parameters of the distributions. Let the joint prior be

$$p\left(\begin{pmatrix}\mathbf{z}\\ \mathbf{f}\end{pmatrix}\right) = \mathcal{N}\left(\mathbf{0}, \begin{pmatrix} K & K_{\mathrm{f}} \\ K_{\mathrm{f}}^{\mathrm{T}} & K_{\mathrm{ff}}\end{pmatrix}\right).$$

One can generalize the prior for $\mathbf{z}$ to have a non-zero mean, but the above suffice for our purpose and simplifies the presentation. In the case where an inducing variable z_i coincide with a latent variable f_j^c, we can "tie" them by setting their prior correlation to one. The marginal distribution $p(\mathbf{f})$ is the Gaussian process prior of the model, but we are now using K_{ff} to denote the covariance induced by K^{c} and $k^{\mathrm{x}}(\cdot,\cdot)$ while reserving K for the covariance of $\mathbf{z}$. This facilitates comparison to the expressions for the non-sparse approximation.

For the approximate posterior, let $q(\mathbf{z}|\mathbf{y}) = \mathcal{N}(\mathbf{m}, V)$, so $\mathbf{m}$ and V are the variational parameters of the approximation. Then $q(\mathbf{f}|\mathbf{y})$ is Gaussian with mean and covariance

$$\mathbf{m}_{\mathrm{f}} = K_{\mathrm{f}}^{\mathrm{T}} K^{-1} \mathbf{m}, \qquad V_{\mathrm{f}} = K_{\mathrm{ff}} - K_{\mathrm{f}}^{\mathrm{T}} K^{-1} K_{\mathrm{f}} + K_{\mathrm{f}}^{\mathrm{T}} K^{-1} V K^{-1} K_{\mathrm{f}}. \tag{31}$$

Therefore, the lower bound on the log marginal likelihood is

$$\log \tilde{Z}_h = \frac{s}{2} + \frac{1}{2}\log|K^{-1}V| - \frac{1}{2}\operatorname{tr} K^{-1}V - \frac{1}{2}\mathbf{m}^{\mathrm{T}} K^{-1}\mathbf{m} + \frac{nC}{2} + \mathbf{m}_{\mathrm{f}}^{\mathrm{T}}\mathbf{y} + \frac{1}{2}\sum_{i=1}^{n}\left(\log|S_i V_{\mathrm{f}i}| - \operatorname{tr} S_i V_{\mathrm{f}i}\right) - \sum_{i=1}^{n}\log\sum_{c=1}^{C} g_i^c, \tag{32}$$

where $V_{\mathrm{f}i}$ is the ith diagonal C-by-C block matrix of V_{f} and

$$g_i^c \stackrel{\text{def}}{=} \exp\left[\mathbf{m}_{\mathrm{f}i}^{\mathrm{T}}\mathbf{e}^c + \frac{1}{2}(\mathbf{b}_i - \mathbf{e}^c)^{\mathrm{T}} S_i^{-1}(\mathbf{b}_i - \mathbf{e}^c)\right].$$

Remark 13 *It is not necessary for* $\mathbf{z}$ *to be drawn from the latent Gaussian process prior directly. Therefore, the covariance* K *of* $\mathbf{z}$ *need not be given by the covariance functions* K^{c} *and* $k^{\mathrm{x}}(\cdot,\cdot)$ *of the latent Gaussian process model. In fact,* $\mathbf{z}$ *can be any linear functional of draws from the latent Gaussian process prior (see, for example, Titsias, 2009b, Section 6). For example, it is almost always necessary to set* $\mathbf{z} = \mathbf{z}' + \boldsymbol{\epsilon}$*, where* $\boldsymbol{\epsilon}$ *is the isotropic noise, so that the matrix inversion of* K *is not ill-conditioned. This matrix inversion cannot be avoided (without involving* $O(s^3)$ *computations) in the sparse approximation because of the need to compute* $K_{\mathrm{f}}^{\mathrm{T}} K^{-1} K_{\mathrm{f}}$*, which is the Nyström approximation to* K_{ff} *if* $\mathbf{z}' \equiv \mathbf{f}$ *(Williams and Seeger, 2001). One attractiveness of having a lower bound associated to the sparse approximation is that the noise variance of* $\boldsymbol{\epsilon}$ *can be treated as a lower bounded variational parameter to be optimized (Titsias, 2009b, Section 6).*

Remark 14 *The inducing variables* $\mathbf{z}$ *are associated with the latent values* $\mathbf{f}$ *and not with the observed data* $(X, \mathbf{y})$*. Therefore, it is not necessary to choose all the latent values* $f_i^1, \ldots, f_i^C$ *for any datum* $\mathbf{x}_i$*. One may choose the inducing sites to include, say,* f_i^c *for datum* $\mathbf{x}_i$ *and* $f_j^{c'}$ *for datum* $\mathbf{x}_j$*, and to exclude* $f_i^{c'}$ *for datum* $\mathbf{x}_i$ *and* f_j^c *for datum* $\mathbf{x}_j$*. This flexibility requires additional bookkeeping in the implementation.*

4.1 Comparing Sparse and Non-sparse Approximations

We can relate the bounds for the non-sparse and sparse approximations:

Theorem 15 *Let*

$$\log Z_B^* \stackrel{\text{def}}{=} \max_{q(\mathbf{f}|\mathbf{y})} \log Z_B, \quad \log Z_h^* \stackrel{\text{def}}{=} \max_{q(\mathbf{f}|\mathbf{y})} \log Z_h, \quad \log \tilde{Z}_B^* \stackrel{\text{def}}{=} \max_{q(\mathbf{z}|\mathbf{y})} \log \tilde{Z}_B, \quad \log \tilde{Z}_h^* \stackrel{\text{def}}{=} \max_{q(\mathbf{z}|\mathbf{y})} \log \tilde{Z}_h,$$

where the sparse bounds are for any inducing set. Then $\log Z_B^* \geq \log \tilde{Z}_B^*$ *and* $\log Z_h^* \geq \log \tilde{Z}_h^*$.

The proof for the first inequality is given in Appendix B.1.1, while the second inequality is a consequence of Proposition 17 derived in Section 6.1. Though intuitive, the second inequality is not obvious because of the additional maximization over the variational parameters $\{\mathbf{b}_i\}$ and $\{S_i\}$. The presented sparse approximation is optimal: if $\mathbf{z} \equiv \mathbf{f}$, then $\log \tilde{Z}_B = \log Z_B$ and $\log \tilde{Z}_h = \log Z_h$, and the sparse approximation becomes the non-sparse approximation.

4.2 Optimization

The sparse approximation requires optimizing $\log \tilde{Z}_h$ with respect to the variational parameters: mean $\mathbf{m}$ and covariance V of the inducing variables; and $\{\mathbf{b}_i\}$ and $\{S_i\}$ for the lower bound on the expected log-likelihood. The optimization with respect to $\{S_i\}$ is the same as that for the non-sparse approximation, but using $\mathbf{m}_{\mathrm{f}}$ and V_{f} for the mean and covariance of the latent variables. For $\{\mathbf{b}_i\}$ and $\mathbf{m}$, a joint optimization akin to that for the non-sparse approximation described in Section 3.3 can be used. Let $\mathbf{b}$ be the nC-vector that is the stacking of the $\mathbf{b}_i$s. At the saddle point with respect to $\{\mathbf{b}_i\}$ and $\mathbf{m}$, we have the self-consistent equations $\mathbf{b}_i^* = \bar{\mathbf{g}}_i^*$ and $\mathbf{m}^* = K_{\mathrm{f}}(\mathbf{y} - \bar{\mathbf{g}}^*)$, from which is obtained

the linear mapping $\mathbf{m}^* = K_{\mathrm{f}}(\mathbf{y} - \mathbf{b}^*)$. For sparse approximation, matrix K_{f} has more columns than rows (that is, a "fat" matrix), so the linear mapping from $\mathbf{b}^*$ to $\mathbf{m}^*$ is many to one. Hence we substitute the constraint $\mathbf{m} = K_{\mathrm{f}}(\mathbf{y} - \mathbf{b})$ into the bound and optimize over $\mathbf{b}$. The optimization is similar to non-sparse case, and is detailed in Appendix C.2.1.

For V, the approach in Section 3.4 for the non-sparse approximation is followed. The gradients with respect to V are

$$\frac{\partial h_i}{\partial V} = \frac{1}{2}K^{-1}K_{\mathrm{f}i}\left(V_{\mathrm{f}i}^{-1} - S_i\right)K_{\mathrm{f}i}^{\mathrm{T}}K^{-1} \qquad \frac{\partial \log \tilde{Z}_h}{\partial V} = \frac{1}{2}V^{-1} - \frac{1}{2}K^{-1} - \frac{1}{2}K^{-1}K_{\mathrm{f}}W_{\mathrm{f}}K_{\mathrm{f}}^{\mathrm{T}}K^{-1}$$
$$= -\frac{1}{2}K^{-1}K_{\mathrm{f}i}W_{\mathrm{f}i}K_{\mathrm{f}i}^{\mathrm{T}}K^{-1}; \qquad = \frac{1}{2}V^{-1} - \frac{1}{2}K^{-1} - \frac{1}{2}W,$$

where $W_{\mathrm{f}i} \stackrel{\text{def}}{=} S_i - V_{\mathrm{f}i}^{-1}$; matrix W_{f} is block diagonal with $W_{\mathrm{f}i}$ as its ith block; and we have introduced $W \stackrel{\text{def}}{=} K^{-1}K_{\mathrm{f}}W_{\mathrm{f}}K_{\mathrm{f}}^{\mathrm{T}}K^{-1}$. The fixed point update for V is

$$V^{\mathrm{fx}} = \left(K^{-1} + W\right)^{-1}, \tag{33}$$

which is obtained by setting $\partial Z_h/\partial V$ at V^{fx} to zero. This update is of the same character as Equation 28 for the non-sparse case. In the case where V^{fx} does not yield an improvement to the objective $\log \tilde{Z}_h$, we search for a $V^{\mathrm{cc}} \stackrel{\text{def}}{=} (1-\eta)V + \eta V^{\mathrm{fx}}$, $\eta \in [0,1]$, using the false position method along η. Further details can be found in Appendix C.3.1.

5. On the Sum-to-zero Property

For many single-machine multi-class support vector machines (SVMs, Vapnik 1998; Bredensteiner and Bennett 1999; Guermeur 2002; Lee, Lin, and Wahba 2004), the sum of the predictive functions over the classes is constrained to be zero everywhere. For these SVMs, the constraint ensures the uniqueness of the solution (Guermeur, 2002). The lack of uniqueness without constraint is similar the non-identifiability of parameters in the multinomial probit model in statistics (see Geweke, Keane, and Runkle, 1994, and references therein). For multi-class classification with Gaussian process prior and multinomial logistic likelihood, the redundancy in representation has been acknowledged, but typically uniqueness has not been enforced to avoid arbitrary asymmetry in the prior (Williams and Barber, 1998; Neal, 1998). An exception is the work by Kim and Ghahramani (2006), where a linear transformation of the latent functions has been used to remove the redundancy. In this section, we show that such *sum-to-zero* property is present in the optimal variational posterior under certain common settings.

Recall from Equation 27 in Section 3.3 that $\mathbf{m} = K(\mathbf{y} - \mathbf{b})$ when the lower bound Z_h is optimized. Let $\boldsymbol{\alpha} \stackrel{\text{def}}{=} K^{-1}\mathbf{m}$. Then the set of self-consistent equations at stationary gives $\boldsymbol{\alpha}_i = \mathbf{y}_i - \mathbf{b}_i$, where $\boldsymbol{\alpha}_i$ is the ith C-dimensional sub-vector of $\boldsymbol{\alpha}$. Since $\mathbf{b}_i = \bar{\mathbf{g}}_i$ at stationary, and $\bar{\mathbf{g}}_i$ is a probability vector, it follows that

$$\forall i \qquad \sum_{c=1}^{C} \alpha_i^c = 0, \qquad \text{and} \qquad \alpha_i^c = \begin{cases} -b_i^c \in]-1, 0[& \text{if } y_i^c = 0 \\ 1 - b_i^c \in]0, 1[& \text{if } y_i^c = 1. \end{cases}$$

Consider an input $\mathbf{x}_*$, which may be in the observed set. Let $\mathbf{k}_*^{\mathrm{x}}$ be the vector of covariances to all the other inputs under the covariance function k^{x}. The posterior latent mean of $\mathbf{f}_*$ at $\mathbf{x}_*$ under

separable covariance (1) is

$$\mathbf{m}_* = \left((\mathbf{k}_*^{\mathrm{x}})^{\mathrm{T}} \otimes K^{\mathrm{c}}\right) \boldsymbol{\alpha} = \sum_{i=1}^{n} \left(k^{\mathrm{x}}(\mathbf{x}_*, \mathbf{x}_i) K^{\mathrm{c}}\right) \boldsymbol{\alpha}_i = K^{\mathrm{c}} \sum_{i=1}^{n} k^{\mathrm{x}}(\mathbf{x}_*, \mathbf{x}_i) \boldsymbol{\alpha}_i. \tag{34}$$

Consider the common case where $K^{\mathrm{c}} = I$. Then the posterior latent mean for the cth class is $m_*^c = \sum_{i=1}^{n} k^{\mathrm{x}}(\mathbf{x}_*, \mathbf{x}_i)\alpha_i^c$, and the covariance from the ith datum has a positive contribution if it is from the cth class and a negative contribution otherwise. Moreover, the sum of the latent means is

$$\mathbf{1}^{\mathrm{T}}\mathbf{m}_* = \mathbf{1}^{\mathrm{T}} \sum_{i=1}^{n} k^{\mathrm{x}}(\mathbf{x}_*, \mathbf{x}_i)\boldsymbol{\alpha}_i = \sum_{i=1}^{n} k^{\mathrm{x}}(\mathbf{x}_*, \mathbf{x}_i)\mathbf{1}^{\mathrm{T}}\boldsymbol{\alpha}_i = 0. \tag{35}$$

Hence that the sum of the latent means for any datum, whether observed or novel, is constant at zero. We call this the sum-to-zero property.

The sum-to-zero property is also present, but in a different way, when

$$K^{\mathrm{c}} = M - M\mathbf{1}\mathbf{1}^{\mathrm{T}}M/\mathbf{1}^{\mathrm{T}}M\mathbf{1}, \tag{36}$$

where M is C-by-C and positive semi-definite. This is reminiscent of Equation 20, which gives a similar parametrization for W_*. Using the rightmost expression in (34) for $\mathbf{m}_*$, we find that $\mathbf{1}^{\mathrm{T}}\mathbf{m}_* = 0$ because $K^{\mathrm{c}}\mathbf{1} = \mathbf{0}$. This is in contrast with (35) for $K^{\mathrm{c}} = I$, where the sum-to-zero property holds because $\mathbf{1}^{\mathrm{T}}\boldsymbol{\alpha}_i = 0$.

Setting K^{c} via (36) leads to a degenerate Gaussian process, since the matrix will have a zero eigenvalue even if M is strictly positive definite. Since degeneracy is usually not desirable, we add to (36) the term ηI, where $\eta > 0$:

$$K^{\mathrm{c}} = M - M\mathbf{1}\mathbf{1}^{\mathrm{T}}M/\mathbf{1}^{\mathrm{T}}M\mathbf{1} + \eta I. \tag{37}$$

This not only ensures that K^{c} is positive definite but also preserves the sum-to-zero property. The parametrization effectively constrains the least dominant eigenvector of K^{c} to $\mathbf{1}/\sqrt{C}$.

5.1 The Sum-to-zero Property in Sparse Approximation

The sum-to-zero property is also present in sparse approximation when the inducing variables are such that if f_i^c is an inducing variable, then so are $f_i^1, \ldots, f_i^C$. That is, the C latent variables associated with any input $\mathbf{x}_i$ are either omitted or included together in the inducing set. The sparsity of single-machine multi-class SVMs is of this nature. Let t be the number of inputs for which their latent variables are included.

Under the separable covariance model (1), covariance between the inducing variables and the latent variables is the Kronecker product $K_{\mathrm{f}} = K_{\mathrm{f}}^{\mathrm{x}} \otimes K^{\mathrm{c}}$, where $K_{\mathrm{f}}^{\mathrm{x}}$ is the covariance on the inputs only. The stationary point of the lower bound $\tilde{Z}_h$ in the sparse approximation has the self-consistent equation $\mathbf{m} = K_{\mathrm{f}}(\mathbf{y} - \bar{\mathbf{g}})$; see Section 4.2. As before, let $\boldsymbol{\alpha} \stackrel{\mathrm{def}}{=} K^{-1}\mathbf{m}$. The Gram matrix K is the Kronecker product $K^{\mathrm{x}} \otimes K^{\mathrm{c}}$ under the separable covariance model. Hence $\boldsymbol{\alpha} = \left((K^{\mathrm{x}})^{-1}K_{\mathrm{f}}^{\mathrm{x}} \otimes I\right)(\mathbf{y} - \bar{\mathbf{g}})$ using the mixed-product property. Vector $\boldsymbol{\alpha}$ is the stacking of vectors $\boldsymbol{\alpha}_1, \ldots, \boldsymbol{\alpha}_t$, where each $\boldsymbol{\alpha}_j$ is for one of the t inputs with their latent variables in the inducing set and can be expressed as

$$\boldsymbol{\alpha}_j = \sum_{i=1}^{n} \left((K^{\mathrm{x}})^{-1}K_{\mathrm{f}}^{\mathrm{x}}\right)_{ji} (\mathbf{y}_i - \bar{\mathbf{g}}_i).$$

One finds that $\mathbf{1}^{\mathrm{T}}\boldsymbol{\alpha}_j = 0$, and the discussion for the non-sparse case applies similarly from Equation 34 onwards.

6. Model Learning

Model learning in a Gaussian process model is achieved by maximizing the marginal likelihood with respect to the parameters $\boldsymbol{\theta}$ of the covariance function. In the case of variational inference, the lower bound on the marginal likelihood is maximized instead. For the non-sparse variational approximation to the multinomial logit Gaussian process, this is

$$\log Z_h^*(\boldsymbol{\theta}) \stackrel{\text{def}}{=} \max_{\mathbf{m},V,\{\mathbf{b}_i\},\{S_i\}} \log Z_h(\mathbf{m},V,\{\mathbf{b}_i\},\{S_i\};X,\mathbf{y},\boldsymbol{\theta}),$$

which is the maximal lower bound on log marginal likelihood on the observed data $(X,\mathbf{y})$. The maximization is achieved by ascending the gradient

$$\begin{aligned}\frac{\mathrm{d}\log Z_h^*}{\mathrm{d}\theta_j} &= -\frac{1}{2}\operatorname{tr}\left(K^{-1}\frac{\partial K}{\partial\theta_j}\right) + \frac{1}{2}\operatorname{tr}\left(K^{-1}VK^{-1}\frac{\partial K}{\partial\theta_j}\right) + \frac{1}{2}\mathbf{m}^{\mathrm{T}}K^{-1}\frac{\partial K}{\partial\theta_j}K^{-1}\mathbf{m} \\ &= \frac{1}{2}\operatorname{tr}\left(\left(\boldsymbol{\alpha}\boldsymbol{\alpha}^{\mathrm{T}} - K^{-1} + K^{-1}VK^{-1}\right)\frac{\partial K}{\partial\theta_j}\right),\end{aligned}$$

where $\boldsymbol{\alpha} \stackrel{\text{def}}{=} K^{-1}\mathbf{m}$. This gradient is also the partial and explicit gradient of $\log Z_h$ with respect to θ_j. The implicit gradients via the variational parameters are not required since the derivative of $\log Z_h$ with respect to each of them is zero at the fixed point $\log Z_h^*$.

For the sparse approximation, we differentiate $\log \tilde{Z}_h^*$—the optimized bound on the log marginal likelihood for the sparse case given by Equation 32—with respect to the covariance function parameter θ_j. The derivation in Appendix C.4 gives

$$\begin{aligned}\frac{\mathrm{d}\log \tilde{Z}_h^*}{\mathrm{d}\theta_j} = -\frac{1}{2}\operatorname{tr}&\left(\left(\boldsymbol{\alpha}\boldsymbol{\alpha}^{\mathrm{T}} - K^{-1} + K^{-1}VK^{-1} + W\right)\frac{\partial K}{\partial\theta_j}\right) \\ &+ \operatorname{tr}\left(\left((\mathbf{y}-\bar{\mathbf{g}})\boldsymbol{\alpha}^{\mathrm{T}} + W_{\mathrm{f}}K_{\mathrm{f}}^{\mathrm{T}}\left(K^{-1} - K^{-1}VK^{-1}\right)\right)\frac{\partial K_{\mathrm{f}}}{\partial\theta_j}\right) - \frac{1}{2}\operatorname{tr}\left(W_{\mathrm{f}}\frac{\partial K_{\mathrm{ff}}}{\partial\theta_j}\right),\end{aligned}$$

where $\boldsymbol{\alpha} \stackrel{\text{def}}{=} K^{-1}\mathbf{m}$, and matrices W_{f} and W are defined in Section 4.2.

The selection of the inducing set in sparse approximation can also be seen as a model learning problem (Snelson and Ghahramani, 2006; Titsias, 2009a).[1] This is addressed in the reminder of this section.

6.1 Active Inducing Set Selection

The quality of the sparse approximation depends on the set of inducing sites. Prior works have suggested using scores to greedily and iteratively add to the set. The Informative Vector Machine (IVM, Lawrence et al. 2003) and its generalization to multiple classes (Seeger and Jordan, 2004) use the differential entropy, which is the amount of additional information to the posterior. Alternatives based on the data likelihood have also been proposed (Girolami and Rogers, 2006; Henao and Winther, 2010). However, since our aim has always been to maximize the marginal likelihood $p(\mathbf{y})$ of the observed data, it is natural to choose the inducing sites that effect the most increase in the marginal likelihood. The same thought is behind the scoring for greedy selection in the

1. However, in the strict sense, the exact model is fixed during the selection of the inducing set: the object that is learned is the approximating model.

sparse variational approximation to Gaussian process regression (Titsias, 2009a). For multi-class classification, it is too expensive to compute the exact increase in the marginal likelihood. Instead, we use the lower bound on the increment to (the lower bound on) the marginal likelihood.

Throughout, the asterisk $*$ will be used to subscript variables pertaining to the newly introduced inducing site $\tilde{\mathbf{x}}_*$. Given the current set of inducing sites $\tilde{X}$, the inclusion of $\tilde{\mathbf{x}}_*$ gives the new set $\tilde{X}_*$. The function values at $\tilde{\mathbf{x}}_*$, $\tilde{X}$ and $\tilde{X}_*$ are denoted by z_*, $\mathbf{z}$ and $\mathbf{z}_* \stackrel{\text{def}}{=} (\mathbf{z}^{\mathrm{T}}, z_*)^{\mathrm{T}}$. There is only one random scalar variable z_* at the inducing site $\tilde{\mathbf{x}}_*$. In contrast, there is a random C-vector $\mathbf{f}_i$ at an observed input $\mathbf{x}_i$; see Remark 14. Hence there are C potential inducing sites from a single observed site $\mathbf{x}_i$: $\tilde{\mathbf{x}}_* \in \{(\mathbf{x}_i, 1), \ldots, (\mathbf{x}_i, C)\}$.

We aim to select $\tilde{\mathbf{x}}_*$ that maximizes the increase in the optimized lower bounds on the marginal likelihood: $d(\tilde{\mathbf{x}}_*; \tilde{X}) \stackrel{\text{def}}{=} \log \tilde{Z}_h^*(\tilde{X}_*) - \log \tilde{Z}_h^*(\tilde{X})$, where $\tilde{X}_* \stackrel{\text{def}}{=} \{\tilde{\mathbf{x}}_*\} \cup \tilde{X}$, and

$$\log \tilde{Z}_h^*(\tilde{X}_*) \stackrel{\text{def}}{=} \max_{\mathbf{m}_*, V_*, \{\mathbf{b}_{*i}\}, \{S_{*i}\}} \log \tilde{Z}_h(\mathbf{m}_*, V_*, \{\mathbf{b}_{*i}\}, \{S_{*i}\}; \tilde{X}_*),$$
$$\log \tilde{Z}_h^*(\tilde{X}) \stackrel{\text{def}}{=} \max_{\mathbf{m}, V, \{\mathbf{b}_i\}, \{S_i\}} \log \tilde{Z}_h(\mathbf{m}, V, \{\mathbf{b}_i\}, \{S_i\}; \tilde{X}).$$

In words, $\tilde{Z}_h^*(\tilde{X})$ is the optimized lower bound on marginal likelihood with the current inducing set $\tilde{X}$, while $\tilde{Z}_h^*(\tilde{X}_*)$ is the optimized lower bound with the proposed new inducing set $\tilde{X}_*$. Because $\tilde{Z}_h$ combines the Kullback-Leibler divergence of the prior from the approximate posterior and the sum of the lower bounds on the expected log-likelihoods, $d(\tilde{\mathbf{x}}_*; \tilde{X})$ includes both the change in the approximate posterior and the effect of this change in explaining the observed data.

Computing $d(\tilde{\mathbf{x}}_*; \tilde{X})$ involves $\tilde{Z}_h^*(\tilde{X}_*)$, and this can be computationally expensive. A more viable alternative is to lower bound $d(\tilde{\mathbf{x}}_*; \tilde{X})$ by fixing selected variational parameters in $\tilde{Z}_h(\cdots; \tilde{X}_*)$ to the optimal ones from $\tilde{Z}_h^*(\tilde{X})$, which has already been computed. Let

$$\{\mathbf{m}, V, \{\mathbf{b}_i\}, \{S_i\}\} \stackrel{\text{def}}{=} \arg \log \tilde{Z}_h^*(\tilde{X}).$$

For the inducing set $\tilde{X}_*$, we set the prior on the inducing and latent variables, and the approximate posterior on the inducing variables to

$$p\left(\begin{pmatrix}\mathbf{z}_* \\ \mathbf{f}\end{pmatrix}\right) \stackrel{\text{def}}{=} \mathcal{N}\left(\mathbf{0}, \begin{pmatrix}K_* & K_{\mathrm{f}*} \\ K_{\mathrm{f}*}^{\mathrm{T}} & K_{\mathrm{ff}}\end{pmatrix}\right), \qquad q(\mathbf{z}_* \mid \mathbf{y}) \stackrel{\text{def}}{=} \mathcal{N}(\mathbf{m}_*, V_*),$$

where

$$K_* \stackrel{\text{def}}{=} \begin{pmatrix}K & \mathbf{k}_* \\ \mathbf{k}_*^{\mathrm{T}} & k_{**}\end{pmatrix}, \qquad K_{\mathrm{f}*} \stackrel{\text{def}}{=} \begin{pmatrix}K_{\mathrm{f}} \\ \mathbf{k}_{\mathrm{f}*}^{\mathrm{T}}\end{pmatrix}, \qquad \mathbf{m}_* \stackrel{\text{def}}{=} \begin{pmatrix}\mathbf{m} \\ m_*\end{pmatrix}, \qquad V_* \stackrel{\text{def}}{=} \begin{pmatrix}V & \mathbf{v}_* \\ \mathbf{v}_*^{\mathrm{T}} & v_{**}\end{pmatrix}. \tag{38}$$

The above choice of posterior fixes the mean and the covariance of $\mathbf{z}$ for $\tilde{X}$ to the mean $\mathbf{m}$ and covariance V in $\log \tilde{Z}_h^*(\tilde{X})$. Further setting $\{\mathbf{b}_{*i}\} \equiv \{\mathbf{b}_i\}$ and $\{S_{*i}\} \equiv \{S_i\}$, the additional variational parameters are those in the posterior of the inducing points for the additional site $\tilde{\mathbf{x}}_*$. Since we are optimizing over only a subset of the possible parameters, we obtain a lower bound on $d(\tilde{\mathbf{x}}_* | \tilde{X})$:

$$d(\tilde{\mathbf{x}}_* | \tilde{X}) \geq d_1(\tilde{\mathbf{x}}_* | \tilde{X}) \stackrel{\text{def}}{=} \max_{m_*, v_{**}, \mathbf{v}_*} \log \tilde{Z}_h(\mathbf{m}_*, V_*, \{\mathbf{b}_i\}, \{S_i\}; \tilde{X}_*) - \log \tilde{Z}_h^*(\tilde{X}), \tag{39}$$

where $\mathbf{m}_*$ and V_* are as defined in Equation 38. By separating $\log \tilde{Z}_h$ into its summands expressed in Equation 30, we write

$$d_1(\tilde{\mathbf{x}}_* | \tilde{X}) = \max_{m_*, v_{**}, \mathbf{v}_*} \left(d_{\mathrm{KL}}(m_*, v_{**}, \mathbf{v}_*, \tilde{\mathbf{x}}_* | \tilde{X}) + \sum_{i=1}^{n} d_h^i(m_*, v_{**}, \mathbf{v}_*, \tilde{\mathbf{x}}_* | \tilde{X}) \right),$$

where

$$d_{\mathrm{KL}}(m_*, v_{**}, \mathbf{v}_*, \tilde{\mathbf{x}}_*|\tilde{X}) \stackrel{\text{def}}{=} -\mathrm{KL}(q(\mathbf{z}_* \mid \mathbf{y}) \,\|\, p(\mathbf{z}_*)) + \mathrm{KL}(q(\mathbf{z} \mid \mathbf{y}) \,\|\, p(\mathbf{z})), \tag{40}$$

$$d_h^i(m_*, v_{**}, \mathbf{v}_*, \tilde{\mathbf{x}}_*|\tilde{X}) \stackrel{\text{def}}{=} h(\mathbf{y}_i; q_{*i}, \mathbf{b}_i, S_i) - h(\mathbf{y}_i; q_i, \mathbf{b}_i, S_i). \tag{41}$$

The expressions for d_{KL} and d_h^i in terms of the variational parameters m_*, v_{**} and $\mathbf{v}_*$ are given in Appendix D.1. On inspecting these expressions, we find that the contributions from m_* and $(\mathbf{v}_*, v_{**})$ are decoupled in objective function $d_{\mathrm{KL}} + \sum d_h$ within d_1, so the search for the optimal m_* and $(\mathbf{v}_*, v_{**})$ are can be perform separately. Moreover, $d_{\mathrm{KL}} + \sum d_h$ is concave in m_* and v_{**} but not necessarily concave in $\mathbf{v}_*$. These findings are elaborated in Appendix D.2, which also gives the gradient updates for $\mathbf{m}_*$ and v_{**} (given a fixed $\mathbf{v}_*$).

The non-concavity in $\mathbf{v}_*$ makes the maximization in d_1 less feasible. To make progress, we fix $\mathbf{v}_*$ to be that which maximizes only d_{KL}. This gives $\mathbf{v}_* = VK^{-1}\mathbf{k}_*$ and leads to a second lower bound. This lower bound is non-trivial in the sense that it is non-negative. This is established in following lemma, which leads to another proposition.

Lemma 16 *Let functions d_1, d_{KL} and d_h^i be as defined in Equations 39, 40 and 41, and let*

$$d_2(\tilde{\mathbf{x}}_*|\tilde{X}) \stackrel{\text{def}}{=} \max_{m_*, v_{**}} \left(d_{\mathrm{KL}}(m_*, v_{**}, VK^{-1}\mathbf{k}_*, \tilde{\mathbf{x}}_*|\tilde{X}) + \sum_{i=1}^{n} d_h^i(m_*, v_{**}, VK^{-1}\mathbf{k}_*, \tilde{\mathbf{x}}_*|\tilde{X}) \right). \tag{42}$$

Then $0 \leq d_2(\tilde{\mathbf{x}}_*|\tilde{X}) \leq d_1(\tilde{\mathbf{x}}_*|\tilde{X})$.

Proof Function d_2 is upper bounded by d_1 because it maximizes over a subset of the variational parameters in d_1. For non-negativity, we observe that the objective function within d_2 is zero when we set $m_* = \mathbf{k}_*^{\mathrm{T}} K^{-1}\mathbf{m}$ and $v_{**} = k_{**} - \mathbf{k}_*^{\mathrm{T}} K^{-1}\mathbf{k}_* + \mathbf{k}_*^{\mathrm{T}} K^{-1} V K^{-1}\mathbf{k}_*$. ■

Proposition 17 *For the sparse variational approximation to the multinomial logit Gaussian process, any site added to the inducing set can never decrease the lower bound $\tilde{Z}_h^*$ to the marginal likelihood.*

This proposition is analogous one for Gaussian process regression (Titsias, 2009a, Proposition 1). Hence, we can interleave the greedy selection of inducing sites with hyper-parameters optimization (Titsias, 2009a, Section 3.1). One might have thought that this proposition is trivial because an additional inducing variable increases the flexibility of the variational model. Such an argument would have worked if we had compared the exact marginal likelihood $p(\mathbf{y})$ or the optimized variational lower bound $\tilde{Z}_B^*$. It would not have worked here because the optimized lower bound $\tilde{Z}_h^*$ is used here.

6.1.1 SUBSAMPLING AND FILTERING

Computation of d_2 for every possible site requires the full Gram matrix. This is because the required vector $\mathbf{k}_{\mathrm{f}*}$ for the site $\tilde{\mathbf{x}}_* = (\mathbf{x}_*, c)$ under consideration is the covariance from $\mathbf{x}_*$ to all the other observed data. This may be undesirable when covariance function is expensive to evaluate. In this case, we propose to approximate $\sum_{i=1}^{n} d_h^i$, which is over the whole data set, with one that is computed over a subset $\mathcal{S}$:

$$d_3(\tilde{\mathbf{x}}_*, \mathcal{S}|\tilde{X}) \stackrel{\text{def}}{=} \max_{m_*, v_{**}} \left(d_{\mathrm{KL}}(m_*, v_{**}, VK^{-1}\mathbf{k}_*, \tilde{\mathbf{x}}_*|\tilde{X}) + \frac{n}{|\mathcal{S}|} \sum_{i \in \mathcal{S}} d_h^i(m_*, v_{**}, VK^{-1}\mathbf{k}_*, \tilde{\mathbf{x}}_*|\tilde{X}) \right), \tag{43}$$

where $\mathcal{S}$ is the set of indices of the data to evaluate against. By partitioning the observed data appropriately, the number of covariance function evaluations can be reduced. This d_3 score can be used to directly choose the site to be added to the inducing set. Alternatively, it can be used as a filtering step so that only sites with high d_3 scores are further evaluated using the more expensive d_2 score function.

7. Computational Complexity

We now discuss the computation complexity of the approximate inference. For the non-sparse approximate inference, $O(n^3C^3)$ computations are required per iteration of the variational bound optimization, where n is the size of the observed data set and C is the number of classes. For the sparse approximate inference, $O(nC^3 + nCs^2 + nC^2s + s^3)$ computations are required per iteration, where s is the number of inducing variables. This complexity is when we exploit the block diagonal structure of the variables. The complexity of computing the set of n d_2-scores for active inducing set selection is the same. For probabilistic prediction with the posterior using sparsity, computing the lower bounds (24) to the predictive probabilities requires $O(C^3 + Cs^2 + C^2s)$ computations per datum, while computing the re-normalized probabilities (25) needs $O(Cs)$, which is less. For prediction with the posterior in the non-sparse case, these are $O(nC^3 + n^2C)$ and $O(nC^2)$ respectively.

One might have thought that complexity can be improved if $K^{\mathrm{c}} = I$ so that K is block diagonal (after re-ordering) in the non-sparse approximation. However, we have not been able to exploit this structure. This is because computing $K^{-1} + W$ in Equation 28 involves W that is block diagonal with a different ordering, which essentially destroys the structure (Seeger and Jordan, 2004).

For the sparse approximate inference, the direct complexity with respect to n is linear. If we let $s \sim \log Cn$, then the overall complexity is $O(n \log^2 n)$ in n. Let us now consider three regimes depending on C. For $n \ll C$, we opine that some clustering process may be more appropriate than the classification model consider here. For $C \ll s$, the dominant complexity is $O(nCs^2)$. For $s \ll C \ll n$, the dominant complexity is $O(nC^3)$, which is for optimizing the variational parameters $\mathbf{b}_i$ and S_i for each of the n observed data.[2] Reducing the cubic complexity in C requires constraining the variational parameters. In particular, one may constrain $S_i = (V_{\mathrm{f}i})^{-1} + W_i$ where $W_i = \gamma_i(\Pi_i - \boldsymbol{\pi}_i\boldsymbol{\pi}_i^{\mathrm{T}})$, $\gamma_i > 0$ and $\boldsymbol{\pi}_i$ is a probability vector. As remarked upon after Theorem 6, we have found that this constraint gives bounds that are quite loose. In addition, our present opinion is that effective inference with such a small inducing set may require rather strong correlations in both K^{c} and $k^{\mathrm{x}}(\cdot,\cdot)$ of the prior. We defer further investigation in the regime $s \ll C \ll n$ to future work.

In the $C \ll s$ regime, Seeger and Jordan (2004) and Girolami and Rogers (2006) have reported $O(nCs^2)$ computational complexity. In their cases, however, this complexity includes both the inference with a subset of the observed data and the active selection of the subset. Direct comparison with our approach can be misleading: the $O(nCs^2)$ in the preceding paragraph does not include active selection, but it does include projecting from the inducing variables to the entire set of observed data in the sparse approximate inference. Including the cost of greedy active selection up to s inducing variables gives $O(nCs^3)$.

2. In this regime, one should optimize the $\mathbf{b}_i$s separately. This is cheaper than optimizing $\mathbf{m}$ and $\mathbf{b}$ jointly.

	Model		Approximating Posterior		Likelihood Approximation	
Citation	prior	likelihood	Family	Principle	Learning	Prediction
Williams and Barber (1998)	i.i.d.	logistic	Gaussian	Laplace	Exact	Monte Carlo
Neal (1998)	i.i.d.	logistic	Samples	MCMC	MCMC	MCMC
Gibbs (1997)	independent	logistic	Factored Gaussian	Variational	Variational	Analytic approximation
Seeger and Jordan (2004)	i.i.d.	logistic	Gaussian	ADF	Quadrature	Quadrature
Kim and Ghahramani (2006)	i.i.d.	uniform	Gaussian	EP	EP	
Girolami and Rogers (2006)	i.i.d.	probit	Factored Gaussian	Variational	Monte Carlo	Monte Carlo
This paper	separable	logistic	Gaussian	Variational	Variational	Variational

Table 1: Existing works in multi-class Gaussian processes and their different aspects. In this paper, the likelihood approximation is for the expectation of the log-likelihood.

8. Related Work

We now discuss related works on multi-class Gaussian process classification. Table 1 tabulates different aspects of the existing related works that we know in the machine learning literature. Most consider the case where the latent functions are independent and identically distribution (i.i.d.), although Williams and Barber (1998) have seen no difficulty in extending to correlated latent functions. Gibbs (1997) has considered the case where the covariance functions of the prior latent Gaussian processes are independent and assumed to be from the same parametric family with possibility different parameters. In this paper, most results are applicable as long as the latent functions are jointly Gaussian, although at specific places we consider the separable covariance in Equation 1.

As with most existing works, our likelihood function is the multinomial logistic (2). Other likelihood functions are possible. In particular, one class of likelihood functions uses auxiliary independent random variables u^cs, $c = 1,\ldots,C$, and determine the class by $\arg\max_c u^c$, The multinomial logistic is in this class, and it is obtained when each auxiliary variable u^c is Gumbel distributed with $p(u^c|f^c) = te^{-t}$, where $t \stackrel{\text{def}}{=} e^{-(u^c - f^c)}$ (McFadden, 1974). If $u^c \sim \mathcal{N}(f^c, 1)$, then the likelihood is the multinomial probit used by Girolami and Rogers (2006). If each auxiliary variable u^c is supported only at f^c, we have the threshold likelihood function used by Kim and Ghahramani (2006). From this perspective, a model with the threshold likelihood function and prior covariance function $k^{\mathrm{x}}(\mathbf{x},\mathbf{x}') + \delta(\mathbf{x},\mathbf{x}')$, where δ is the Kronecker delta function, is essentially the same as the model with the multinomial probit likelihood and prior covariance function $k^{\mathrm{x}}(\mathbf{x},\mathbf{x}')$. Kim and Ghahramani (2006) have also used uniform noise (Angluin and Laird, 1988) with the threshold likelihood.

With any of these likelihoods, exact inference is non-tractable and approximations must be used. Except for the work of Neal (1998) where the approximation is a set of samples obtained from Markov Chain Monte Carlo (MCMC), all existing works have used a Gaussian approximation to the true posterior. The approximating Gaussian can be determined through fitting using different principles: Laplace (Williams and Barber, 1998), assumed density filtering (ADF, Seeger and Jordan, 2004) and expectation propagation (EP, Kim and Ghahramani, 2006), and variational bounding (Gibbs, 1997; Girolami and Rogers, 2006).

This paper uses the variational approach, for which a lower bound on the marginal likelihood can be obtained. However, the variational approach used in this paper differs from those in existing

works (Gibbs, 1997; Girolami and Rogers, 2006). Gibbs has placed Gaussian-type bounds that factorizes over the classes on the multinomial logistic likelihood functions. Since the prior also factorizes over the classes, the approximate posterior factorizes similarly. Girolami and Rogers have constrained the approximating Gaussian to factorize over classes from the onset, and have proceeded to use variational mean field to obtain the factors. In contrast, the approximating Gaussian in this paper does not factorize over classes. We begin from an unconstrained Gaussian. This is followed by the Kullback-Leibler divergence and a bound on the expected log-likelihood (Theorem 6). Neither of these steps needs factorization over classes.

In general, the approximating Gaussian has covariance $(K^{-1}+W)^{-1}$, where W is a block diagonal matrix of n C-by-C blocks. Let W_i be the ith $C\times C$ block in W. The matrix W is diagonal when the assumed likelihood factorizes over classes and data (Gibbs, 1997; Girolami and Rogers, 2006). Let $\boldsymbol{\pi}_i$ be a probability vector, and let Π_i be the diagonal matrix with $\boldsymbol{\pi}_i$ along its diagonal. Then the ith block W_i of W in the Laplace approximation (Williams and Barber, 1998) is $\Pi_i - \boldsymbol{\pi}_i\boldsymbol{\pi}_i^{\mathrm{T}}$, where the cth element in $\boldsymbol{\pi}_i$ is the multinomial logistic $p(y_i^c|\mathbf{f}_i)$. This parametrization of W_i follows directly from fitting principle of Laplace approximation. If computational time complexity is important, one can also use the parameterization $W_i \stackrel{\text{def}}{=} \gamma_i\left(\Pi_i - \boldsymbol{\pi}_i\boldsymbol{\pi}_i^{\mathrm{T}}\right)$, where $\gamma_i > 0$ and $\boldsymbol{\pi}_i$ are to be estimated, to obtain the same computational complexity as factorized mean-field (Seeger and Jordan, 2004). If computational time complexity is not a major consideration, any positive definite W_i can be used for a tighter approximation (see Kim and Ghahramani, 2006, for example). In the present paper, each block W_i is determined by optimizing the expected log-likelihood of the ith datum. The optimized W_i has null space $\{\eta\mathbf{1} \mid \eta \in \mathbb{R}\}$. Further discussions in relation to the works of Williams and Barber (1998) and Seeger and Jordan (2004) have been given after Theorem 6.

The approximate predictive probability in the multi-class Gaussian process model is the expected likelihood under the approximate posterior. This is intractable and approximations are needed. Two common approaches are Monte Carlo (Williams and Barber, 1998; Neal, 1998; Girolami and Rogers, 2006) and traditional numerical integration (Seeger and Jordan, 2004). Analytic approximation has also been used (Gibbs, 1997). In this paper, we have given a variational approximation of the expected log-likelihood through Theorem 6. In Section 9.1.1, we will see that this is quite tight on average.

In previous works, sparsity in multi-class Gaussian processes is achieved by performing inference using only a subset of the observed data (SoD), which can be selected actively (Seeger and Jordan, 2004; Girolami and Rogers, 2006). In contrast, the sparsity in this paper is achieved through using the subset to induce the entire set. Quiñonero-Candela et al. (2007) have discussed in detail the SoD approach and the more general inducing approaches in the context of regression. We use the variational approach for both sparse approximation and active selection of the subset. For regression, this approach has been shown to have several desirable characteristics over the other approaches (Titsias, 2009a).

9. Experiments and Results

We evaluate our approach to multi-class logit Gaussian process classification in various aspects. In Section 9.1, we compare the bounds in the marginal likelihood and the predictive likelihood provided by our variational approach with those provided by the variational mean-field approximation

Name	train set	test set	classes	attributes	task description
iris	90	60	3	4	determine class of iris plant
thyroid	129	86	3	5	diagnosis of a patient's thyroid
wine	106	72	3	13	determine the cultivar of wine
glass	128	86	7	9	determine the type of glass

Table 2: Summary of the four UCI data sets used in our experiments. For the glass data, there is no instance of class "vehicle windows that are non-float processed" in the set.

to multinomial probit regression (Girolami and Rogers, 2006).[3] We also look at how the quality of our bounds vary with the prior variance of the latent process. In Section 9.2, we relate the logit to the probit, and we also look at the the prior correlation between the latent process. Section 9.3 investigates the effectiveness of active inducing set selection using the criteria proposed in Section 6.1. In Section 9.4, we compare with single-machine multi-class support vector machines.

For comparison, we use a tight approximation to the exact posterior of the multi-class logit and probit Gaussian process model. This is obtained by importance sampling where the proposal is the multivariate-t distribution (Kotz and Nadarajah, 2004) with four degrees of freedom, centered at the mean $\mathbf{m}^*$ of our variational approximation to $p(\mathbf{f}|\mathbf{y})$ and with covariance $2K$. We have found this to be more effective than the Gibbs sampling used by Girolami and Rogers (2006) and the anneal importance sampling (Neal, 2001) used by Nickisch and Rasmussen (2008). Due to the central limit theorem, the Monte Carlo estimate $\hat{p}(\mathbf{y})$ on the marginal likelihood has distribution $\mathcal{N}(p(\mathbf{y}), \sigma^2/n_s)$, where σ^2 is the true variance of the importance weights and n_s is the number of samples. When reporting the marginal likelihood estimate, we use $\hat{p}(\mathbf{y}) + 3.09\sigma/\sqrt{n_s}$ to upper bound $p(\mathbf{y})$ with probability 0.999, where σ is estimated from the samples. In our experiments, $n_s = 100,000$ for each Monte Carlo run. Details are in Appendix F.[4]

Our experiments are conducted on four data sets from the UCI Machine Learning Repository (Frank and Asuncion, 2010): *iris*, *thyroid*, *wine* and *glass*. Following Girolami and Rogers (2006), for each data set, 60% is used for training and 40% for testing. Each input attribute is normalized to zero mean and unit variance on the training set. Our experiments are conducted with fifty such random splits for each data set. The summary statistics for the data sets are given in Table 2.

9.1 Comparing Variational Approaches

We evaluate our approach against variational mean-field (Girolami and Rogers, 2006) and importance sampling. We fix the latent random functions to be independent and identically distributed (i.i.d.); that is, $K^{\mathrm{c}} = I$. The covariance function on inputs $\mathbf{x}$ and $\mathbf{x}'$ in $\mathbb{R}^d$ is the unit variance squared-

3. We use version 1.6.0 of the *R* package available at `http://www.bioconductor.org/packages/devel/bioc/html/vbmp.html`.

4. We have also experimented with using the approximate posterior $q(\mathbf{f}|\mathbf{y})$ directly as the proposal distribution (Ghahramani and Beal, 2000a,b). The set of estimates obtained in this way is generally indistinguishable from that obtained using the multivariate-t distribution. However, the latter comes with a convergence rate guarantee because the tail of the t distribution is heavier than that of a Gaussian.

exponential covariance function

$$k^{\mathrm{x}}(\mathbf{x},\mathbf{x}') = \mathrm{usqexpard}(\mathbf{x},\mathbf{x}';\boldsymbol{\theta}) \stackrel{\text{def}}{=} \exp\left(-\frac{1}{2}\sum_{j=1}^{d}\left(\frac{x_j - x'_j}{\theta_j}\right)^2\right), \tag{44}$$

where x_j (resp. x'_j) is the jth dimensional of $\mathbf{x}$ (resp. $\mathbf{x}'$), θ_j is the length-scale of the jth dimension, and $\boldsymbol{\theta} \stackrel{\text{def}}{=} (\theta_1,\ldots,\theta_d)^{\mathrm{T}}$ collects parameters. Each length-scale affects the influence of the dimension; this allows automatic relevance determination (ARD, Neal 1996).

Table 3 reports the results comparing the log marginal likelihood $\log Z$ or its bounds on the training sets, and the predictive error and the log joint predictive probability $\log p(\mathbf{y}_*)$ on the test sets. The means and standard deviations over the fifty partitions for each data set are given. We use MNL to denote multinomial logistic likelihood and MNP to denote multinomial probit likelihood. KL-MNL is our variational approach with multinomial logistic likelihood, and MF-MNP is the variational mean-field with multinomial probit likelihood (Girolami and Rogers, 2006). MC-MNP and MC-MNL are the Monte Carlo approximations using importance sampling. The marginal likelihood estimates for importance sampling are the high confidence upper bounds. Column *Theorem 10* gives theoretic lower bounds on the marginal likelihood for the logistic likelihood. Results with two sets of hyper-parameters are given: one from the variational mean parameter estimation for MF-MNP (Girolami and Rogers, 2006), and the other from the model learning for KL-MNL (Section 6). With either set of hyper-parameters, the prior latent process has unit variance due to the choice of covariance function. Our results for MC-MNP are consistent with, but tighter than, those reported by Girolami and Rogers (2006, Table 1, column *Gibbs Sampler*).

We first compare the marginal likelihoods on the training data along the rows headed by $\log Z$. For either set of hyper-parameters, our variational approach (KL-MNL) gives lower bounds that are very close to the high confidence upper bounds on the marginal likelihoods obtained by sampling (MC-MNL). In fact, it is this tightness that leads us to finally use importance sampling: with Gibbs sampling and annealed importance sampling, we are unable to obtain estimates that are larger than our lower bounds. For MF-MNP, we find that it consistently gives lower bounds that are looser than the theoretical ones under column *Theorem 10*. This suggests that the theoretic bounds may be useful as sanity checks for variational approaches, although we must qualify that the theoretic bounds are for the multinomial logistic likelihood rather than for the probit one. Using the reasoning to be outlined in Section 9.2.1, we obtain approximate theoretic bounds for the probit using $\sigma^2 = \pi^2/6$ in Theorem 10, and these bounds are -139.84, -200.44, -164.70 and -331.06 for the iris, thyroid, wine and glass data sets respectively. These are lower than the theoretical ones in Table 3, but are still higher the bounds given by MF-MNP. The looseness of the bounds given by MF-MNP is also evident when compared with the estimates from sampling (MC-MNP).

Next, we compare the log joint predictive probability $\log p(\mathbf{y}_*)$ on the test set. One set of results obtained using Monte Carlo is reported for MF-MNP, MC-MNP and MC-MNL. Two sets of results are reported for KL-MNL: the upper set uses the re-normalized probabilities given by Equation 25 and the lower set uses Equation 24. As discussed in Section 2.3.3, the probabilities based on Equation 24 are approximate lower bounds on the exact predictive probabilities, while the re-normalized probabilities are always larger than these lower bounds. In the table, we see that these two sets of predictive probabilities from KL-MNL bound the probabilities from MC-MNL rather tightly. For MF-MNP, its predictive probabilities are close to those given by sampling (MC-MNP). This suggests that variational mean-field, which couples of posterior means of the latent function

		MF-MNP-$\boldsymbol{\theta}$				KL-MNL-$\boldsymbol{\theta}$	
	Theorem 10	MF-MNP	MC-MNP	KL-MNL	MC-MNL	KL-MNL	MC-MNL
Iris							
$\log Z$	-125.28	-187.32 ± 1.71	-27.71 ± 1.74	-32.63 ± 1.60	-32.48 ± 1.60	-31.46 ± 1.36	-31.32 ± 1.37
Error		2.66 ± 1.27	2.64 ± 1.26	2.66 ± 1.26	2.64 ± 1.22	2.28 ± 1.25	2.28 ± 1.25
$\log p(\mathbf{y}_*)$		-10.35 ± 1.87	-10.53 ± 1.89	-11.27 ± 1.79	-12.61 ± 1.70	-9.45 ± 1.38	-10.90 ± 1.35
				-13.15 ± 1.82		-11.38 ± 1.48	
Thyroid							
$\log Z$	-179.57	-270.63 ± 3.60	-41.54 ± 3.70	-47.15 ± 3.49	-46.97 ± 3.49	-45.13 ± 2.85	-44.95 ± 2.85
Error		7.84 ± 2.54	7.86 ± 2.54	7.92 ± 2.75	8.00 ± 2.85	6.44 ± 2.92	6.52 ± 3.03
$\log p(\mathbf{y}_*)$		-22.02 ± 4.57	-22.10 ± 4.61	-23.08 ± 4.58	-24.29 ± 4.27	-20.55 ± 4.58	-22.13 ± 4.19
				-25.67 ± 4.66		-23.85 ± 4.64	
Wine							
$\log Z$	-147.56	-222.63 ± 1.91	-36.41 ± 2.07	-42.56 ± 1.96	-42.38 ± 1.96	-41.18 ± 1.74	-41.01 ± 1.74
Error		4.88 ± 2.74	4.88 ± 2.88	4.96 ± 2.73	4.94 ± 2.75	3.22 ± 1.83	3.22 ± 1.73
$\log p(\mathbf{y}_*)$		-16.19 ± 3.84	-16.27 ± 3.94	-17.19 ± 3.83	-19.01 ± 3.59	-14.47 ± 2.03	-16.74 ± 1.98
				-20.37 ± 3.96		-18.02 ± 2.28	
Glass							
$\log Z$	-300.61	-827.58 ± 6.46	-150.23 ± 6.88	-158.16 ± 5.77	-157.53 ± 5.79	-154.74 ± 5.08	-154.08 ± 5.04
Error		33.72 ± 4.03	36.00 ± 4.16	34.20 ± 4.03	34.40 ± 4.09	32.62 ± 4.09	33.02 ± 4.05
$\log p(\mathbf{y}_*)$		-89.63 ± 6.15	-95.62 ± 8.78	-92.78 ± 6.09	-94.79 ± 5.50	-88.82 ± 5.49	-91.31 ± 5.00
				-101.00 ± 5.93		-97.97 ± 5.58	

Table 3: Results with the usqexpard covariance function (44) on inputs and with i.i.d. latent functions. The log marginal likelihood $\log Z$ (or its bounds), the empirical error and the log joint predictive probability $\log p(\mathbf{y}_*)$ (or its bounds and approximations) are reported with means and standard deviations over 50 partitions. Theoretic lower bounds are given under Theorem 10. Methods with MNP after the dash uses of the multinomial probit likelihood, while those with MNL uses the multinomial logistic likelihood. MF-MNP is the variational mean-field method (Girolami and Rogers, 2006), KL-MNL is the variational approach of this paper, while MC-MNP and MC-MNL are importance sampling. Columns under MF-MNP-$\boldsymbol{\theta}$ use the estimated mean hyper-parameters for MF-MNP; those under KL-MNL-$\boldsymbol{\theta}$ use the hyper-parameters optimized for KL-MNL. Method KL-MNL reports two sets of approximations to $\log p(\mathbf{y}_*)$: the upper set uses the re-normalized probabilities given by Equation 25 and the lower set uses the lower bound in Equation 24.

values, is perhaps sufficient for accurate predictive probabilities. In addition, the figures suggest that the predictive probabilities by MF-MNP upper bound those by MC-MNP. Further analysis is needed to confirm this for the general case.

Finally, we compare errors on test data. For the MF-MNP-$\boldsymbol{\theta}$ hyper-parameters, we find the errors to be similar across all methods, although KL-MNL and MC-MNL, which use the multinomial logistic likelihood, give marginally more errors. However, with hyper-parameters optimized for KL-MNL, both KL-MNL and MC-MNL give less errors consistently. This suggests that model learning is better performed with a tight approximation to the marginal likelihood, as is provided by KL-MNL.

9.1.1 EFFECT OF PRIOR VARIANCE

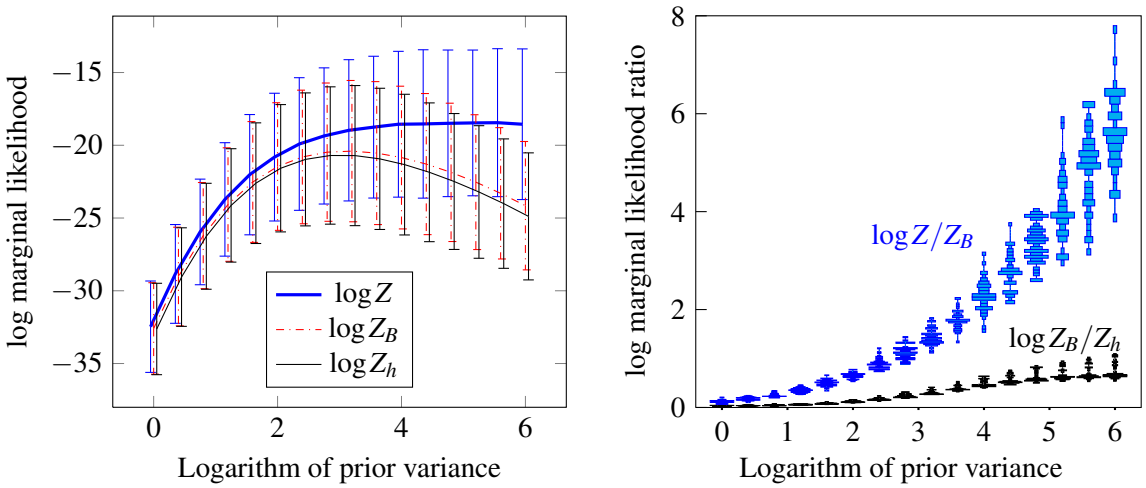
The results in Table 3 are where the latent processes have unit prior variance. Using the iris data set, we investigate the quality of our marginal likelihood bound when the prior variance increases. For each random partition, we fix the ARD hyper-parameters to that estimated for MF-MNP. The prior variance is then increased in steps. For each step, we obtain the marginal likelihoods using our variational inference and using importance sampling. The former is denoted by Z_h, and the latter by Z. Using Equation 3, we also obtain Z_B, which approximates the posterior with a Gaussian but computes the expected log-likelihood exactly. The Kullback-Leibler divergence is computed exactly, while $\mathcal{L} \stackrel{\text{def}}{=} \sum_{i=1}^{n} \ell_i(\mathbf{y}_i; q)$ is computed with Monte Carlo using $n_s = 100,000$ samples. For a sample $\mathbf{f}^{(s)}$ from the variational posterior, let $w^{(s)} \stackrel{\text{def}}{=} \sum_{i=1}^{n} \log p(\mathbf{y}_i|\mathbf{f}_i^{(s)})$. Then the Monte Carlo estimate of $\mathcal{L}$ is the sample mean $\bar{w}$ of the $w^{(s)}s$. We use the 99.9% confidence upper bound on Z_B by estimating $\mathcal{L}$ with $\bar{w} + 3.09\sigma/\sqrt{n_s}$, where σ^2 is the sample variance of the $w^{(s)}s$.[5]

Figure 1 gives plots against the prior variance of the latent process. The left figure (a) plots the log marginal likelihoods while the right figure (b) gives the violin plots of the log ratios of the marginal likelihoods. The plots show that the quality of the bounds Z_B and Z_h decreases with prior variance. This is largely due to the Gaussian approximation to the posterior, which is given by Z_B, rather than the approximation h to the expected likelihood: the violin plot for $\log Z_B/Z_h$ shows only slight increase as the prior variance increase, while the violin plot for $\log Z/Z_B$ increases more significantly. This illustrates the robustness of the approximation h to the expected log-likelihood. The deterioration of Gaussian approximation to the posterior is also present in binary Gaussian process classification (Nickisch and Rasmussen, 2008, Figure 3). The intuition is that a higher prior variance allows the posterior latent process more flexibility to become less Gaussian.

9.2 Comparing Models

The availability of fairly tight approximations to the exact posterior opens the opportunity for model comparison on each model's own merit without being confounded by the gap that results from approximation. In this section, we investigate the Gaussian process models in two areas: the choice of likelihood and the choice of prior correlation between the functions.

5. Since log-probability is unbounded, the true distribution of $w^{(s)}$ may not have finite variance. We eliminate this possibility empirically by verifying that the running sample variance has converged and that the estimated tail is not heavy (Koopman et al., 2009).



(a) Line plot of marginal likelihood

(b) Violin plot of log ratios of marginal likelihood

Figure 1: The quality of the lower bound on marginal likelihood for the iris data set by fixing the ARD hyper-parameters to those estimated for MF-MNP-$\boldsymbol{\theta}$ and then increasing the prior variance. Figure (a) plots the log marginal likelihood against the prior variance (on log-scale). The topmost curve $\log Z$ is for the marginal likelihood obtained using importance sampling; the middle curve $\log Z_B$ approximates the posterior of the latent process with a Gaussian; while the bottom curve $\log Z_h$ further approximates the expected log-likelihood. The error bars give 95% confidence interval computed over 50 random partitions of the data. The curves are translated slightly horizontally to reduce overlap in the error bars. Figure (b) plots the log marginal likelihood ratios against the prior variance (on log-scale) using the violin plot. The upper violin plot is for $\log Z/Z_B$ and the lower one is for $\log Z_B/Z_h$. These figures illustrate that the bound $\log Z_h$ becomes looser with increase in prior variance, and that this is mainly contributed by the Gaussian approximation to the posterior.

9.2.1 LIKELIHOOD

In Table 3, when we compare MC-MNP and MC-MNL on the set of hyper-parameters given by MF-MNP-$\boldsymbol{\theta}$, we see that the multinomial probit likelihood (MNP) fits the four training data sets better than the multinomial logistic likelihood (MNL). This difference can be explained by an equivalent model for each likelihood.

As outlined in Section 8, the Gaussian process latent model with covariance function $k^{\mathrm{x}}(\mathbf{x},\mathbf{x}')$ on the inputs and with the multinomial probit likelihood is equivalent to the model with covariance function $k^{\mathrm{x}}(\mathbf{x},\mathbf{x}')+\delta(\mathbf{x},\mathbf{x}')$ and the threshold likelihood function. This threshold likelihood partitions the space $\mathbb{R}^{nC}$ into orthants, one of which corresponds to the vector of observed data classes $\mathbf{y}$. Let us call this the $\mathbf{y}$-orthant. The marginal likelihood is hence the fraction of the prior probability mass in the $\mathbf{y}$-orthant. For a centered Gaussian prior, this fraction is determined by the correlation.

For multinomial logit likelihood, each auxiliary variable u^c is Gumbel distributed around f^c instead of Gaussian distributed; see Section 8. A moment matching approximation to the distribution of u^c is a Gaussian cent-red at f^c with variance $\pi^2/6$. Hence an approximation to the logit model is a Gaussian process latent model with covariance $k^{\mathrm{x}}(\mathbf{x},\mathbf{x}')+\delta(\mathbf{x},\mathbf{x}')\pi^2/6$ and with the threshold likelihood. As before, the marginal likelihood is the prior probability mass in the $\mathbf{y}$-orthant, and this is determined by the correlation.

The correlation functions of the equivalent models for the multinomial probit and multinomial logistic likelihoods are different. The former is obtained by removing the variance in $k^{\mathrm{x}}(\mathbf{x},\mathbf{x}')+\delta(\mathbf{x},\mathbf{x}')$, while the latter, $k^{\mathrm{x}}(\mathbf{x},\mathbf{x}')+\delta(\mathbf{x},\mathbf{x}')\pi^2/6$. One way to match the two correlation functions is to scale the original latent Gaussian process for the logit model by $\pi^2/6$, so that the equivalent covariance function becomes $\pi^2/6[k^{\mathrm{x}}(\mathbf{x},\mathbf{x}')+\delta(\mathbf{x},\mathbf{x}')]$. Consulting Figure 1a, the mean exact log marginal likelihood for the logit model on the iris data set is $\log Z\approx -0.28$ at $\log(\pi^2/6)\approx 1/2$ on the x-axis. This is consistent with the -27.82 under MC-MNP for the iris data set in Table 3.

9.2.2 PRIOR CORRELATION AMONG LATENT PROCESSES

It is common to assume prior independence among the latent functions for two reasons: to reduce computational complexity and to adhere to the principle of parsimony. In this section, we investigate if parsimony is a reason enough to exclude considering prior dependence among the latent functions. We evaluate on the four UCI data sets using the separable covariance structure in Equation 1.

For this evaluation, the covariance function on the inputs in $\mathbb{R}^d$ is the squared-exponential covariance function with equal length-scales along all the dimensions:

$$k^{\mathrm{x}}(\mathbf{x},\mathbf{x}')=\mathrm{sqexpiso}(\mathbf{x},\mathbf{x}';\sigma_{\mathrm{x}},\theta)\stackrel{\mathrm{def}}{=}\sigma_{\mathrm{x}}^2\exp\left(-\frac{1}{2}\sum_{j=1}^{d}\left(\frac{x_j-x_j'}{\theta}\right)^2\right). \tag{45}$$

We consider five models $\mathcal{M}_1,\ldots,\mathcal{M}_5$ of covariances K^{c} between the latent functions for the classes. In each model, we keep the total variance $\mathrm{tr}\,K^{\mathrm{c}}$ to be constant at C.[6] The first model $\mathcal{M}_1$ is where the latent functions are i.i.d., so K^{c} is the C-by-C identity matrix. The second is a diagonal matrix where the diagonal entries are positive and sum to C. Here, the latent functions remain a-priori independent, but they can have different variances while keeping the total variance the same as the first model. For the third and fourth models, we scale the K^{c} given by Equation 37 to have the same

6. The total variance in K^{c} is then scaled by the σ_{x}^2 in k^{x}, so the total variance of the latent process at each datum is $C\sigma_{\mathrm{x}}^2$.

total variance as the first two models:

$$\tilde{K}^{\mathrm{c}} = M - M\mathbf{1}\mathbf{1}^{\mathrm{T}}M/\mathbf{1}^{\mathrm{T}}M\mathbf{1} + I, \qquad K^{\mathrm{c}} = \frac{C}{\mathrm{tr}\,\tilde{K}^{\mathrm{c}}}\tilde{K}^{\mathrm{c}}. \tag{46}$$

The equation for $\tilde{K}^{\mathrm{c}}$ omits the weight for the identity matrix because the normalization in K^{c} makes this unnecessary. Model $\mathcal{M}_3$ sets M to be diagonal with positive diagonal entries; this is an attempt to approximate the correlation of multinomial or Dirichlet random variables. Model $\mathcal{M}_4$ allows M to be any positive semi-definite matrix. Finally, in the fifth model, K^{c} is any positive definite matrix with $\mathrm{tr}\,K^{\mathrm{c}} = C$; this allows the latent functions to be correlated arbitrarily.

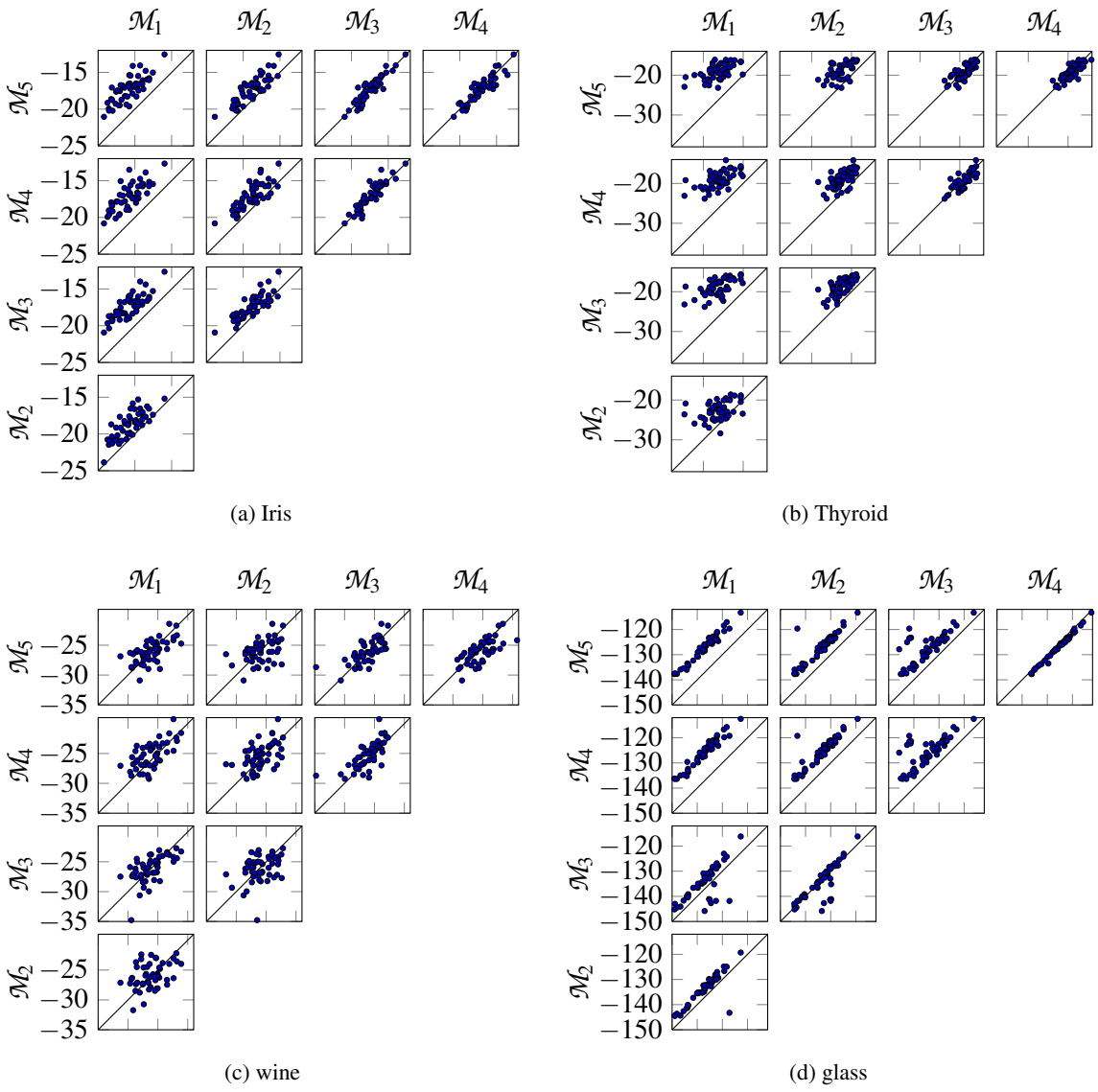
The first, third and fourth models satisfy the sum-to-zero property discussed in Section 5. Using $\mathcal{M}_i \supset \mathcal{M}_j$ to indicate that $\mathcal{M}_i$ is more expressive than $\mathcal{M}_j$, we have the ordering $\mathcal{M}_5 \supset \mathcal{M}_4 \supset \mathcal{M}_3 \supset \mathcal{M}_1$ and $\mathcal{M}_5 \supset \mathcal{M}_2 \supset \mathcal{M}_1$. The second model is not comparable with the third and fourth models in this ordering. The K^{c} for $\mathcal{M}_1$ is fixed and therefore parameter-free. The number of free parameters of K^{c} in models $\mathcal{M}_2$ and $\mathcal{M}_3$ are $C-1$. For $\mathcal{M}_4$ and $\mathcal{M}_5$, these are $C(C+1)/2-2$ and $C(C+1)/2-1$.

We estimate the parameters of the models in the following way. First, the hyper-parameters σ_{x} and θ for model $\mathcal{M}_1$ are optimized for the variational bound on marginal likelihood on the observed data. These two hyper-parameters are then considered fixed when optimizing matrix K^{c} for the other models using the variational bound.

After the hyper-parameters for each model have been estimated, we use sampling to obtain better estimates of the marginal likelihoods, errors and predictive likelihoods given the model and its hyper-parameters. This is done for each of the fifty partitions of the four UCI data sets. The sampling procedure is that outlined in the introduction to this section except for the glass data set, for which the Monte Carlo estimates to the marginal likelihood are lower than the variational lower bounds. There are two reasons for the lower estimates: (a) the sampling space is larger than in the other data sets because this data set has seven classes; and (b) for each data set partition, the prior variance is around 16, so the true posterior is conceivably less Gaussian and more different from the prior. To obtain Monte Carlo estimates that are better than the variational ones for this data set, we instead sample from the multivariate-t distribution that has covariance $2V$ instead of $2K$, where V is the covariance of the variational approximation.

Figure 2 gives a paired comparison between the models $\mathcal{M}_1, \ldots, \mathcal{M}_5$ based on their marginal likelihoods on the observed data. There are four sub-figures, one for each data set. Each graph is a scatter-plot, in which each point is for one partition of the data set named in the sub-caption, and the location of each point is the log marginal likelihoods of the two models indicated on the top and the left edge of the sub-figure. For a scatter-plot, if the points are mostly above the diagonal line, then model named on the left edge is better than the model named on the top edge. From Figure 2, we see no noticeable difference among the models for the wine data set, while we make the following four observations for the other data sets. (a) More free parameters generally results in better marginal likelihoods, as expected.[7] (b) Although $\mathcal{M}_2$ and $\mathcal{M}_3$ have the same number of free parameters, $\mathcal{M}_3$ generally gives better marginal likelihoods. (c) The marginal likelihoods of $\mathcal{M}_4$ and $\mathcal{M}_5$ are similar, showing that the additional free parameter in $\mathcal{M}_5$ over $\mathcal{M}_4$ is not useful. Observations (b) and (c) suggest that it is worthwhile to consider the sum-to-zero constraint, which is satisfied by $\mathcal{M}_3$ and $\mathcal{M}_4$ but not by $\mathcal{M}_2$ and $\mathcal{M}_5$.

7. There are two reasons why more free parameters is *not always* better. First, the hyper-parameters are optimized using the variational approximations and not the true marginal likelihoods. Although our approximations are rather tight, the hyper-parameters may be sensitive to the remaining gaps in the approximations. Second, the marginal likelihood surface can be multi-modal with respect to the hyper-parameters, hence gradient ascent can be stuck at local maxima.



(a) Iris (b) Thyroid

(c) wine (d) glass

Figure 2: Paired comparisons of the marginal likelihood between five models of prior correlations K^{c} between the latent functions: $\mathcal{M}_1$ gives i.i.d. latent functions; $\mathcal{M}_2$ gives independent latent functions with different variances; $\mathcal{M}_3$ allows the functions to be correlated using Equation 46 where M is diagonal; $\mathcal{M}_4$ also uses Equation 46 but allows M to be any positive semi-definite matrix; and $\mathcal{M}_5$ allows the functions to be correlated arbitrarily. For each model, K^{c} is scaled such that the total variance $\operatorname{tr} K^{\mathrm{c}}$ is constant C. Each figure is for the data set indicated in its caption. Each graph in a figure plots the log marginal likelihood ($\log Z$) of the model named at the left edge of the figure versus that named at the top edge. Each point in the scatter-plot is for one of the fifty random partitions of the data set. To ease comparison, the $x = y$ line is plotted in each graph. For example, each point in top left graph of Figure (a) is at the location (x, y), where x (resp. y) is the $\log Z$ for $\mathcal{M}_1$ (resp. $\mathcal{M}_5$) on one partition of the iris data set. All the points in this graph are above the $x = y$ line, so $\mathcal{M}_5$ gives better marginal likelihood than $\mathcal{M}_1$ for all the partitions.

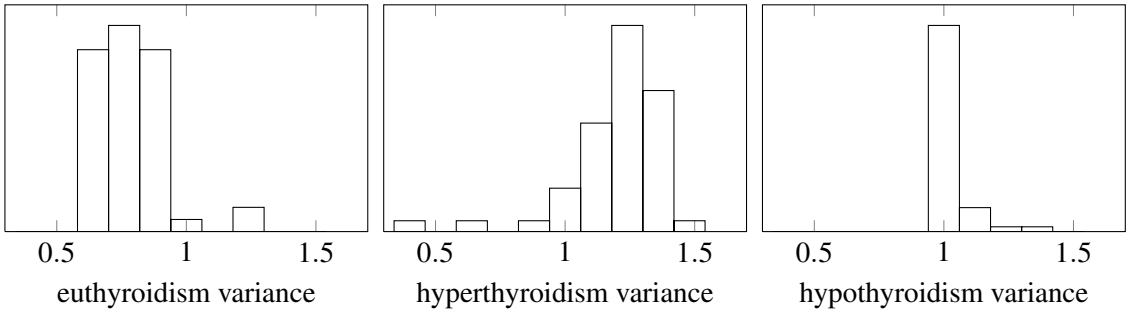


Figure 3: The histograms of the prior variances of the latent functions in K^{c} estimated from the thyroid data under model $\mathcal{M}_4$. From left to right, we have histograms for euthyroidism, hyperthyroidism and hypothyroidism.

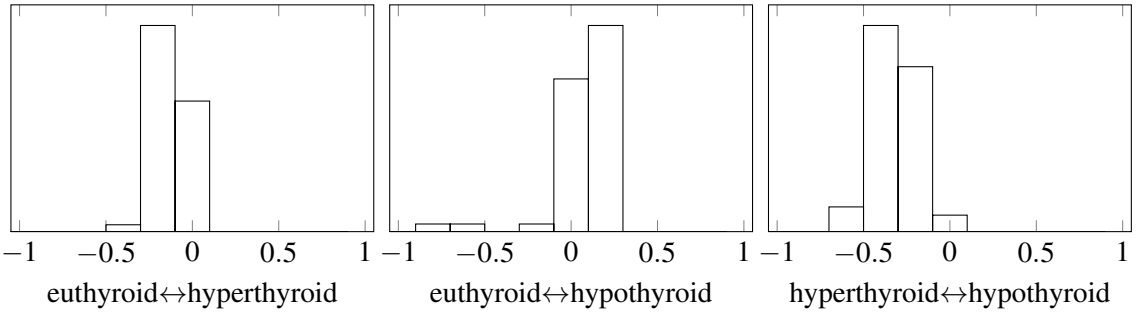


Figure 4: The histograms of the prior correlation of the latent functions estimated from the thyroid data under model $\mathcal{M}_4$. From left to right, we have histograms for the correlation between euthyroidism and hyperthyroidism, between euthyroidism and hypothyroidism, and between hyperthyroidism and hypothyroidism.

For the errors and predictive likelihoods, we observe no significant difference among the models, based on plots similar to those in Figure 2. For the purpose of prediction then, it seems sufficient to rely on the likelihood to provide the necessary posterior coupling between the latent functions. Nonetheless, we believe there may still be applications where prior coupling between the latent functions is helpful in prediction.

More insights into the various models can be obtained by looking at the estimated variances and correlations between the latent functions. As an example we shall use the thyroid data with model $\mathcal{M}_4$; examination with model $\mathcal{M}_3$ or model $\mathcal{M}_5$ gives similar conclusions. The task for the thyroid data is to predict the state of a subject's thyroid given the results of five different laboratory tests. This state can be one of three classes: euthyroidism (having normal functioning thyroid), hyperthyroidism (having overactive thyroid) and hypothyroidism (having underactive thyroid). Figure 3 gives the histogram of the prior variances of the latent functions in K^{c} estimated by $\mathcal{M}_4$ over the fifty different partitions. From left to right, the histograms are for euthyroidism, hyperthyroidism and hypothyroidism. Each histogram is concentrated around a single mode. Bearing in mind that we have constrained the total variance in K^{c} to be 3, the evidence in the data suggests that class hy-

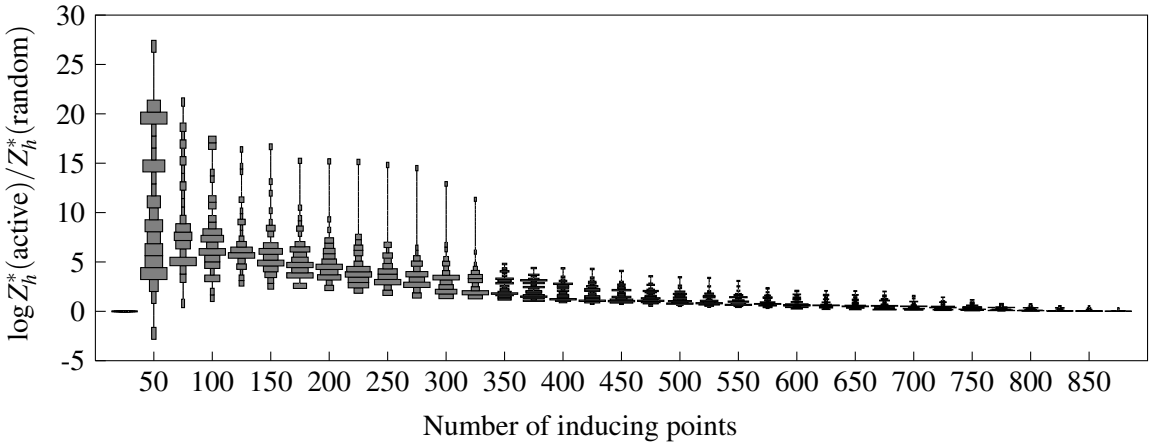


Figure 5: The performance of active selection over random selection as the number of inducing point is incremented in steps of 25. Each histogram is for the log ratios of Z_h^* for active selection to random selection. The plot shows that active selection is better than random selection, though the advantage decreases with the number of inducing points.

perthyroidism varies more than class hypothyroidism, which varies more than class euthyroidism. Figure 4 gives the histogram of the correlations between the latent functions. From left to right, we have the correlations between euthyroidism and hyperthyroidism, between euthyroidism and hypothyroidism, and between hyperthyroidism and hypothyroidism. As for the variances, each histogram is concentrated around a single mode. The most significant result is the right histogram, which shows hyperthyroidism and hypothyroidism to be negatively correlated. This is intuitive: the two classes correspond to overproduction and underproduction of thyroid hormones respectively.

9.3 Active Inducing Set Selection

Sparse approximation is commonly used for efficient inference in large data sets. The quality of this approximation is dependent on the inducing set. In this section, we evaluate the effectiveness of criteria d_2 and d_3 (Equations 42 and 43 in Section 6.1) in selecting the inducing set actively. We do this by comparing with random selection on the glass data set.

We use the usqexpard covariance function (44) on the inputs and assume that the latent functions are i.i.d. For each training set partition, we fix the hyper-parameters to those optimized for our variational lower bound using the entire training set; these are the KL-MNL-$\boldsymbol{\theta}$ hyper-parameters estimated in Section 9.1. Given the training set partition and the hyper-parameters, the random approach selects δ sites to be added to the inducing set at each iteration. The active approach begins with the same δ random sites as the random approach, but subsequent choices of the δ sites are selected based on the d_2 and d_3 criteria. For each random variable induced by the training set, we use d_3 with subsample set of size $|\mathcal{S}|$. Next, we compute the d_2 scores of the t random variables with the highest d_3 scores. The δ random variables having the highest d_2 scores computed in this manner will be added to the inducing set. We use $\delta = 25$, $|\mathcal{S}| = 5$ and $t = 40$ in the experiment.

The optimized variational lower bound Z_h^* on the marginal likelihood for random selection is computed at each iteration for each training set partition. The same is computed for active selection. Figure 5 gives the violin plot of the log ratios of the Z_h^* for the active selection to the Z_h^* for the random selection. The horizontal axis gives the size of the inducing set—there are 896 potential inducing sites from the 7 types of glass and the 128 training $\mathbf{x}$s. The histogram at each iteration is over the fifty random training set partitions.

At the first iteration with 25 inducing sites, the ratio is zero because both the random selection and the active selection begin with the same 25 random sites. At the second iteration with an additional 25 inducing sites, active selection usually provides higher Z_h^*, but it is possible for active selection to be worse than random selection. This is because the d_2 and d_3 criteria are designed for single inducing sites, so they are not optimal for selecting more than one site—25 in this experiment—at once. Nevertheless, in subsequent iterations, active selection always provides higher Z_h^* than random selection. As the size of the inducing set increases, the benefit from active selection decreases because the value of any inducing site decreases.

9.4 Comparing with Single-machine Multi-class Support Vector Machines

Support vector machines (SVMs, Vapnik, 1998) are popular for classification and they have been known to give good classification accuracies in general. Although originally formulated for binary classification, several extensions have been proposed for multi-class classification. These extensions can be grouped roughly into two: one is to transform the multi-class problem into several binary class problems together with a decoding step; the other, called the single-machine approach, is to solve a single optimization problem for multiple classes, keeping to the broad principles of structural risk minimization (Vapnik, 1998). Comparisons between the two groups have been done by Rifkin and Klautau (2004). In this section, we compare our proposed variational approximation on multi-class Gaussian processes to four different single-machine multi-class SVMs using the MSVMpack package (Lauer and Guermeur, 2011). We denote the four single machines by WW (Vapnik, 1998; Weston and Watkins, 1999), CS (Crammer and Singer, 2001), LLW (Lee et al., 2004) and MSVM2 (Guermeur and Monfrini, 2011). The comparison is on the four UCI data sets.

For both Gaussian processes and SVMs, we use the sqexpiso covariance function or kernel (45). The signal variance σ_{x}^2 in the Gaussian process covariance function corresponds to the soft-margin trade-off parameter in the SVM objective functions, usually denoted by C. For the multi-class Gaussian processes, the parameters of the covariance function are estimated by optimizing our variational lower bound on the marginal likelihood. For the single-machine SVMs, the parameters C and θ are estimated from a grid $(\log_{10} C, \theta) \in \{-2,-1,0,1,2,3\} \times \{0.1,1,5,10,15\}$ using five-fold cross validation on the training set.[8] This is repeated for each of the fifty train/test set partitions.

Table 4 gives the means and standard deviations of the errors over the fifty random train/test set partitions. The results for WW, CS and LLW are consistent with those reported by Weston and Watkins (1999), Hsu and Lin (2002) and Lee et al. (2004), when we take into perspective that their results are for 90%/10% train/test splits instead of the 60%/40% here. Comparing the errors for the Gaussian processes under column KL-MNL with the errors for the single-machine SVMs, we see that the Gaussian processes give better performances on the average. One reason is that model learning is achieved using continuous optimization with Gaussian processes, while only discrete

8. When MSVMpack does not seem to converge on its stopping criterion for a parameter pair (C, θ), the learning is forced to terminate for that pair, and the validation score is computed based on the model at the point of termination.

		Single-machine multi-class support vector machines			
Data set	KL-MNL	WW	CS	LLW	MSVM2
Iris	2.18 ± 1.42	3.02 ± 2.51	3.22 ± 4.34	2.74 ± 1.45	2.88 ± 1.47
Thyroid	3.54 ± 1.68	4.42 ± 3.45	4.82 ± 1.89	5.28 ± 2.08	5.58 ± 2.56
Wine	1.40 ± 0.90	1.40 ± 0.69	2.20 ± 1.34	1.62 ± 1.03	1.50 ± 0.95
Glass	27.44 ± 3.78	29.30 ± 10.70	28.70 ± 3.17	29.54 ± 9.72	29.42 ± 3.59

Table 4: Errors of variational multinomial logit Gaussian process (column *KL-MNL*) and four variants of single-machine multi-class SVMs. The means and standard deviations of the errors over fifty partitions are reported. The sqexpiso covariance function/kernel (45) is used on the inputs, and there is no inter-class correlations between the functions.

	Errors		Number of support vectors				
Data set	Sparse KL-MNL	SVM-WW	min	Q_1	Q_2	Q_3	max
Iris	2.14 ± 1.39	3.02 ± 2.51	12	17	20	24	36
Thyroid	7.38 ± 5.22	4.42 ± 3.45	2	3	4	7	39
Wine	1.32 ± 0.91	1.40 ± 0.69	4	10	25	34	37
Glass	28.42 ± 4.05	29.30 ± 10.70	8	18	56	65	75

Table 5: Errors of sparse variational multinomial logit Gaussian process and the WW variant of multi-class SVM. The means and standard deviations of the errors over fifty partitions are reported. The sqexpiso covariance function/kernel (45) is used on the inputs, and there is no inter-class correlations. The number of inducing variables for the sparse approximation is the number of support vectors given by WW times the number of classes. The last five columns give the statistics of number of support vectors over the partitions. Column *SVM-WW* duplicates column *WW* in Table 4.

optimization on the grid is used with the SVMs. Hence, the Gaussian processes can give finer parameter estimates.

The above uses the full (or non-sparse) approximation to the Gaussian process model. We also experiment with the sparse approximation. For each data set and each partition of the set, the target number inducing variables is fixed to the number of support vectors selected by WW multiplied by the number of classes. The initial inducing set is C randomly chosen variables. Inducing variables are added using the strategy described in Section 9.3, but now with $\delta = C$, $|\mathcal{S}| = 5$ and $t = 20$. This expansion of the inducing set is alternated with one gradient-line-search to optimize the hyper-parameters of the model. After all the inducing variables are added, the hyper-parameters are further optimized. Table 5 reports the results repeated over the fifty partitions for each data set. The table also gives the minimum, maximum and quartiles (Q_1, Q_2, Q_3) of the number of support vectors.

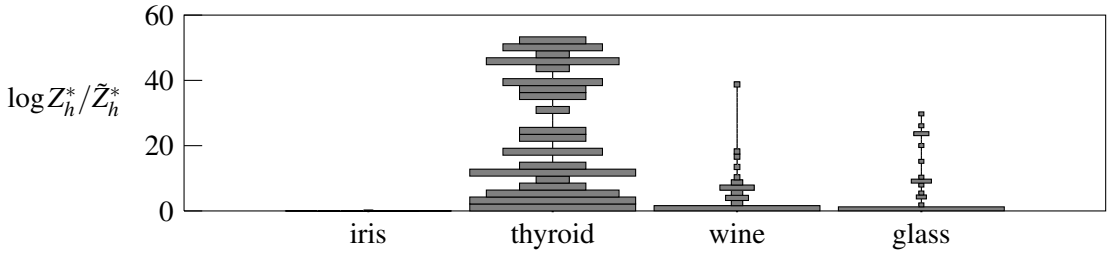


Figure 6: The quality of sparse approximations, at the same level of sparsity as the WW variant of SVM. Each histogram is for the log ratios of Z_h^* for the full model to $\tilde{Z}_h^*$ for the sparse model. The approximation is mostly tight for the iris, wine and glass data sets, but is loose for the thyroid data. The histogram for the iris data concentrated at 0, so it is barely visible. The hyper-parameters for Z_h^* and for $\tilde{Z}_h^*$ are different.

From Table 5, we see that the sparse Gaussian process model with active selection of inducing set compares favourably with the WW variant of SVM in terms of errors for three of the four data sets. The sparse Gaussian process model gives significantly more errors for the thyroid data. We believe there are two reasons for this. First, most of the fifty repetitions for the thyroid data have very small number inducing variables: the median is $4 \times 3 = 12$. Second, the sparsity in the Gaussian process is an *imposed* approximation to the full Gaussian process, while the sparsity in the SVM is a direct consequence of its objective function. Therefore, limited to a median of only 12 inducing variables, the sparse approximation is unsatisfactory. This is reflected in Figure 6, which gives the violin plot of the log ratios of the marginal likelihood for the full model to the sparse model. From the figure, we see that approximation of the Gaussian process model with the same level of sparsity as WW is unsatisfactory for the thyroid data. Finally, we remark that the full Gaussian process model gives 3.54 ± 1.68 errors; see Table 4.

10. Conclusion and Discussion

We have introduced a tractable variational approximation to the multinomial logit Gaussian processes for multi-class classification in Section 2.3, and we have provided the necessary updates to optimize this approximation in Section 3. Empirical results in Section 9.1 have indicated that our approximation is very faithful to the exact distribution, in contrast to the variational mean-field approximation (Girolami and Rogers, 2006). One key to the success of this approximation is Theorem 6, which gives a variational lower bound on the expected log-likelihood at each observation. In addition, bounds on the train data marginal likelihood and test data predictive likelihoods have been given in Sections 2.3.2 and 2.3.3, and these bounds have been shown to be supported by empirical results in Section 9.1.

In Section 4, the proposed variational approximation has been combined with the sparse variational approximation approach previously advocated for regression (Titsias, 2009a). This sparse approximation to the multinomial logit Gaussian processes has the property that incremental increases in the inducing set will lead to tighter bounds on the marginal likelihood. This property has been exploited in Section 6.1 to derive scores for potential inducing sites. An active selection

strategy making greedy use of these scores has been compared favorably with random selection in Section 9.3.

The present paper is mostly independent of the covariance structure of the Gaussian process. Nevertheless, at various points, we have focused on the case where the covariance is separable into the covariance on the inputs and the covariance on the classes. Such separable covariance has been investigated previously in the context of multi-task Gaussian process regression (see, for example, Bonilla et al., 2008). In Section 5, we have looked into the cases where the optimized variational posterior satisfies the sum-to-zero property; this property is also present in many single-machine multi-class SVMs. In Section 9.2.2, we have compared several models of prior correlation between the latent functions. Although the experimental results are neither general nor conclusive against or for $K^{\mathrm{c}} = I$, further investigation into the thyroid data has suggested that useful knowledge can indeed be extracted if inter-latent-function correlations are permitted.

There are several possibilities building upon and extending this work. From the model perspective, it is worthwhile to have more interesting models in which latent functions can be related than, for example, the separable covariance of Equation 1. For this, covariance models developed for multi-task learning in the regression setting can be assessed for multinomial logit Gaussian process. Here, two questions specific to multi-class classification are of interest. First, should one consider models where a pair of latent functions are allowed to be positively correlated? On the one hand, the classes are *mutually exclusive*, so an increase in the probability of one class necessarily entails a decrease in the probability of another class when the probability of other classes are held constant; hence we can expect negative correlations between the latent functions. On the other hand, if there is a natural *hierarchical* structure to the classes, then the probability of two classes can rise in tandem against the probability of the other classes; hence we may also find positive correlations. The second question is: should the set of length-scales of the latent functions be the same? To argue for the same set of length-scales, one may say that a *single* property of the given object $\mathbf{x}$ is being predicted. The counter argument is that there are *different* values for this property, and the latent function for each value may demand its own set of length-scales.

From the variational approximation perspective, further constraints can be placed on the any of the variational parameters: $\mathbf{m}$, V, the $\mathbf{b}_i$s and the S_is. Some constraints will lead to more efficient algorithms though with less faithful approximations, and trade-offs between the two conflicting goals will have to be examined. From a purely algorithmic perspective, more efficient updates than the ones presented in Section 3 can be explored.

Theorem 6 gives a variational lower bound on the expected log-likelihood at each observation. We have seen that it is rather tight on the average in Section 9.1. This bound can be applied to on-line multi-class classification under the assumed density filtering framework, following a prior work on sparse on-line binary classification (Csató and Opper, 2002).

Acknowledgments

We thank Chris Williams, Hai Leong Chieu and the reviewers for their helpful comments and suggestions. We are grateful to DSO National Laboratories, Singapore, for the permission to publish this work. This work is funded by DR-Tech, Singapore.

Appendix A. Mathematical Preliminaries

We provide general results required in the proofs for the main results of this paper.

A.1 Gaussians

Lemma 18 *Let $\mathbf{x}_1 \in \mathbb{R}^{n_1}$ and $\mathbf{x}_2 \in \mathbb{R}^{n_2}$ be random vectors with two jointly normal distributions $p(\mathbf{x}_1, \mathbf{x}_2) \stackrel{\text{def}}{=} \mathcal{N}(\mathbf{n}, U)$ and $q(\mathbf{x}_1, \mathbf{x}_2) \stackrel{\text{def}}{=} \mathcal{N}(\mathbf{m}, V)$, where the parameters are partitioned as*

$$\mathbf{n} \stackrel{\text{def}}{=} \begin{pmatrix} \mathbf{n}_1 \\ \mathbf{n}_2 \end{pmatrix}, \qquad U \stackrel{\text{def}}{=} \begin{pmatrix} U_{11} & U_{12} \\ U_{21} & U_{22} \end{pmatrix}, \qquad \mathbf{m} \stackrel{\text{def}}{=} \begin{pmatrix} \mathbf{m}_1 \\ \mathbf{m}_2 \end{pmatrix}, \qquad V \stackrel{\text{def}}{=} \begin{pmatrix} V_{11} & V_{12} \\ V_{21} & V_{22} \end{pmatrix}.$$

The difference between the Kullback-Leibler divergences on $\mathbf{x} \stackrel{\text{def}}{=} (\mathbf{x}_1^{\mathrm{T}}, \mathbf{x}_2^{\mathrm{T}})^{\mathrm{T}}$ and $\mathbf{x}_1$ is

$$\begin{aligned} &\mathrm{KL}(p(\mathbf{x}) \| q(\mathbf{x})) - \mathrm{KL}(p(\mathbf{x}_1) \| q(\mathbf{x}_1)) \\ &\qquad = -\frac{n_2}{2} - \frac{1}{2} \log \left| V_{2|1}^{-1} U_{2|1} \right| + \frac{1}{2} \operatorname{tr} V_{2|1}^{-1} \left(U_{2|1} + W U_{11}^{-1} W^{\mathrm{T}} \right) + \frac{1}{2} \mathbf{t}^{\mathrm{T}} V_{2|1}^{-1} \mathbf{t}, \end{aligned}$$

where

$$\begin{aligned} U_{2|1} &\stackrel{\text{def}}{=} U_{22} - U_{21} U_{11}^{-1} U_{12}, & V_{2|1} &\stackrel{\text{def}}{=} V_{22} - V_{21} V_{11}^{-1} V_{12}, \\ W &\stackrel{\text{def}}{=} U_{21} - V_{21} V_{11}^{-1} U_{11}, & \mathbf{t} &\stackrel{\text{def}}{=} (\mathbf{m}_2 - \mathbf{n}_2) - V_{21} V_{11}^{-1} (\mathbf{m}_1 - \mathbf{n}_1). \end{aligned}$$

Proof The conditional distributions $p(\mathbf{x}_2|\mathbf{x}_1)$ and $q(\mathbf{x}_2|\mathbf{x}_1)$ have means

$$\mathbf{n}_{2|1} \stackrel{\text{def}}{=} \mathbf{n}_2 + U_{21} U_{11}^{-1} (\mathbf{x}_1 - \mathbf{n}_1), \qquad \mathbf{m}_{2|1} \stackrel{\text{def}}{=} \mathbf{m}_2 + V_{21} V_{11}^{-1} (\mathbf{x}_1 - \mathbf{m}_1)$$

and covariances $U_{2|1}$ and $V_{2|1}$. The difference between the Kullback-Leibler divergences can be derived through the Kullback-Leibler divergence of these conditionals:

$$\begin{aligned} &\mathrm{KL}(p(\mathbf{x}) \| q(\mathbf{x})) - \mathrm{KL}(p(\mathbf{x}_1) \| q(\mathbf{x}_1)) = \int p(\mathbf{x}) \log \frac{p(\mathbf{x})/p(\mathbf{x}_1)}{q(\mathbf{x})/q(\mathbf{x}_1)} \mathrm{d}\mathbf{x} = \int p(\mathbf{x}) \log \frac{p(\mathbf{x}_2|\mathbf{x}_1)}{q(\mathbf{x}_2|\mathbf{x}_1)} \mathrm{d}\mathbf{x} \\ &= \frac{1}{2} \int p(\mathbf{x}_1) \left[-n_2 - \log \left| V_{2|1}^{-1} U_{2|1} \right| + \operatorname{tr} V_{2|1}^{-1} U_{2|1} + \left(\mathbf{m}_{2|1} - \mathbf{n}_{2|1} \right)^{\mathrm{T}} V_{2|1}^{-1} \left(\mathbf{m}_{2|1} - \mathbf{n}_{2|1} \right) \right] \mathrm{d}\mathbf{x}_1 \end{aligned}$$

In the last expression above, only the final quadratic term in the integrand depends on $\mathbf{x}_1$, so we can move the other terms out of the integral. For this final term, its integral under $p(\mathbf{x}_1)\mathrm{d}\mathbf{x}_1$ is $\operatorname{tr} V_{2|1}^{-1} \left(W U_{11}^{-1} W^{\mathrm{T}} \right) + \mathbf{t}^{\mathrm{T}} V_{2|1}^{-1} \mathbf{t}$. ■

A.2 Matrix

Proposition 19 *For positive semi-definite matrices A and B of the same order, we have $B \preceq A$ implies* $\mathrm{null}(A) \subseteq \mathrm{null}(B)$.

Proof Let $\mathbf{x} \in \mathrm{null}(A)$, then $A\mathbf{x} = \mathbf{0}$ by definition. Hence $\mathbf{x}^{\mathrm{T}} A \mathbf{x} = 0$. Since $0 \preceq B \preceq A$, we have $0 \leq \mathbf{x}^{\mathrm{T}} B \mathbf{x} \leq \mathbf{x}^{\mathrm{T}} A \mathbf{x} = 0$. Therefore $\mathbf{x}^{\mathrm{T}} B \mathbf{x} = 0$. This means $B\mathbf{x} = 0$, or $\mathbf{x} \in \mathrm{null}(B)$ (Horn and Johnson, 1985, Section 7.5, Problem 14). Thus $\mathbf{x} \in \mathrm{null}(A) \Longrightarrow \mathbf{x} \in \mathrm{null}(B)$. ■

Lemma 20 *(Matrix determinant lemma) For an n-by-n non-singular matrix A and vectors* $\mathbf{u}$ *and* $\mathbf{v}$ *in* $\mathbb{R}^n$, $|A+\mathbf{u}\mathbf{v}^{\mathrm{T}}| = (1+\mathbf{v}^{\mathrm{T}}A^{-1}\mathbf{u})|A|$.

Lemma 21 *Under the setting in Lemma 20*

$$|A+\mathbf{u}\mathbf{v}^{\mathrm{T}}+\mathbf{v}\mathbf{u}^{\mathrm{T}}| = \left[(1+\mathbf{v}^{\mathrm{T}}A^{-1}\mathbf{u})(1+\mathbf{u}^{\mathrm{T}}A^{-1}\mathbf{v}) - \mathbf{u}^{\mathrm{T}}A^{-1}\mathbf{u}\,\mathbf{v}^{\mathrm{T}}A^{-1}\mathbf{v})\right]|A|.$$

Proof Apply Lemma 20 twice; then use the Sherman-Morrison formula on $(A+\mathbf{u}\mathbf{v}^{\mathrm{T}})^{-1}$. ■

Corollary 22 *If A is also symmetric, then*

$$|A+\mathbf{u}\mathbf{v}^{\mathrm{T}}+\mathbf{v}\mathbf{u}^{\mathrm{T}}| = \left[(1+\mathbf{u}^{\mathrm{T}}A^{-1}\mathbf{v})^2 - \mathbf{u}^{\mathrm{T}}A^{-1}\mathbf{u}\,\mathbf{v}^{\mathrm{T}}A^{-1}\mathbf{v})\right]|A|.$$

Lemma 23 *Further, with* $a \in \mathbb{R}$*, we have*

$$|A+a\mathbf{u}\mathbf{u}^{\mathrm{T}}+\mathbf{u}\mathbf{v}^{\mathrm{T}}+\mathbf{v}\mathbf{u}^{\mathrm{T}}| = \left(\left(1+\mathbf{u}^{\mathrm{T}}A^{-1}\mathbf{v}\right)^2 - \mathbf{u}^{\mathrm{T}}A^{-1}\mathbf{u}\,\mathbf{v}^{\mathrm{T}}A^{-1}\mathbf{v} + a\mathbf{u}^{\mathrm{T}}A^{-1}\mathbf{u}\right)|A|.$$

Proof Let $X \stackrel{\text{def}}{=} A+a\mathbf{u}\mathbf{u}^{\mathrm{T}}$ and $b \stackrel{\text{def}}{=} 1+a\mathbf{u}^{\mathrm{T}}A^{-1}\mathbf{u}$. We apply Corollary 22, then we use Lemma 20 on $|X|$ and the Sherman-Morrison formula on X^{-1}:

$$\begin{aligned}
&\left[(1+\mathbf{u}^{\mathrm{T}}X^{-1}\mathbf{v})^2 - \mathbf{u}^{\mathrm{T}}X^{-1}\mathbf{u}\,\mathbf{v}^{\mathrm{T}}X^{-1}\mathbf{v})\right]|X| \\
&= |A|b\left[\left(1+\frac{1}{b}\mathbf{u}^{\mathrm{T}}A^{-1}\mathbf{v}\right)^2 - \frac{1}{b}\mathbf{u}^{\mathrm{T}}A^{-1}\mathbf{u}\left(\mathbf{v}^{\mathrm{T}}A^{-1}\mathbf{v} - \frac{a}{b}\left(\mathbf{u}^{\mathrm{T}}A^{-1}\mathbf{v}\right)^2\right)\right] \\
&= |A|b\left[1+\frac{2}{b}\mathbf{u}^{\mathrm{T}}A^{-1}\mathbf{v} + \frac{1}{b}\left(\mathbf{u}^{\mathrm{T}}A^{-1}\mathbf{v}\right)^2 - \frac{1}{b}\mathbf{u}^{\mathrm{T}}A^{-1}\mathbf{u}\mathbf{v}^{\mathrm{T}}A^{-1}\mathbf{v}\right] \\
&= |A|\left[(1+\mathbf{u}^{\mathrm{T}}A^{-1}\mathbf{v})^2 + a\mathbf{u}^{\mathrm{T}}A^{-1}\mathbf{u} - \mathbf{u}^{\mathrm{T}}A^{-1}\mathbf{u}\mathbf{v}^{\mathrm{T}}A^{-1}\mathbf{v}\right].
\end{aligned}$$

The second step in the derivation applies the identity $1-(a/b)\mathbf{u}^{\mathrm{T}}A^{-1}\mathbf{u} = 1/b$. ■

Theorem 24 *(Matrix quadratic equation, a special case of Potter 1966). Let A and B be two real symmetric n-by-n matrices such that* $A+B^2$ *is positive semi-definite and B is positive definite. Then the positive definite solution to the equation* $-X^2+BX+XB+A=0$ *is* $X = P\Lambda^{\frac{1}{2}}P^{\mathrm{T}} + B$*, where* $P\Lambda P^{\mathrm{T}}$ *is the eigen-decomposition of* $A+B^2$.

Proof The solution X can be proved by direct substitution into the equation, or by completing the square, or by following a construction due to Potter (1966). For positive definiteness, since $B \succ 0$, we only require that $(A+B^2)$ is positive semi-definite. ■

Appendix B. Bounds

This appendix provides the proofs for the bounds stated in the main text.

B.1 Variational Lower Bound on the Marginal Likelihood

We derive the variational lower bounds $\tilde{Z}_B$ on the true marginal likelihood in the sparse case. Using Jensen's inequality, we have

$$\log p(\mathbf{y}) = \log \int p(\mathbf{y},\mathbf{f},\mathbf{z})\,\mathrm{d}\mathbf{z}\,\mathrm{d}\mathbf{f} = \log \int p(\mathbf{y},\mathbf{f},\mathbf{z}) \frac{q(\mathbf{f},\mathbf{z}|\mathbf{y})}{q(\mathbf{f},\mathbf{z}|\mathbf{y})}\,\mathrm{d}\mathbf{z}\,\mathrm{d}\mathbf{f} \geq \log \tilde{Z}_B,$$
$$\text{where} \qquad \log \tilde{Z}_B \stackrel{\text{def}}{=} \int q(\mathbf{f},\mathbf{z}|\mathbf{y}) \log \frac{p(\mathbf{y},\mathbf{f},\mathbf{z})}{q(\mathbf{f},\mathbf{z}|\mathbf{y})}\,\mathrm{d}\mathbf{z}\,\mathrm{d}\mathbf{f}. \tag{47}$$

Given the model, the joint distribution $p(\mathbf{y},\mathbf{f},\mathbf{z})$ factorizes into $p(\mathbf{y}|\mathbf{f})p(\mathbf{f}|\mathbf{z})p(\mathbf{z})$ exactly; there is no approximation involved. For the approximate posterior $q(\mathbf{f},\mathbf{z}|\mathbf{y})$, however, the factorization $q(\mathbf{f},\mathbf{z}|\mathbf{y}) = p(\mathbf{f}|\mathbf{z})q(\mathbf{z}|\mathbf{y})$ is assumed. Using these two factorizations, we have

$$\begin{aligned}
\log \tilde{Z}_B &= \int p(\mathbf{f}|\mathbf{z})q(\mathbf{z}|\mathbf{y}) \log \frac{p(\mathbf{y}|\mathbf{f})p(\mathbf{z})}{q(\mathbf{z}|\mathbf{y})} \mathrm{d}\mathbf{z}\mathrm{d}\mathbf{f} \\
&= \int q(\mathbf{z}|\mathbf{y}) \left[\log \frac{p(\mathbf{z})}{q(\mathbf{z}|\mathbf{y})} + \int p(\mathbf{f}|\mathbf{z}) \log p(\mathbf{y}|\mathbf{f}) \mathrm{d}\mathbf{f} \right] \mathrm{d}\mathbf{z} \\
&= -\mathrm{KL}(q(\mathbf{z}|\mathbf{y}) \| p(\mathbf{z})) + \int q(\mathbf{f}|\mathbf{y}) \log p(\mathbf{y}|\mathbf{f}) \mathrm{d}\mathbf{f},
\end{aligned}$$

where $q(\mathbf{f}|\mathbf{y}) \stackrel{\text{def}}{=} \int q(\mathbf{f},\mathbf{z}|\mathbf{y})\mathrm{d}\mathbf{z}$. Since the joint likelihood factorizes across the n data points, this is also $\log \tilde{Z}_B = -\mathrm{KL}\left(q(\mathbf{z}|\mathbf{y}) \,\|\, p(\mathbf{z})\right) + \sum_{i=1}^{n} \ell_i(\mathbf{y}_i; q)$, where $\ell_i(\mathbf{y}_i; q) \stackrel{\text{def}}{=} \int q(\mathbf{f}_i|\mathbf{y}) \log p(\mathbf{y}_i|\mathbf{f}_i)\,\mathrm{d}\mathbf{f}_i$.

The bound Z_B in the non-sparse case can be obtained in a similar manner with

$$\log Z_B \stackrel{\text{def}}{=} \int q(\mathbf{f}|\mathbf{y}) \log \frac{p(\mathbf{y},\mathbf{f})}{q(\mathbf{f}|\mathbf{y})}\,\mathrm{d}\mathbf{f}. \tag{48}$$

B.1.1 Relation between Bounds for Non-sparse and Sparse Approximations

We show that the optimized non-sparse bound $\log Z_B^*$ is not smaller than the optimized sparse bound $\log \tilde{Z}_B^*$. We begin by constraining the approximate posterior in $\log Z_B$:

$$\log Z_B^* \stackrel{\text{def}}{=} \max_{q(\mathbf{f}|\mathbf{y})} \log Z_B \geq \max_{q(\mathbf{z}|\mathbf{y})} \log Z_B \quad \left(\text{where } q(\mathbf{f}|\mathbf{y}) = \textstyle\int p(\mathbf{f}|\mathbf{z})q(\mathbf{z}|\mathbf{y})\mathrm{d}\mathbf{z}\right). \tag{49}$$

We introduce an arbitrary distribution r on $\mathbf{z}$ and use Jensen's inequality to get

$$\log p(\mathbf{y},\mathbf{f}) = \log \int p(\mathbf{y},\mathbf{f},\mathbf{z})\mathrm{d}\mathbf{z} = \log \int r(\mathbf{z}) \frac{p(\mathbf{y},\mathbf{f},\mathbf{z})}{r(\mathbf{z})}\mathrm{d}\mathbf{z} \geq \int r(\mathbf{z}) \log \frac{p(\mathbf{y},\mathbf{f},\mathbf{z})}{r(\mathbf{z})}\mathrm{d}\mathbf{z}.$$

The above inequality is substituted into $\log Z_B$ through its definition (48), and the result is applied to the leftmost expression in (49):

$$\log Z_B^* \geq \max_{q(\mathbf{z}|\mathbf{y})} \int q(\mathbf{f}|\mathbf{y})r(\mathbf{z}) \log \frac{p(\mathbf{y},\mathbf{f},\mathbf{z})}{q(\mathbf{f}|\mathbf{y})r(\mathbf{z})}\,\mathrm{d}\mathbf{z}\,\mathrm{d}\mathbf{f} \quad \left(\text{where } q(\mathbf{f}|\mathbf{y}) = \textstyle\int p(\mathbf{f}|\mathbf{z})q(\mathbf{z}|\mathbf{y})\mathrm{d}\mathbf{z}\right).$$

This is for any $r(\mathbf{z})$. We choose $r(\mathbf{z}) \stackrel{\text{def}}{=} q(\mathbf{z}|\mathbf{f},\mathbf{y}) = q(\mathbf{f},\mathbf{z}|\mathbf{y})/q(\mathbf{f}|\mathbf{y})$ and cancel out $q(\mathbf{f}|\mathbf{y})$ to obtain

$$\log Z_B^* \geq \max_{q(\mathbf{z}|\mathbf{y})} \int q(\mathbf{f},\mathbf{z}|\mathbf{y}) \log \frac{p(\mathbf{y},\mathbf{f},\mathbf{z})}{q(\mathbf{f},\mathbf{z}|\mathbf{y})}\,\mathrm{d}\mathbf{z}\,\mathrm{d}\mathbf{f}.$$

The objective on the right is $\log \tilde{Z}_B$ by definition (47).

B.2 Derivation of $r(\mathbf{f})$ and $r(\mathbf{y})$ for Lemma 2

Let $r(\mathbf{f})$ be a prior distribution such that the posterior

$$r(\mathbf{f}|\mathbf{y}) = p(\mathbf{y}|\mathbf{f})\, r(\mathbf{f})/r(\mathbf{y}), \qquad \text{where } r(\mathbf{y}) \stackrel{\text{def}}{=} \int p(\mathbf{y}|\mathbf{f})\, r(\mathbf{f}) \mathrm{d}\mathbf{f},$$

is a C-variate Gaussian density on $\mathbf{f}$ with mean $\mathbf{a}$ and precision W. Rearranging gives

$$\frac{r(\mathbf{f})}{r(\mathbf{y})} = \frac{r(\mathbf{f}|\mathbf{y})}{p(\mathbf{y}|\mathbf{f})} = \sum_{c=1}^{C} \frac{|W|^{1/2}}{(2\pi)^{C/2}} \exp -\frac{1}{2}\Big[(\mathbf{f}-\mathbf{a})^{\mathrm{T}} W(\mathbf{f}-\mathbf{a}) - 2(\mathbf{e}^c - \mathbf{y})^{\mathrm{T}}\mathbf{f})\Big]. \tag{50}$$

Let $\mathbf{a}^c$ be such that

$$W\mathbf{a}^c = W\mathbf{a} + \mathbf{e}^c - \mathbf{y}, \tag{51}$$

and define

$$r^c(\mathbf{f}) \stackrel{\text{def}}{=} \frac{|W|^{1/2}}{(2\pi)^{C/2}} \exp\left[-\frac{1}{2}(\mathbf{f}-\mathbf{a}^c)^{\mathrm{T}} W(\mathbf{f}-\mathbf{a}^c)\right].$$

By completing the square the terms within the brackets of (50), we obtain

$$r(\mathbf{f}) = r(\mathbf{y}) \exp\left[-\frac{1}{2}\mathbf{a}^{\mathrm{T}} W\mathbf{a}\right] \sum_{c=1}^{C} \exp\left[\frac{1}{2}(\mathbf{a}^c)^{\mathrm{T}} W\mathbf{a}^c\right] r^c(\mathbf{f}).$$

This is a mixture of Gaussians model, so let $r(\mathbf{f}) = \sum_c \gamma^c r^c(\mathbf{f})$. Normalization gives

$$r(\mathbf{y}) = \frac{\exp\left[\frac{1}{2}\mathbf{a}^{\mathrm{T}} W\mathbf{a}\right]}{\sum_{c=1}^{C} \exp\left[\frac{1}{2}(\mathbf{a}^c)^{\mathrm{T}} W\mathbf{a}^c\right]}, \qquad \gamma^c \stackrel{\text{def}}{=} \frac{\exp\left[\frac{1}{2}(\mathbf{a}^c)^{\mathrm{T}} W\mathbf{a}^c\right]}{\sum_{c'} \exp\left[\frac{1}{2}(\mathbf{a}^{c'})^{\mathrm{T}} W\mathbf{a}^{c'}\right]}. \tag{52}$$

B.3 Derivation of Lower Bound h on the Expected Log-likelihood for Lemma 5

Recall from (12) that

$$h(\mathbf{y}; q, r) \stackrel{\text{def}}{=} \int q(\mathbf{f}|\mathbf{y}) \log r(\mathbf{f}|\mathbf{y}) \mathrm{d}\mathbf{f} + \log r(\mathbf{y}) - \log \sum_{c=1}^{C} \gamma^c \int q(\mathbf{f}|\mathbf{y}) r^c(\mathbf{f}) \mathrm{d}\mathbf{f}. \tag{53}$$

We simplify the first two terms on the right:

$$\int q(\mathbf{f}|\mathbf{y}) \log r(\mathbf{f}|\mathbf{y}) \mathrm{d}\mathbf{f} = \frac{1}{2}\Big(-C\log 2\pi + \log|W| - (\mathbf{m}-\mathbf{a})^{\mathrm{T}} W(\mathbf{m}-\mathbf{a})^{\mathrm{T}} - \mathrm{tr}\, WV\Big), \tag{54}$$

$$\log r(\mathbf{y}) = \frac{1}{2}\mathbf{a}^{\mathrm{T}} W\mathbf{a} - \log \sum_{c=1}^{C} \exp\left[\frac{1}{2}(\mathbf{a}^c)^{\mathrm{T}} W\mathbf{a}^c\right]. \tag{55}$$

For the third term on the right of (53), we introduce $S \stackrel{\text{def}}{=} V^{-1} + W$ and use

$$\begin{aligned} &\int q(\mathbf{f}|\mathbf{y}) r^c(\mathbf{f}) \mathrm{d}\mathbf{f} \\ &= \sqrt{\frac{|W|}{(2\pi)^C |V|\,|S|}} \exp -\frac{1}{2}\left[\mathbf{m}^{\mathrm{T}} V^{-1}\mathbf{m} + (\mathbf{a}^c)^{\mathrm{T}} W\mathbf{a}^c - \left(V^{-1}\mathbf{m} + W\mathbf{a}^c\right)^{\mathrm{T}} S^{-1}\left(V^{-1}\mathbf{m} + W\mathbf{a}^c\right)\right] \end{aligned} \tag{56}$$

to obtain

$$\log \sum_{c=1}^{C} \gamma^c \int q(\mathbf{f}|\mathbf{y}) r^c(\mathbf{f}) \mathrm{d}\mathbf{f} = -\log \sum_{c=1}^{C} \exp\left[\frac{1}{2}(\mathbf{a}^c)^{\mathrm{T}} W \mathbf{a}^c\right] + \frac{1}{2}\log|W| - \frac{C}{2}\log 2\pi - \frac{1}{2}\log|SV| + \log \sum_{c=1}^{C} \exp -\frac{1}{2}\left\{\mathbf{m}^{\mathrm{T}} V^{-1}\mathbf{m} - \left(V^{-1}\mathbf{m} + W\mathbf{a}^c\right)^{\mathrm{T}} S^{-1}\left(V^{-1}\mathbf{m} + W\mathbf{a}^c\right)\right\}, \quad (57)$$

where the first term is from the denominator of γ^c (Equation 52) and the second to fourth terms are from the first factor in Equation 56. At present, let us focus on the term within the braces in the above equation. Let $\mathbf{b} \stackrel{\text{def}}{=} W(\mathbf{m}-\mathbf{a}) + \mathbf{y}$ and use the identity $W\mathbf{a}^c = W\mathbf{a} + \mathbf{e}^c - \mathbf{y}$ (Equation 51) and definition $S \stackrel{\text{def}}{=} V^{-1} + W$. Then

$$\begin{aligned}
&\mathbf{m}^{\mathrm{T}} V^{-1}\mathbf{m} - \left(V^{-1}\mathbf{m} + W\mathbf{a}^c\right)^{\mathrm{T}} S^{-1}\left(V^{-1}\mathbf{m} + W\mathbf{a}^c\right) \\
&= \mathbf{m}^{\mathrm{T}} V^{-1}\mathbf{m} - \left(V^{-1}\mathbf{m} + W\mathbf{a} + \mathbf{e}^c - \mathbf{y}\right)^{\mathrm{T}} S^{-1}\left(V^{-1}\mathbf{m} + W\mathbf{a} + \mathbf{e}^c - \mathbf{y}\right) \\
&= \mathbf{m}^{\mathrm{T}} V^{-1}\mathbf{m} - \left(V^{-1}\mathbf{m} + W\mathbf{m} + \mathbf{e}^c - \mathbf{b}\right)^{\mathrm{T}} S^{-1}\left(V^{-1}\mathbf{m} + W\mathbf{m} + \mathbf{e}^c - \mathbf{b}\right) \\
&= \mathbf{m}^{\mathrm{T}} V^{-1}\mathbf{m} - (S\mathbf{m} + \mathbf{e}^c - \mathbf{b})^{\mathrm{T}} S^{-1}(S\mathbf{m} + \mathbf{e}^c - \mathbf{b}) \\
&= \mathbf{m}^{\mathrm{T}} V^{-1}\mathbf{m} - \mathbf{m}^{\mathrm{T}} S\mathbf{m} - 2\mathbf{m}^{\mathrm{T}}(\mathbf{e}^c - \mathbf{b}) - (\mathbf{e}^c - \mathbf{b})^{\mathrm{T}} S^{-1}(\mathbf{e}^c - \mathbf{b}) \\
&= -\mathbf{m}^{\mathrm{T}} W\mathbf{m} + 2\mathbf{m}^{\mathrm{T}} W(\mathbf{m}-\mathbf{a}) + 2\mathbf{m}^{\mathrm{T}}\mathbf{y} - 2\mathbf{m}^{\mathrm{T}}\mathbf{e}^c - (\mathbf{b}-\mathbf{e}^c)^{\mathrm{T}} S^{-1}(\mathbf{b}-\mathbf{e}^c) \\
&= \mathbf{m}^{\mathrm{T}} W\mathbf{m} - 2\mathbf{m}^{\mathrm{T}} W\mathbf{a} + 2\mathbf{m}^{\mathrm{T}}\mathbf{y} - 2\mathbf{m}^{\mathrm{T}}\mathbf{e}^c - (\mathbf{b}-\mathbf{e}^c)^{\mathrm{T}} S^{-1}(\mathbf{b}-\mathbf{e}^c) \\
&= (\mathbf{m}-\mathbf{a})^{\mathrm{T}} W(\mathbf{m}-\mathbf{a}) - \mathbf{a}^{\mathrm{T}} W\mathbf{a} + 2\mathbf{m}^{\mathrm{T}}\mathbf{y} - 2\mathbf{m}^{\mathrm{T}}\mathbf{e}^c - (\mathbf{b}-\mathbf{e}^c)^{\mathrm{T}} S^{-1}(\mathbf{b}-\mathbf{e}^c) \\
&= (\mathbf{m}-\mathbf{a})^{\mathrm{T}} W(\mathbf{m}-\mathbf{a}) - \mathbf{a}^{\mathrm{T}} W\mathbf{a} + 2\mathbf{m}^{\mathrm{T}}\mathbf{y} - 2\log g^c(\mathbf{y}; q, r),
\end{aligned}$$

where $g^c(\mathbf{y}; q, r) \stackrel{\text{def}}{=} \exp\left[\mathbf{m}^{\mathrm{T}}\mathbf{e}^c + \frac{1}{2}(\mathbf{b}-\mathbf{e}^c)^{\mathrm{T}} S^{-1}(\mathbf{b}-\mathbf{e}^c)\right]$. By pulling out the terms independent of the dummy variable c in the last term of (57), we can rewrite

$$\log \sum_{c=1}^{C} \gamma^c \int q(\mathbf{f}|\mathbf{y}) r^c(\mathbf{f}) \mathrm{d}\mathbf{f} = -\log \sum_{c=1}^{C} \exp\left[\frac{1}{2}(\mathbf{a}^c)^{\mathrm{T}} W \mathbf{a}^c\right] + \frac{1}{2}\log|W| - \frac{C}{2}\log 2\pi - \frac{1}{2}\log|SV| - \frac{1}{2}(\mathbf{m}-\mathbf{a})^{\mathrm{T}} W(\mathbf{m}-\mathbf{a}) + \frac{1}{2}\mathbf{a}^{\mathrm{T}} W\mathbf{a} - \mathbf{m}^{\mathrm{T}}\mathbf{y} + \log \sum_{c=1}^{C} g^c(\mathbf{y}; q, r). \quad (58)$$

Finally, putting (54), (55) and (58) into (53) and cancelling terms yields

$$h(\mathbf{y}; q, r) = \frac{C}{2} + \frac{1}{2}\log|SV| - \frac{1}{2}\mathrm{tr}\, SV + \mathbf{m}^{\mathrm{T}}\mathbf{y} - \log \sum_{i=1}^{C} g^c(\mathbf{y}; q, r).$$

Since distribution $r(\mathbf{f}|\mathbf{y})$ is completely determined by its mean $\mathbf{a}$ and precision W, we may use these parameters instead of r in our notation; that is, $h(\mathbf{y}; q, \mathbf{a}, W)$ instead of $h(\mathbf{y}; q, r)$.

B.4 Lemmas to Prove Theorem 6

This section collects the necessary lemmas to prove Theorem 6. Function $g^c(q, \mathbf{b}, S)$ and function $h(\mathbf{y}; q, \mathbf{b}, S)$ are given by Equations 16 and 19 in the main text, while variables $\bar{\mathbf{g}}$ and A are defined by Equations 17 and 18.

Lemma 25 *Function h is jointly concave in $\mathbf{b}$ and S.*

Proof The following facts are used: (a) the log-determinant term is concave in S (Horn and Johnson, 1985, Theorem 7.6.7); (b) the matrix trace term is both concave and convex in S; (c) the quadratic term in the exponent of g^c is jointly convex in S and $\mathbf{b}$ (Ando, 1979); and (d) the sum of log-convex functions is log-convex. ■

Lemma 26 *The maximum of h given S with respect to $\mathbf{b}$ is at $\mathbf{b}=\mathbf{b}^*$ that satisfies $\mathbf{b}^*=\bar{\mathbf{g}}^*$, where $\bar{\mathbf{g}}^*$ is obtained by evaluating $\bar{\mathbf{g}}$ at $\mathbf{b}^*$.*

Proof Proved by setting the gradient $\partial h/\partial \mathbf{b}$ to zero. ■

Lemma 27 *The maximum of h given $\mathbf{b}$ with respect to S is at $S=S^*$ that satisfies the implicit equation $-S^*VS^*+S^*+A^*=0$, where $A^*\neq 0$ is A evaluated at S^*.*

Proof Proved by equating the gradient

$$\frac{\partial h}{\partial S}=-\frac{1}{2}V+\frac{1}{2}S^{-1}+\frac{1}{2}S^{-1}AS^{-1} \tag{59}$$

to zero and pre- and post-multiplying both sides by S (valid since $S\succ 0$ by definition). ■

Lemma 28 *Let $A\neq 0$, $A\succeq 0$ and $V\succ 0$. Let S^{fx} be the fixed point given implicitly by*

$$-S^{\text{fx}}VS^{\text{fx}}+S^{\text{fx}}+A=0. \tag{60}$$

Then $V^{-1}\preceq S^{\text{fx}}\preceq V^{-1}+A$, and $S\neq V^{-1}$ and $S^{\text{fx}}\neq V^{-1}+A$; that is, there exists a matrix W satisfying $0\preceq W^{\text{fx}}\preceq A$ and $W\notin\{0,A\}$ such that $S^{\text{fx}}=V^{-1}+W^{\text{fx}}$. Furthermore, $\text{null}(W^{\text{fx}})=\text{null}(A)$.

Proof Let V factorizes to LL^{T}, where L is non-singular; for example, matrix L can be the lower Cholesky factor of V. We pre- and post-multiply Equation 60 by L^{T} and L to obtain the equation $-(L^{\text{T}}S^{\text{fx}}L)(L^{\text{T}}S^{\text{fx}}L)+(L^{\text{T}}S^{\text{fx}}L)+L^{\text{T}}AL=0$. This is a matrix quadratic equation in $L^{\text{T}}S^{\text{fx}}L$, so we use Theorem 24 to reach the solution

$$L^{\text{T}}S^{\text{fx}}L=P\tilde{\Lambda}P^{\text{T}},\qquad\qquad \tilde{\Lambda}\stackrel{\text{def}}{=}(\Lambda+I/4)^{1/2}+I/2, \tag{61}$$

where $P\Lambda P^{\text{T}}$ is the eigen-decomposition of $L^{\text{T}}AL$. Matrix A is positive semi-definite, so similarly is $L^{\text{T}}AL$ (Horn and Johnson, 1985, Observation 7.7.2) and $L^{\text{T}}AL+I/4$. Therefore, $L^{\text{T}}S^{\text{fx}}L$ is positive definite; see Theorem 24. Since L is non-singular, we can write $S^{\text{fx}}=L^{-\text{T}}\left(L^{\text{T}}S^{\text{fx}}L\right)L^{-1}$, so S^{fx} is positive definite (Horn and Johnson, 1985, Observation 7.7.2). Define $W^{\text{fx}}\stackrel{\text{def}}{=}S^{\text{fx}}-V^{-1}$, then

$$W^{\text{fx}}=L^{-\text{T}}\left(L^{\text{T}}S^{\text{fx}}L\right)L^{-1}-(LL^{\text{T}})^{-1}=L^{-\text{T}}\left(L^{\text{T}}S^{\text{fx}}L-I\right)L^{-1}=L^{-\text{T}}P\left(\tilde{\Lambda}-I\right)P^{\text{T}}L^{-1},$$

where (61) is used. Since the least diagonal value in $\tilde{\Lambda}$ is one, W^{fx} is positive semi-definite, so $S^{\text{fx}} \succeq V^{-1}$. Moreover, since $V \neq 0$ and $A \neq 0$, so $L^{\mathrm{T}}AL \neq 0$, $\Lambda \neq 0$, $\tilde{\Lambda} \neq I$ and $W^{\text{fx}} \neq 0$. Hence $S^{\text{fx}} \neq V^{-1}$. Substitute $S^{\text{fx}} = V^{-1} + W^{\text{fx}}$ into (60) and rearranging gives

$$W^{\text{fx}} = A - W^{\text{fx}} V W^{\text{fx}} \preceq A. \tag{62}$$

Thus $S^{\text{fx}} \preceq V^{-1} + A$. Moreover, $W^{\text{fx}} \neq 0$ and $V \neq 0$ shows that $S^{\text{fx}} \neq V^{-1} + A$.

We now prove $\text{null}(W^{\text{fx}}) = \text{null}(A)$. Already, $W^{\text{fx}} \preceq A$ gives $\text{null}(A) \subseteq \text{null}(W)$ with Proposition 19, so it remains to proof $\text{null}(W^{\text{fx}}) \subseteq \text{null}(A)$. Let $\mathbf{x} \in \text{null}(W^{\text{fx}})$. Post-multiply both sides of the equality in (62) by $\mathbf{x}$ and use $W^{\text{fx}}\mathbf{x} = \mathbf{0}$ to give $A\mathbf{x} = \mathbf{0}$. Thus $\mathbf{x} \in \text{null}(A)$. ■

B.5 Proof of Lemma 7

We introduce $u(\eta)$, where $u(0) = \ell(\mathbf{y}; q)$ and $u(1) = \log p(\mathbf{y}|\mathbf{m})$, and obtain its first two derivatives:

$$\begin{aligned}
u(\eta) &\stackrel{\text{def}}{=} \int q(\mathbf{f}|\mathbf{y}) \left([(1-\eta)\mathbf{f} + \eta\mathbf{m}]^{\mathrm{T}} \mathbf{y} - \log \sum_{c=1}^{C} \exp [(1-\eta)\mathbf{f} + \eta\mathbf{m}]^{\mathrm{T}} \mathbf{e}^c \right) \mathrm{d}\mathbf{f}, \\
\frac{\mathrm{d}u}{\mathrm{d}\eta} &= \int q(\mathbf{f}|\mathbf{y}) \left([\mathbf{m} - \mathbf{f}]^{\mathrm{T}} \mathbf{y} - [\mathbf{m} - \mathbf{f}]^{\mathrm{T}} \boldsymbol{\pi}_\eta \right) \mathrm{d}\mathbf{f} = - \int q(\mathbf{f}|\mathbf{y}) [\mathbf{m} - \mathbf{f}]^{\mathrm{T}} \boldsymbol{\pi}_\eta \, \mathrm{d}\mathbf{f}, \\
\frac{\mathrm{d}^2 u}{\mathrm{d}\eta^2} &= - \int q(\mathbf{f}|\mathbf{y}) [\mathbf{m} - \mathbf{f}]^{\mathrm{T}} \left(\Pi_\eta - \boldsymbol{\pi}_\eta \boldsymbol{\pi}_\eta^{\mathrm{T}} \right) [\mathbf{m} - \mathbf{f}] \, \mathrm{d}\mathbf{f},
\end{aligned}$$

where

$$\pi_\eta^c \stackrel{\text{def}}{=} \frac{\exp [(1-\eta)\mathbf{f} + \eta\mathbf{m}]^{\mathrm{T}} \mathbf{e}^c}{\sum_{c'=1}^{C} \exp [(1-\eta)\mathbf{f} + \eta\mathbf{m}]^{\mathrm{T}} \mathbf{e}^{c'}}, \qquad \boldsymbol{\pi}_\eta \stackrel{\text{def}}{=} \left(\pi_\eta^1, \ldots, \pi_\eta^C \right)^{\mathrm{T}},$$

and Π_η is a diagonal matrix with $\boldsymbol{\pi}_\eta$ along its diagonal. The first derivative $\mathrm{d}u/\mathrm{d}\eta$ at $\eta = 1$ is zero because $\boldsymbol{\pi}_1$ is independent of $\mathbf{f}$ and the mean of $\mathbf{f}$ under $q(\mathbf{f}|\mathbf{y})$ is $\mathbf{m}$. Moreover, the second derivative $\mathrm{d}^2u/\mathrm{d}\eta^2$ is non-positive because the matrix within the parentheses is positive semi-definite. Hence u is concave in η, and a maximal is $u(1) = \log p(\mathbf{y}|\mathbf{m})$ where the gradient is zero. ■

B.6 A Data-independent Lower Bound on the Marginal Likelihood

We first introduce a bound on $h(\mathbf{y}; q, \mathbf{b}, S)$ when the variational posterior is chosen to be an isotropic Gaussian.

Lemma 29 *Let $q \equiv \mathcal{N}(\mathbf{m}, \sigma_{\text{v}}^2 I)$, and let $h(\mathbf{y}; q, \mathbf{b}, S)$ be as defined by Equation 19. Then*

$$\max_{\mathbf{b}, S} h(\mathbf{y}; q, \mathbf{b}, S) \geq -\frac{C-1}{2} \left[2\sqrt{\frac{\sigma_{\text{v}}^2}{C} + \frac{1}{4}} - \log \left(\sqrt{\frac{\sigma_{\text{v}}^2}{C} + \frac{1}{4}} + \frac{1}{2} \right) - 1 \right] - \frac{1}{2} \log C + \frac{1}{2} \log \frac{\exp 2\mathbf{m}^{\mathrm{T}} \mathbf{y}}{\sum_{c=1}^{C} \exp 2\mathbf{m}^{\mathrm{T}} \mathbf{e}^c}.$$

with equality when $\mathbf{m} = \mathbf{0}$. *This is a decreasing function of* C *and* σ_{v}^2. *Moreover*

$$\max_{\mathbf{b},S} h(\mathbf{y};q,\mathbf{b},S) > -\frac{1}{2}\sigma_{\mathrm{v}}^2 - \frac{1}{2}\log C + \frac{1}{2}\log\frac{\exp 2\mathbf{m}^{\mathrm{T}}\mathbf{y}}{\sum_{c=1}^{C}\exp 2\mathbf{m}^{\mathrm{T}}\mathbf{e}^c}.$$

Proof Let $\tilde{g}^c(\mathbf{b},S) \stackrel{\text{def}}{=} \exp(\mathbf{b}-\mathbf{e}^c)^{\mathrm{T}}S^{-1}(\mathbf{b}-\mathbf{e}^c)$. Using Cauchy-Schwarz inequality on $\sum_{c=1}^{C} g^c$ gives

$$\log\sum_{c=1}^{C} g^c \le \frac{1}{2}\log\sum_{c=1}^{C}\exp 2\mathbf{m}^{\mathrm{T}}\mathbf{e}^c + \frac{1}{2}\log\sum_{c=1}^{C}\tilde{g}^c.$$

We use this inequality together with the choice of distribution q, which has variance σ_{v}^2 in all directions, to obtain $h(\mathbf{y};q,\mathbf{b},S) \ge \tilde{h}(\mathbf{y};q,\mathbf{b},S)$, where

$$\tilde{h}(\mathbf{y};q,\mathbf{b},S) \stackrel{\text{def}}{=} \frac{C}{2} + \frac{C}{2}\log\sigma_{\mathrm{v}}^2 + \frac{1}{2}\log|S| - \frac{\sigma_{\mathrm{v}}^2}{2}\operatorname{tr}S - \frac{1}{2}\log\sum_{c=1}^{C}\tilde{g}^c(\mathbf{b},S) + \frac{1}{2}\log\frac{\exp 2\mathbf{m}^{\mathrm{T}}\mathbf{y}}{\sum_{c=1}^{C}\exp 2\mathbf{m}^{\mathrm{T}}\mathbf{e}^c}.$$

Let $\bar{\tilde{g}}^c \stackrel{\text{def}}{=} \tilde{g}^c / \sum_{c'=1}^{C}\tilde{g}^{c'}$ and $\bar{\tilde{\mathbf{g}}} \stackrel{\text{def}}{=} (\bar{\tilde{g}}^1,\ldots,\bar{\tilde{g}}^C)^{\mathrm{T}}$, where the arguments $(\mathbf{b},S)$ are suppressed in the notation. Let $\bar{\tilde{G}}$ be the diagonal matrix with $\bar{\tilde{\mathbf{g}}}$ along its diagonal. Also, define $\tilde{A} \stackrel{\text{def}}{=} \mathbf{b}\mathbf{b}^{\mathrm{T}} - \bar{\tilde{\mathbf{g}}}\mathbf{b}^{\mathrm{T}} - \mathbf{b}\bar{\tilde{\mathbf{g}}}^{\mathrm{T}} + \bar{\tilde{G}}$. These two definitions are analogous to the definitions of $\bar{\mathbf{g}}$ and A in Equations 17 and 18.

Let the maximum of $\tilde{h}$ be at $(\mathbf{b}^*,S^*)$. It is straightforward to modify Lemmas 25, 26 and 27 for $\tilde{h}$. The modified Lemma 25 says that $(\mathbf{b}^*,S^*)$ is unique. The other two modified lemmas will give the self-consistent equations

$$\mathbf{b}^* = \bar{\tilde{\mathbf{g}}}(\mathbf{b}^*,S^*), \qquad\qquad -\sigma_{\mathrm{v}}^2(S^*)^2 + S^* + A^* = 0.$$

By symmetry, $\mathbf{b}^* = \mathbf{1}/C$ and $A^* = I/C - \mathbf{1}\mathbf{1}^{\mathrm{T}}/C^2$. An eigenpair of A^* is $(0,\mathbf{1}/\sqrt{C})$; the other $(C-1)$ eigenpairs are $(1/C,\mathbf{u}_d)$, $d = 1\ldots(C-1)$, where $\mathbf{1}^{\mathrm{T}}\mathbf{u}_d = 0$ in addition to the orthonormal conditions. Let

$$\lambda \stackrel{\text{def}}{=} \sqrt{\sigma_{\mathrm{v}}^2/C + 1/4} + 1/2. \tag{63}$$

Since $\sigma_{\mathrm{v}}^2 S^* = (\sigma_{\mathrm{v}}^2 A^* + I/4)^{1/2} + I/2$ (see the proof for Lemma 28 in Appendix B.4), the eigenvalues of S^* are σ_{v}^{-2} (with algebraic multiplicity one) and $\sigma_{\mathrm{v}}^{-2}\lambda$ (with algebraic multiplicity $C-1$), and the eigenvectors of S^* are those of A^*. Thus the determinant and trace of S^* can be readily obtained. With $\mathbf{b}^* = \mathbf{1}/C$, observe that

$$(\mathbf{b}^* - \mathbf{e}^c)^{\mathrm{T}}\mathbf{1}/\sqrt{C} = 0, \qquad\qquad (\mathbf{b}^* - \mathbf{e}^c)^{\mathrm{T}}\mathbf{u}_d = -u_{dc},$$

where u_{dc} is the cth entry in the eigenvector $\mathbf{u}_d$. For the exponent of $\tilde{g}^c$, using $(S^*)^{-1}$ in its eigendecomposition and the two observations above gives $(\mathbf{b}^* - \mathbf{e}^c)^{\mathrm{T}}(S^*)^{-1}(\mathbf{b}^* - \mathbf{e}^c) = (\sigma_{\mathrm{v}}^2/\lambda)\sum_{d=1}^{C-1} u_{dc}^2$. But the eigenvectors of S^* are orthonormal, so $\left((1/\sqrt{C})^2 + \sum_{d=1}^{C-1} u_{dc}^2\right)$ is unity. Hence, we have $(\mathbf{b}^* - \mathbf{e}^c)^{\mathrm{T}}(S^*)^{-1}(\mathbf{b}^* - \mathbf{e}^c) = (\sigma_{\mathrm{v}}^2/\lambda)(1 - 1/C)$. This is independent of c. Therefore

$$\begin{aligned}
&\max_{\mathbf{b},S}\tilde{h}(\mathbf{y};q,\mathbf{b},S) \\
&= \frac{C}{2} + \frac{C}{2}\log\sigma_{\mathrm{v}}^2 + \frac{1}{2}\log\sigma_{\mathrm{v}}^{-2}(\sigma_{\mathrm{v}}^{-2}\lambda)^{C-1} - \frac{\sigma_{\mathrm{v}}^2}{2}\left(\sigma_{\mathrm{v}}^{-2} + (C-1)\sigma_{\mathrm{v}}^{-2}\lambda\right) \\
&\qquad\qquad - \frac{1}{2}\log C\exp\left[\frac{\sigma_{\mathrm{v}}^2}{\lambda}\left(1 - \frac{1}{C}\right)\right] + \frac{1}{2}\log\frac{\exp 2\mathbf{m}^{\mathrm{T}}\mathbf{y}}{\sum_{c=1}^{C}\exp 2\mathbf{m}^{\mathrm{T}}\mathbf{e}^c} \\
&= \frac{C}{2} + \frac{C-1}{2}\log\lambda - \frac{1}{2}(1 + (C-1)\lambda) - \underline{\frac{1}{2}\frac{\sigma_{\mathrm{v}}^2}{\lambda}\left(1 - \frac{1}{C}\right)} - \frac{1}{2}\log C + \frac{1}{2}\log\frac{\exp 2\mathbf{m}^{\mathrm{T}}\mathbf{y}}{\sum_{c=1}^{C}\exp 2\mathbf{m}^{\mathrm{T}}\mathbf{e}^c}.
\end{aligned}$$

The underlined term can be simplified to $(C-1)(1-\lambda)/2$ by expressing σ_{v}^2 in λ using (63). Further simplification and substitution with the definition of λ gives

$$\max_{\mathbf{b},S} \tilde{h}(\mathbf{y};q,\mathbf{b},S) = -\frac{C-1}{2}\left[2\sqrt{\frac{\sigma_{\mathrm{v}}^2}{C}+\frac{1}{4}} - \log\left(\sqrt{\frac{\sigma_{\mathrm{v}}^2}{C}+\frac{1}{4}}+\frac{1}{2}\right)-1\right] - \frac{1}{2}\log C + \frac{1}{2}\log\frac{\exp 2\mathbf{m}^{\mathrm{T}}\mathbf{y}}{\sum_{c=1}^{C}\exp 2\mathbf{m}^{\mathrm{T}}\mathbf{e}^c}.$$

Combining this with $h(\mathbf{y};q,\mathbf{b},S) \geq \tilde{h}(\mathbf{y};q,\mathbf{b},S)$ gives the the first inequality in the lemma statement. When $\mathbf{m}=\mathbf{0}$, we can obtain a modification of the proof using $\sum_{c=1}^{C} g^c$ directly without bounding through the Cauchy-Schwarz inequality. This modified proof shows that

$$\max_{\mathbf{b},S} h(\mathbf{y};q,\mathbf{b},S) = -\frac{C-1}{2}\left[2\sqrt{\frac{\sigma_{\mathrm{v}}^2}{C}+\frac{1}{4}} - \log\left(\sqrt{\frac{\sigma_{\mathrm{v}}^2}{C}+\frac{1}{4}}+\frac{1}{2}\right)-1\right] - \log C. \qquad (\mathbf{m}=\mathbf{0})$$

The first term is a decreasing function of C, and we now show that this first term is bounded by $-\sigma_{\mathrm{v}}^2/2$ from below. Let $f(x) \stackrel{\text{def}}{=} x^2 - 3x + 2 + \log x$. Then $f=0$ and $\mathrm{d}f/\mathrm{d}x = 0$ at $x=1$, and $\mathrm{d}^2 f/\mathrm{d}x^2 > 0$ in the domain $x \geq 1$. Therefore, $f(x) \geq 0$ for all $x \geq 1$. Then, for function $f(x)$ we set $x \stackrel{\text{def}}{=} \sqrt{\sigma_{\mathrm{v}}^2/C + 1/4} + 1/2$ and use $C-1<C$ to complete the proof after rearrangement. ■

Proof (of Theorem 9)

$$\begin{aligned}\log p(\mathbf{y}) \geq \max_{q,\{\mathbf{b}_i\},\{S_i\}} \log Z_h &\geq \max_{\{\mathbf{b}_i\},\{S_i\}} \log Z_h|_{q(\mathbf{f}|\mathbf{y})=\mathcal{N}(\mathbf{0},\sigma_{\mathrm{v}}^2 I)} \\ &= \frac{nC}{2} + \frac{nC}{2}\log\sigma_{\mathrm{v}}^2 - \frac{1}{2}\log|K| - \frac{\sigma_{\mathrm{v}}^2}{2}\operatorname{tr}K^{-1} + \sum_{i=1}^{n}\max_{\mathbf{b}_i,S_i} h(\mathbf{y}_i,\mathcal{N}(\mathbf{0},\sigma_{\mathrm{v}}^2 I),\mathbf{b}_i,S_i) \\ &= \frac{nC}{2} + \frac{nC}{2}\log\sigma_{\mathrm{v}}^2 - \frac{1}{2}\log|K| - \frac{\sigma_{\mathrm{v}}^2}{2}\operatorname{tr}K^{-1} + n\max_{\mathbf{b},S} h(\mathbf{y},\mathcal{N}(\mathbf{0},\sigma_{\mathrm{v}}^2 I),\mathbf{b},S).\end{aligned}$$

Lemma 29 is then applied on $\max h$. ■

Proof (of Theorem 10)

$$\begin{aligned}\log p(\mathbf{y}) &\geq \max_{q,\{\mathbf{b}_i\},\{S_i\}} \log Z_h \\ &\geq \max_{\{\mathbf{b}_i\},\{S_i\}} \log Z_h|_{q(\mathbf{f})=p(\mathbf{f})} \\ &= \max_{\{\mathbf{b}_i\},\{S_i\}}\left[\frac{nC}{2} + \frac{1}{2}\sum_{i=1}^{n}\left(\log|S_iK_i| - \operatorname{tr}S_iK_i\right) - \sum_{i=1}^{n}\log\sum_{c=1}^{C}\exp\left[\frac{1}{2}(\mathbf{b}_i-\mathbf{e}^c)^{\mathrm{T}}S_i^{-1}(\mathbf{b}_i-\mathbf{e}^c)\right]\right].\end{aligned}$$

The same expression can be obtained by setting $V=K$ and $\mathbf{m}=\eta\mathbf{1}$ and maximizing the resultant expression with respect to η. For $K_1 = K_2, \ldots K_n = K^{\mathrm{c}}$, we have

$$\frac{1}{n}\log p(\mathbf{y}) \geq \max_{\mathbf{b},S}\left[\frac{C}{2} + \frac{1}{2}\log|SK^{\mathrm{c}}| - \frac{1}{2}\operatorname{tr}SK^{\mathrm{c}} - \log\sum_{c=1}^{C}\exp\left[\frac{1}{2}(\mathbf{b}-\mathbf{e}^c)^{\mathrm{T}}S^{-1}(\mathbf{b}-\mathbf{e}^c)\right]\right].$$

For the choice of $K^{\mathrm{c}} \stackrel{\text{def}}{=} \sigma^2 I$, we apply Lemma 29. ■

B.7 Lower Bound on Predictive Probability: Proof of Theorem 12

For a set of n_* test inputs $X_* \stackrel{\text{def}}{=} \{\mathbf{x}_{*1}, \ldots, \mathbf{x}_{*n_*}\}$, the log joint predictive probability for $\mathbf{x}_{*j}$ to be in class c_j ($j = 1 \ldots n_*$) is

$$\begin{aligned}
\log p(\{y_{*j}^{c_j} = 1\}_{j=1}^{n_*}|\mathbf{y}) &= \log \int p(\{y_{*j}^{c_j} = 1\}_{j=1}^{n_*}|\mathbf{f}_*)\, p(\mathbf{f}_*|\mathbf{y})\, \mathrm{d}\mathbf{f}_* \\
&= \log \int p(\{y_{*j}^{c_j} = 1\}_{j=1}^{n_*}|\mathbf{f}_*)\, p(\mathbf{f}_*, \mathbf{f}|\mathbf{y}) \mathrm{d}\mathbf{f}_*\, \mathrm{d}\mathbf{f} \\
&= \log \int p(\{y_{*j}^{c_j} = 1\}_{j=1}^{n_*}|\mathbf{f}_*)\, q(\mathbf{f}_*, \mathbf{f}|\mathbf{y}) \frac{p(\mathbf{f}_*, \mathbf{f}|\mathbf{y})}{q(\mathbf{f}_*, \mathbf{f}|\mathbf{y})}\, \mathrm{d}\mathbf{f}_* \mathrm{d}\mathbf{f}.
\end{aligned}$$

Applying Jensen's inequality gives the inequality

$$\begin{aligned}
\log p(\{y_{*j}^{c_j} = 1\}_{j=1}^{n_*}|\mathbf{y}) &\geq \int q(\mathbf{f}_*, \mathbf{f}|\mathbf{y}) \log p(\{y_{*j}^{c_j} = 1\}_{j=1}^{n_*}|\mathbf{f}_*) \frac{p(\mathbf{f}_*, \mathbf{f}|\mathbf{y})}{q(\mathbf{f}_*, \mathbf{f}|\mathbf{y})}\, \mathrm{d}\mathbf{f}_* \mathrm{d}\mathbf{f} \\
&= \int q(\mathbf{f}_*|\mathbf{y}) \log p(\{y_{*j}^{c_j} = 1\}_{j=1}^{n_*}|\mathbf{f}_*)\, \mathrm{d}\mathbf{f}_* - \mathrm{KL}(q(\mathbf{f}_*, \mathbf{f}|\mathbf{y}) \| p(\mathbf{f}_*, \mathbf{f}|\mathbf{y})).
\end{aligned}$$

Within the first term, the conditional joint predictive probability factorizes across the $\mathbf{x}_{*j}$s. The second term is the Kullback-Leibler divergence from $q(\mathbf{f}_*, \mathbf{f}|\mathbf{y})$ to $p(\mathbf{f}_*, \mathbf{f}|\mathbf{y})$, which can be written as

$$\begin{aligned}
\mathrm{KL}(q(\mathbf{f}_*, \mathbf{f}|\mathbf{y}) \| p(\mathbf{f}_*, \mathbf{f}|\mathbf{y})) &\stackrel{\text{def}}{=} \int q(\mathbf{f}, \mathbf{f}_*|\mathbf{y}) \log \frac{q(\mathbf{f}, \mathbf{f}_*|\mathbf{y})}{p(\mathbf{f}, \mathbf{f}_*|\mathbf{y})}\, \mathrm{d}\mathbf{f}_* \mathrm{d}\mathbf{f} \\
&= \int q(\mathbf{f}|\mathbf{y})\, p(\mathbf{f}_*|\mathbf{f}) \log \frac{q(\mathbf{f}|\mathbf{y})\, p(\mathbf{f}_*|\mathbf{f})}{p(\mathbf{f}|\mathbf{y})\, p(\mathbf{f}_*|\mathbf{f})}\, \mathrm{d}\mathbf{f}_* \mathrm{d}\mathbf{f} = \int q(\mathbf{f}|\mathbf{y}) \log \frac{q(\mathbf{f}|\mathbf{y})}{p(\mathbf{f}|\mathbf{y})}\, \mathrm{d}\mathbf{f}, \quad (64)
\end{aligned}$$

which is $\mathrm{KL}(q(\mathbf{f}|\mathbf{y}) \| p(\mathbf{f}|\mathbf{y}))$. Hence

$$\log p(\{y_{*j}^{c_j} = 1\}_{j=1}^{n_*}|\mathbf{y}) \geq \sum_{j=1}^{n_*} \int q(\mathbf{f}_{*j}|\mathbf{y}) \log p(y_{*j}^{c_j} = 1|\mathbf{f}_{*j})\, \mathrm{d}\mathbf{f}_{*j} - \mathrm{KL}(q(\mathbf{f}|\mathbf{y}) \| p(\mathbf{f}|\mathbf{y})).$$

Theorem 6 can now be applied to each summand within the first term. For the second term, the KL-divergence is also $\log p(\mathbf{y}) - \log Z_B$, and $\log Z_B$ is lower bounded by $\log Z_h$. ■

Remark 30 *Derivation 64 has been shown in (Seeger, 2002, Section 2.2) and (Rasmussen and Williams, 2006, Section 7.4.3), but there the exact prior has been used instead of the exact posterior. Our presentation closely follows (Rasmussen and Williams, 2006)'s.*

Appendix C. Optimization

We provide details on the optimization with respect to the variational parameters $\mathbf{m}$, V, $\{\mathbf{b}_i\}$ and $\{S_i\}$ in Sections C.1 to C.3. In Section C.4, we give the derivation for the updates to the hyper-parameters required for model learning in the sparse approximation.

Parameters $\mathbf{m}$ and $\mathbf{b}_i$s are updated together using Newton-Raphson in Section C.2. In regions of high-curvature, this update can be modified to include include a step-size η, the value of which can be determined using the method of false position.

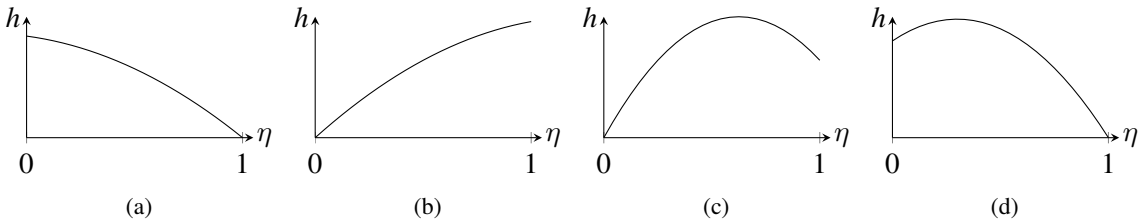


Figure 7: Four possible shapes of a segment of the concave function h within the convex combination coefficient $\eta \in [0,1]$. The horizontal axis is along η, with $S^{\mathrm{cc}} = S$ at $\eta = 0$ and $S^{\mathrm{cc}} = S^{\mathrm{fx}}$ at $\eta = 1$. The vertical axis is the variational lower bound $h(\mathbf{y}; q, \mathbf{b}, S)$. Cases (b) and (c) can be removed from consideration, since update S^{cc} is only used when h is higher at S than at S^{fx}. Case (a) is eliminated by showing that the gradient with respect to η at $\eta = 0$ is non-negative.

For V and the S_is, their fixed point updates given in Section 3 are computed and tested for improvement in the variational lower bound.[9] When the bound is worse at the fixed point updates, we search for the optimal convex combination between the previous value and the fixed-point update. For example, $S_i^{\mathrm{cc}} = (1-\eta)S_i + \eta S_i^{\mathrm{fx}}$, where S_i is the previous value and S_i^{fx} is the fixed point update. We optimize η using the method of false position with end-point down-weighting. Sections C.1 and C.3 give the gradients with respect to η and guarantee the existence of an optimal η.

C.1 Optimization for S along η

When the fixed-point S^{fx} improves the bound over S, it is accepted as an update. Otherwise, we use $S^{\mathrm{cc}} \stackrel{\mathrm{def}}{=} (1-\eta)S + \eta S^{\mathrm{fx}}$, and we search for a $\eta \in [0,1]$ that optimizes the bound using the false position method. Matrix S^{cc} is guaranteed to be positive definite, since it is a convex combination of two positive definite matrices. Let $W = S - V^{-1}$ and $W^{\mathrm{fx}} = S^{\mathrm{fx}} - V^{-1}$. Matrices W and and W^{fx} are positive semi-definite; see Lemma 28.[10] Define the gradient $\Delta \stackrel{\mathrm{def}}{=} \mathrm{d}S^{\mathrm{cc}}/\mathrm{d}\eta = S^{\mathrm{fx}} - S = W^{\mathrm{fx}} - W$. The search gradient along η is

$$\frac{\partial h(\mathbf{y}; q, \mathbf{b}, S^{\mathrm{cc}})}{\partial \eta} = \frac{1}{2} \operatorname{tr}\left[\left\{-V + (S^{\mathrm{cc}})^{-1} + (S^{\mathrm{cc}})^{-1} A^{\mathrm{cc}} (S^{\mathrm{cc}})^{-1}\right\} \Delta\right],$$

where A^{cc} is given by (18) evaluated at S^{cc} (recall that A depends on g^c, a function of S).

The optimal value of η is found using the false position method, which requires the maximal to be bracketed within $S^{\mathrm{cc}} = S$ and $S^{\mathrm{cc}} = S^{\mathrm{fx}}$. Figure 7 enumerates the four possible segments of a one-dimensional concave function. Let $\eta = 0$ at the start of the segment and $\eta = 1$ at the end. If update S^{cc} is only used when $h(\mathbf{y}; q, \mathbf{b}, S) > h(\mathbf{y}; q, \mathbf{b}, S^{\mathrm{fx}})$, then segments (b) and (c) need not be considered further. To show that the segment is always of the type given by Figure 7d, we require

9. These fixed point updates guarantees positive definiteness, a property which is absent in straightforward gradient ascent. To guarantee positive definiteness, one may suppose a viable alternative is to update the Cholesky factors or eigenvectors and eigenvalues. Unfortunately, the variational lower bound is not concave with respect to these factorizations, so they cannot be used straightforwardly.

10. In the beginning, if W is not positive semi-definite, we can re-initialize it to be so, either by using a fixed positive semi-definite matrix, or by letting W be W^{fx} in the first iteration.

$\partial h/\partial\eta$ to be non-negative at $\eta = 0$. We proceed to show this. At $\eta = 0$, we have $S^{\mathrm{cc}} = S$ and $A^{\mathrm{cc}} = A$ evaluated at S. Furthermore, we have $A = S^{\mathrm{fx}} V S^{\mathrm{fx}} - S^{\mathrm{fx}}$ since S^{fx} satisfies (60). Thus

$$\begin{aligned}
\left.\frac{\partial h(\mathbf{y}; q, \mathbf{b}, S^{\mathrm{cc}})}{\partial \eta}\right|_{\eta=0} &= \frac{1}{2}\operatorname{tr}\left[\left(-V + S^{-1} + S^{-1}\left(S^{\mathrm{fx}} V S^{\mathrm{fx}} - S^{\mathrm{fx}}\right) S^{-1}\right)\Delta\right] \\
&= \frac{1}{2}\operatorname{tr}\left[S^{-1}\left(S^{\mathrm{fx}} V S^{\mathrm{fx}} - SVS - S^{\mathrm{fx}} + S\right) S^{-1}\Delta\right] \\
&= \frac{1}{2}\operatorname{tr}\left[S^{-1}\left(W^{\mathrm{fx}} V W^{\mathrm{fx}} - WVW\right) S^{-1}\Delta\right] + \frac{1}{2}\operatorname{tr}\left[S^{-1}\Delta S^{-1}\Delta\right].
\end{aligned}$$

The second term on the right of the equality is non-negative, so its removal gives

$$\begin{aligned}
\left.\frac{\partial h(\mathbf{y}; q, \mathbf{b}, S^{\mathrm{cc}})}{\partial \eta}\right|_{\eta=0} &\geq \frac{1}{2}\operatorname{tr}\left[S^{-1}\left(W^{\mathrm{fx}} V W^{\mathrm{fx}} - WVW\right) S^{-1}\Delta\right] \\
&= \frac{1}{2}\operatorname{tr}\left[S^{-1}\left(W^{\mathrm{fx}} V W^{\mathrm{fx}} - WVW + W^{\mathrm{fx}} VW - W^{\mathrm{fx}} VW\right) S^{-1}\Delta\right] \\
&= \frac{1}{2}\operatorname{tr}\left[S^{-1}\left(W^{\mathrm{fx}} VW - WVW + W^{\mathrm{fx}} V W^{\mathrm{fx}} - W^{\mathrm{fx}} VW\right) S^{-1}\Delta\right] \\
&= \frac{1}{2}\operatorname{tr}\left[S^{-1}\left(\Delta VW + W^{\mathrm{fx}} V\Delta\right) S^{-1}\Delta\right] \\
&= \frac{1}{2}\operatorname{tr}\left[S^{-1}\Delta VW S^{-1}\Delta\right] + \frac{1}{2}\operatorname{tr}\left[S^{-1} W^{\mathrm{fx}} V\Delta S^{-1}\Delta\right] \\
&\geq 0.
\end{aligned}$$

C.2 Joint Optimization for m and b

Let $K \stackrel{\text{def}}{=} (K_1 | K_2 | \ldots | K_n)$ be a partition of K, where each K_i is a Cn-by-C matrix. We wish to optimize the variational lower bound $\log Z_h$ (21) with respect to $\mathbf{b}$ by setting $\mathbf{m} = K(\mathbf{y} - \mathbf{b})$. Call this particular setting of parameters $\log Z_h^*$. The gradient of $\log Z_h^*$ with respect to $\mathbf{b}$ including the indirect contribution from $\mathbf{m}$ is

$$\begin{aligned}
\frac{\partial \log Z_h^*}{\partial \mathbf{b}} &= \frac{\partial \log Z_h^*\big|_{\mathbf{m}\text{ constant}}}{\partial \mathbf{b}} + \frac{\mathrm{d}\mathbf{m}}{\mathrm{d}\mathbf{b}} \frac{\partial \log Z_h^*\big|_{\mathbf{b}\text{ constant}}}{\partial \mathbf{m}} \\
&= -S^{-1}(\mathbf{b} - \bar{\mathbf{g}}) - K\left(-K^{-1}\mathbf{m} + \mathbf{y} - \bar{\mathbf{g}}\right) \\
&= -(K + S^{-1})(\mathbf{b} - \bar{\mathbf{g}}).
\end{aligned}$$

Unlike case of per-datum update for $\mathbf{b}_i$, we find the fixed-point update setting $\mathbf{b}$ to $\bar{\mathbf{g}}$ ineffective. Therefore, we use the Newton-Raphson update. The required Hessian is

$$\frac{\partial^2 \log Z_h^*}{\partial \mathbf{b} \partial \mathbf{b}^{\mathrm{T}}} = -(K + S^{-1})\left(I - \frac{\partial \bar{\mathbf{g}}}{\partial \mathbf{b}^{\mathrm{T}}}\right) = -(K + S^{-1})\left(I + (\bar{G} - \tilde{G})(K + S^{-1})\right).$$

The Hessian is negative definite, so $\log Z_h$ is concave in $\mathbf{b}$. The second order update is

$$\mathbf{b}^{\mathrm{NR}} = \mathbf{b} - \left(I + (\bar{G} - \tilde{G})(K + S^{-1})\right)^{-1}(\mathbf{b} - \bar{\mathbf{g}}). \tag{65}$$

C.2.1 Joint Optimization for m and b in Sparse Approximation

For sparse approximation, the update is similar to Equation 65, the only difference being the replacement of K with $K_{\mathrm{f}}^{\mathrm{T}} K^{-1} K_{\mathrm{f}}$:

$$\mathbf{b}^{\mathrm{NR}} = \mathbf{b} - \left(I + (\bar{G} - \tilde{G})\left(K_{\mathrm{f}}^{\mathrm{T}} K^{-1} K_{\mathrm{f}} + S^{-1}\right)\right)^{-1}(\mathbf{b} - \bar{\mathbf{g}}).$$

C.3 Optimization for V along η

Let W be the block diagonal matrix with the ith block given by $W_i \stackrel{\text{def}}{=} S_i - V_i^{-1} \succ 0$. When the fixed-point $V^{\text{fx}} = (K^{-1} + W)^{-1}$ improves the bound over V, it is accepted as an update. Otherwise, we use $V^{\text{cc}} \stackrel{\text{def}}{=} (1-\eta)V + \eta V^{\text{fx}}$, and we search for a $\eta \in [0,1]$ that optimizes the bound using the false position method. Matrix V^{cc} is guaranteed to be positive definite, since it is a convex combination of two positive definite matrices. Let $\Delta \stackrel{\text{def}}{=} \mathrm{d}V^{\text{cc}}/\mathrm{d}\eta = V^{\text{fx}} - V$. Below, we shall make explicit that the lower bound $\log Z_h$ is parameterized by the covariance V of the variational posterior. The search gradient is

$$\frac{\partial \log Z_h(V^{\text{cc}})}{\partial \eta} = \frac{1}{2}\operatorname{tr}\left((V^{\text{cc}})^{-1}\Delta\right) - \frac{1}{2}\operatorname{tr}\left(K^{-1}\Delta\right) + \frac{1}{2}\sum_{i=1}^{n}\operatorname{tr}\left((V_i^{\text{cc}})^{-1}\Delta_i\right) - \frac{1}{2}\sum_{i=1}^{n}\operatorname{tr}\left(S_i\Delta_i\right),$$

where V_i^{cc} and Δ_i are the ith blocks along the diagonal of V^{cc} and Δ respectively. The update V^{cc} is only used when $\log Z_h(V) > \log Z_h(V^{\text{cc}})$. By arguments similar to those for the update S^{cc} discussed in Appendix C.1, we can guarantee that there is a maximum between V and V^{fx} by showing that $\partial Z_h/\partial\eta$ is non-negative at $\eta = 0$:

$$\begin{aligned}
&\left.\frac{\partial \log Z_h(V^{\text{cc}})}{\partial \eta}\right|_{\eta=0} \\
&= \frac{1}{2}\operatorname{tr}\left(V^{-1}\Delta\right) - \frac{1}{2}\operatorname{tr}\left(K^{-1}\Delta\right) - \frac{1}{2}\sum_{i=1}^{n}\operatorname{tr}\left(W_i\Delta_i\right) \\
&= \frac{1}{2}\operatorname{tr}\left(V^{-1}\Delta\right) - \frac{1}{2}\operatorname{tr}\left(K^{-1}\Delta\right) - \frac{1}{2}\operatorname{tr}\left(W\Delta\right) && \text{(since } W \text{ is block diagonal)} \\
&= \frac{1}{2}\operatorname{tr}\left(V^{-1}\Delta\right) - \frac{1}{2}\operatorname{tr}\left(K^{-1}\Delta\right) - \frac{1}{2}\operatorname{tr}\left(\left((V^{\text{fx}})^{-1} - K^{-1}\right)\Delta\right) && \text{(since } V^{\text{fx}} = (K^{-1}+W)^{-1}\text{)} \\
&= \frac{1}{2}\operatorname{tr}\left(\left(V^{-1} - (V^{\text{fx}})^{-1}\right)\Delta\right) \\
&= \frac{1}{2}\operatorname{tr}\left((V^{\text{fx}})^{-1}(V^{\text{fx}} - V)V^{-1}\Delta\right) \\
&= \frac{1}{2}\operatorname{tr}\left((V^{\text{fx}})^{-1}\Delta V^{-1}\Delta\right) \\
&\geq 0.
\end{aligned}$$

C.3.1 Optimization for V along η in Sparse Approximation

We use the same strategy in the sparse approximation. Let the covariance of the inducing variables be $V^{\text{cc}} \stackrel{\text{def}}{=} (1-\eta)V + \eta V^{\text{fx}}$. The covariance of the latent variables is $V_{\text{f}}^{\text{cc}} = (1-\eta)V_{\text{f}} + \eta V_{\text{f}}^{\text{fx}}$. Let $\Delta \stackrel{\text{def}}{=} \mathrm{d}V^{\text{cc}}/\mathrm{d}\eta = V^{\text{fx}} - V$, and let $\Delta_{\text{f}} \stackrel{\text{def}}{=} \mathrm{d}V_{\text{f}}^{\text{cc}}/\mathrm{d}\eta = V_{\text{f}}^{\text{fx}} - V_{\text{f}} = K_{\text{f}}^{\text{T}}K^{-1}\Delta K^{-1}K_{\text{f}}$. The gradient along $\eta \in [0,1]$ for the false position update is

$$\frac{\partial \log \tilde{Z}_h(V^{\text{cc}})}{\partial \eta} = \frac{1}{2}\operatorname{tr}((V^{\text{cc}})^{-1}\Delta) - \frac{1}{2}\operatorname{tr}(K^{-1}\Delta) + \frac{1}{2}\sum_{i=1}^{n}\operatorname{tr}((V_{\text{f}i}^{\text{cc}})^{-1}\Delta_{\text{f}i}) - \frac{1}{2}\sum_{i=1}^{n}\operatorname{tr}(S_i\Delta_{\text{f}i}).$$

The proof that $\partial \log \tilde{Z}_h(V^{\text{cc}})/\partial\eta$ is non-negative at $\eta = 0$ follows the same reasoning as that for the non-sparse approximation.

C.4 Hyper-parameter Estimation in Sparse Approximation

In this section, we give the gradients of the optimized variational lower bound $\tilde{Z}_h^*$ for the sparse case. First, we introduce

$$T \stackrel{\text{def}}{=} K^{-1} - K^{-1} V K^{-1}, \qquad \Gamma_j \stackrel{\text{def}}{=} K\left(\frac{\partial K^{-1} K_{\mathrm{f}}}{\partial \theta_j}\right) = \frac{\partial K_{\mathrm{f}}}{\partial \theta_j} - \frac{\partial K}{\partial \theta_j} K^{-1} K_{\mathrm{f}}. \tag{66}$$

Then

$$\frac{\partial \mathbf{m}_{\mathrm{f}}}{\partial \theta_j} = \Gamma_j^{\mathrm{T}} K^{-1} \mathbf{m} = \Gamma_j^{\mathrm{T}} \boldsymbol{\alpha}, \tag{67}$$

$$\frac{\partial V_{\mathrm{f}}}{\partial \theta_j} = \frac{\partial K_{\mathrm{ff}}}{\partial \theta_j} - K_{\mathrm{f}}^{\mathrm{T}} K^{-1} \frac{\partial K}{\partial \theta_j} K^{-1} K_{\mathrm{f}} - \Gamma_j^{\mathrm{T}} T K_{\mathrm{f}} - K_{\mathrm{f}}^{\mathrm{T}} T \Gamma_j. \tag{68}$$

The gradient is

$$\begin{aligned}
\frac{\mathrm{d}\log \tilde{Z}_h^*}{\mathrm{d}\theta_j} = & -\frac{1}{2}\operatorname{tr}\left(K^{-1}\frac{\partial K}{\partial \theta_j}\right) + \frac{1}{2}\operatorname{tr}\left(K^{-1} V K^{-1}\frac{\partial K}{\partial \theta_j}\right) + \frac{1}{2}\mathbf{m}^{\mathrm{T}} K^{-1}\frac{\partial K}{\partial \theta_j} K^{-1}\mathbf{m} \\
& + \frac{\partial \mathbf{m}_{\mathrm{f}}^{\mathrm{T}}}{\partial \theta_j}\mathbf{y} + \frac{1}{2}\sum_{i=1}^{n}\operatorname{tr}\left((V_{\mathrm{f}i})^{-1}\frac{\partial V_{\mathrm{f}i}}{\partial \theta_j}\right) - \frac{1}{2}\sum_{i=1}^{n}\operatorname{tr}\left(S_i\frac{\partial V_{\mathrm{f}i}}{\partial \theta_j}\right) - \sum_{i=1}^{n}\sum_{c=1}^{C}\bar{g}_i^c\frac{\partial \mathbf{m}_{\mathrm{f}i}^c}{\partial \theta_j} \\
= & \frac{1}{2}\operatorname{tr}\left(\left(\boldsymbol{\alpha}\boldsymbol{\alpha}^{\mathrm{T}} - T\right)\frac{\partial K}{\partial \theta_j}\right) + \frac{\partial \mathbf{m}_{\mathrm{f}}^{\mathrm{T}}}{\partial \theta_j}(\mathbf{y} - \bar{\mathbf{g}}) - \frac{1}{2}\operatorname{tr}\left(W_{\mathrm{f}}\frac{\partial V_{\mathrm{f}}}{\partial \theta_j}\right),
\end{aligned}$$

where $\boldsymbol{\alpha} \stackrel{\text{def}}{=} K^{-1}\mathbf{m}$, $W_{\mathrm{f}i} \stackrel{\text{def}}{=} S_i - V_{\mathrm{f}i}^{-1}$ and W_{f} is a block diagonal matrix of the $W_{\mathrm{f}i}$s. Let us investigate the second term in the last expression above. Using (67), the definition of Γ_j in (66) and the identity $\mathbf{m} = K_{\mathrm{f}}(\mathbf{y} - \bar{\mathbf{g}})$ at optimality (see Section 4.2), we have

$$\begin{aligned}
\frac{\partial \mathbf{m}_{\mathrm{f}}^{\mathrm{T}}}{\partial \theta_j}(\mathbf{y} - \bar{\mathbf{g}}) = \boldsymbol{\alpha}^{\mathrm{T}}\Gamma_j(\mathbf{y} - \bar{\mathbf{g}}) &= \boldsymbol{\alpha}^{\mathrm{T}}\frac{\partial K_{\mathrm{f}}}{\partial \theta_j}(\mathbf{y} - \bar{\mathbf{g}}) - \boldsymbol{\alpha}^{\mathrm{T}}\frac{\partial K}{\partial \theta_j}K^{-1}K_{\mathrm{f}}(\mathbf{y} - \bar{\mathbf{g}}) \\
&= \boldsymbol{\alpha}^{\mathrm{T}}\frac{\partial K_{\mathrm{f}}}{\partial \theta_j}(\mathbf{y} - \bar{\mathbf{g}}) - \boldsymbol{\alpha}^{\mathrm{T}}\frac{\partial K}{\partial \theta_j}\boldsymbol{\alpha}.
\end{aligned}$$

We now turn to the trace expression in the gradient of $\log \tilde{Z}_h^*$. Using (68), the definition of Γ_j in (66) and the invariance of trace under cyclic permutations, we obtain

$$\operatorname{tr}\left(W_{\mathrm{f}}\frac{\partial V_{\mathrm{f}}}{\partial \theta_j}\right) = \operatorname{tr}\left(W_{\mathrm{f}}\frac{\partial K_{\mathrm{ff}}}{\partial \theta_j}\right) - \operatorname{tr}\left(W\frac{\partial K}{\partial \theta_j}\right) - 2\operatorname{tr}\left(W_{\mathrm{f}}K_{\mathrm{f}}^{\mathrm{T}}T\frac{\partial K_{\mathrm{f}}}{\partial \theta_j}\right) + 2\operatorname{tr}\left(WKT\frac{\partial K}{\partial \theta_j}\right),$$

where we have used $W \stackrel{\text{def}}{=} K^{-1}K_{\mathrm{f}}W_{\mathrm{f}}K_{\mathrm{f}}^{\mathrm{T}}K^{-1}$. Further substituting the definition for T from (66) into the last term and simplifying using $W = V^{-1} - K^{-1}$ at optimality (see Equation 33) gives

$$\operatorname{tr}\left(W_{\mathrm{f}}\frac{\partial V_{\mathrm{f}}}{\partial \theta_j}\right) = \operatorname{tr}\left(W_{\mathrm{f}}\frac{\partial K_{\mathrm{ff}}}{\partial \theta_j}\right) + \operatorname{tr}\left((W - 2T)\frac{\partial K}{\partial \theta_j}\right) - 2\operatorname{tr}\left(W_{\mathrm{f}}K_{\mathrm{f}}^{\mathrm{T}}T\frac{\partial K_{\mathrm{f}}}{\partial \theta_j}\right).$$

Putting the simplifications back into the gradient of $\log \tilde{Z}_h^*$ gives

$$\frac{\mathrm{d}\log \tilde{Z}_h^*}{\mathrm{d}\theta_j} = -\frac{1}{2}\operatorname{tr}\left(\left(\boldsymbol{\alpha}\boldsymbol{\alpha}^{\mathrm{T}} - T + W\right)\frac{\partial K}{\partial \theta_j}\right) + \operatorname{tr}\left(\left((\mathbf{y} - \bar{\mathbf{g}})\boldsymbol{\alpha}^{\mathrm{T}} + W_{\mathrm{f}}K_{\mathrm{f}}^{\mathrm{T}}T\right)\frac{\partial K_{\mathrm{f}}}{\partial \theta_j}\right) - \frac{1}{2}\operatorname{tr}\left(W_{\mathrm{f}}\frac{\partial K_{\mathrm{ff}}}{\partial \theta_j}\right).$$

Appendix D. Selection of Inducing Sites

This appendix details the derivation of criterion d_1 used for selecting inducing sites actively.

D.1 A Lower Bound on the Increase to the Marginal Likelihood Bound

Our objective is to add an inducing site $\tilde{\mathbf{x}}_*$ to the current inducing set $\tilde{X}$ so as to maximize the lower bound (30) on the increase in $\log \tilde{Z}_h$. The random variables at $\tilde{\mathbf{x}}_*$ and $\tilde{X}$ are denoted by z_* and $\mathbf{z}$. Let $\mathbf{z}_* \stackrel{\text{def}}{=} (\mathbf{z}^{\mathrm{T}}, z_*)^{\mathrm{T}}$ and $\tilde{X}_* \stackrel{\text{def}}{=} \tilde{X} \cup \{\tilde{\mathbf{x}}_*\}$. The prior on $\mathbf{z}_*$ and $\mathbf{f}$ is

$$p\left(\begin{pmatrix}\mathbf{z}_*\\ \mathbf{f}\end{pmatrix}\right) \stackrel{\text{def}}{=} \mathcal{N}\left(\mathbf{0}, \begin{pmatrix}K_* & K_{\mathrm{f}*}\\ K_{\mathrm{f}*}^{\mathrm{T}} & K_{\mathrm{ff}}\end{pmatrix}\right), \qquad \text{where} \qquad K_* \stackrel{\text{def}}{=} \begin{pmatrix}K & \mathbf{k}_*\\ \mathbf{k}_*^{\mathrm{T}} & k_{**}\end{pmatrix} \qquad K_{\mathrm{f}*} \stackrel{\text{def}}{=} \begin{pmatrix}K_{\mathrm{f}}\\ \mathbf{k}_{\mathrm{f}*}^{\mathrm{T}}\end{pmatrix}.$$

Let

$$\{\mathbf{m}, V, \{\mathbf{b}_i\}, \{S_i\}\} = \arg \max_{\mathbf{m}, V, \{\mathbf{b}_i\}, \{S_i\}} \log \tilde{Z}_h(\mathbf{m}, V, \{\mathbf{b}_i\}, \{S_i\}; \tilde{X}),$$

where $\mathbf{m}$ and V are the mean and covariance of $\mathbf{z}$ in the approximate posterior using inducing set $\tilde{X}$. Let $\log \tilde{Z}_h^*(\tilde{X})$ be the optimal value of the objective function in the equation above. Then a lower bound on the increase is

$$d_1(\tilde{\mathbf{x}}_* | \tilde{X}) \stackrel{\text{def}}{=} \max_{m_*, v_{**}, \mathbf{v}_*} \log Z_h(\mathbf{m}_*, V_*, \{\mathbf{b}_i\}, \{S_i\}; \tilde{X}_*) - \log Z_h^*(\tilde{X}), \tag{69}$$

where the mean $\mathbf{m}_*$ and covariance V_* of the approximate posterior on $\mathbf{z}_*$ are constrained:

$$\mathbf{m}_* \stackrel{\text{def}}{=} \begin{pmatrix}\mathbf{m}\\ m_*\end{pmatrix}, \qquad V_* \stackrel{\text{def}}{=} \begin{pmatrix}V & \mathbf{v}_*\\ \mathbf{v}_*^{\mathrm{T}} & v_{**}\end{pmatrix}. \tag{70}$$

Denote the posterior distribution of the latent function values $\mathbf{f}$ under the sparse approximation by $q_*(\mathbf{f}|\mathbf{y}) \stackrel{\text{def}}{=} q(\mathbf{f}|\mathbf{y}, \tilde{X}_*)$. This is the approximate posterior using the inducing sites $\tilde{X}_*$, while $q(\mathbf{f}|\mathbf{y}) \stackrel{\text{def}}{=} q(\mathbf{f}|\mathbf{y}, \tilde{X})$ is the posterior using $\tilde{X}$. The choice of the factored form of the approximate posterior in Equation 29 means that

$$q(\mathbf{f}|\mathbf{y}) = \int p(\mathbf{f}|\mathbf{z}) q(\mathbf{z}|\mathbf{y}) \mathrm{d}\mathbf{z}, \qquad q_*(\mathbf{f}|\mathbf{y}) = \int p(\mathbf{f}|\mathbf{z}_*) q(\mathbf{z}_*|\mathbf{y}) \mathrm{d}\mathbf{z}_*.$$

Expressions for the mean $\mathbf{m}_{\mathrm{f}}$ and covariance V_{f} of $\mathbf{f}$ under $q(\mathbf{f}|\mathbf{y})$ are given in Equation 31. The mean $\mathbf{m}_{\mathrm{f}*}$ and covariance $V_{\mathrm{f}*}$ of $\mathbf{f}$ under $q_*(\mathbf{f}|\mathbf{y})$ are

$$\begin{aligned}\mathbf{m}_{\mathrm{f}*} &= K_{\mathrm{f}*}^{\mathrm{T}} K_*^{-1} \mathbf{m}_* & V_{\mathrm{f}*} &= K_{\mathrm{ff}} - K_{\mathrm{f}*}^{\mathrm{T}} K_*^{-1} K_{\mathrm{f}*} + K_{\mathrm{f}*}^{\mathrm{T}} K_*^{-1} V_* K_*^{-1} K_{\mathrm{f}*}\\ &= \mathbf{m}_{\mathrm{f}} + \mu \boldsymbol{\kappa}; & &= V_{\mathrm{f}} - (\kappa - \chi) \boldsymbol{\kappa} \boldsymbol{\kappa}^{\mathrm{T}} + \boldsymbol{\psi} \boldsymbol{\kappa}^{\mathrm{T}} + \boldsymbol{\kappa} \boldsymbol{\psi}^{\mathrm{T}},\end{aligned} \tag{71}$$

where

$$\kappa \stackrel{\text{def}}{=} k_{**} - \mathbf{k}_*^{\mathrm{T}} K^{-1} \mathbf{k}_*, \qquad \nu \stackrel{\text{def}}{=} v_{**} - \mathbf{v}_*^{\mathrm{T}} V^{-1} \mathbf{v}_*, \qquad \chi \stackrel{\text{def}}{=} \nu + \boldsymbol{\nu}^{\mathrm{T}} V^{-1} \boldsymbol{\nu}, \qquad \mu \stackrel{\text{def}}{=} m_* - \mathbf{k}_*^{\mathrm{T}} K^{-1} \mathbf{m},$$
$$\boldsymbol{\kappa} \stackrel{\text{def}}{=} (\mathbf{k}_{\mathrm{f}*} - K_{\mathrm{f}}^{\mathrm{T}} K^{-1} \mathbf{k}_*)/\kappa, \qquad \boldsymbol{\nu} \stackrel{\text{def}}{=} \mathbf{v}_* - V K^{-1} \mathbf{k}_*, \qquad \boldsymbol{\psi} \stackrel{\text{def}}{=} K_{\mathrm{f}}^{\mathrm{T}} K^{-1} \boldsymbol{\nu}.$$

The two expressions in (71) relate the parameters for $q_*(\mathbf{f}|\mathbf{y})$ to those for $q(\mathbf{f}|\mathbf{y})$. The derivation uses the Banachiewicz inversion formula (Puntanen and Styan, 2005) on $(K_*)^{-1}$. The term

$(\kappa-\chi)$ in the expression for $V_{\mathrm{f}*}$ is non-negative because $K_* \succeq V_*$ at the stationary, which gives $(\kappa-\chi) = \begin{pmatrix}-K^{-1}\mathbf{k}_* & 1\end{pmatrix}(K_*-V_*)\begin{pmatrix}-K^{-1}\mathbf{k}_* & 1\end{pmatrix}^{\mathrm{T}} \geq 0$. The posterior covariance for ith data point under q_* is the ith C-by-C diagonal block matrix of $V_{\mathrm{f}*}$:

$$V_{\mathrm{f}*i} = V_{\mathrm{f}i} - (\kappa-\chi)\boldsymbol{\kappa}_i\boldsymbol{\kappa}_i^{\mathrm{T}} + \boldsymbol{\psi}_i\boldsymbol{\kappa}_i^{\mathrm{T}} + \boldsymbol{\kappa}_i\boldsymbol{\psi}_i^{\mathrm{T}}, \tag{72}$$

where $V_{\mathrm{f}i}$ is the ith C-by-C diagonal block matrix of of V_{f}, and $\boldsymbol{\kappa}_i$ (resp. $\boldsymbol{\psi}_i$) is the ith C-vector of $\boldsymbol{\kappa}$ (resp. $\boldsymbol{\psi}$). Using Lemma 23, we obtain

$$|V_{\mathrm{f}*i}| = \omega_i|V_{\mathrm{f}i}|, \qquad \text{where} \qquad \omega_i \stackrel{\text{def}}{=} \left(1+\boldsymbol{\kappa}_i^{\mathrm{T}}V_{\mathrm{f}i}^{-1}\boldsymbol{\psi}_i\right)^2 - \boldsymbol{\kappa}_i^{\mathrm{T}}V_{\mathrm{f}i}^{-1}\boldsymbol{\kappa}_i\left(\kappa-\chi+\boldsymbol{\psi}_i^{\mathrm{T}}V_{\mathrm{f}i}^{-1}\boldsymbol{\psi}_i\right). \tag{73}$$

Since $|V_{\mathrm{f}*i}| > 0$ and $|V_{\mathrm{f}i}| > 0$, so $\omega_i > 0$. We are now ready to express d_1 defined by (69) in terms of the parameters, separating $\log\tilde{Z}_h$ into its summands expressed in Equation 30:

$$d_1(\tilde{\mathbf{x}}_*|\tilde{X}) = \max_{m_*,v_{**},\mathbf{v}_*}\left(d_{\mathrm{KL}}(m_*,v_{**},\mathbf{v}_*,\tilde{\mathbf{x}}_*|\tilde{X}) + \sum_{i=1}^{n} d_h^i(m_*,v_{**},\mathbf{v}_*,\tilde{\mathbf{x}}_*|\tilde{X})\right),$$

where

$$\begin{aligned}
d_{\mathrm{KL}}(m_*,v_{**},\mathbf{v}_*,\tilde{\mathbf{x}}_*|\tilde{X}) &\stackrel{\text{def}}{=} -\mathrm{KL}(q(\mathbf{z}_* \mid \mathbf{y}) \,\|\, p(\mathbf{z}_*)) + \mathrm{KL}(q(\mathbf{z} \mid \mathbf{y}) \,\|\, p(\mathbf{z})) \\
&= \frac{1}{2} + \frac{1}{2}\log\frac{\nu}{\kappa} - \frac{\chi}{2\kappa} - \frac{\mu^2}{2\kappa}; \\
d_h^i(m_*,v_{**},\mathbf{v}_*,\tilde{\mathbf{x}}_*|\tilde{X}) &\stackrel{\text{def}}{=} h(\mathbf{y}_i;q_{*i},\mathbf{b}_i,S_i) - h(\mathbf{y}_i;q_i,\mathbf{b}_i,S_i) \\
&= \frac{1}{2}\log\omega_i + \frac{\kappa-\chi}{2}\boldsymbol{\kappa}_i^{\mathrm{T}}S_i\boldsymbol{\kappa}_i - \boldsymbol{\kappa}_i^{\mathrm{T}}S_i\boldsymbol{\psi}_i + \mu\boldsymbol{\kappa}_i^{\mathrm{T}}\mathbf{y}_i - \log\sum_{c=1}^{C}\bar{g}_i^c e^{\mu\boldsymbol{\kappa}_i^{\mathrm{T}}\mathbf{e}^c}.
\end{aligned}$$

Lemma 18 is used to obtain the second expression for d_{KL}, and (71) to (73) are used to obtain the second expression for d_h^i. The q_{*i} in the definition of d_h^i refers to the the marginal for $\mathbf{f}_i$ under $q_*(\mathbf{f}|\mathbf{y})$, while the $\bar{g}_i^c$s in the term for d_h^i is evaluated under $q(\mathbf{f}|\mathbf{y})$.

D.2 Optimizing the Lower Bound on the Increase

Let

$$d_1(m_*,v_{**},\mathbf{v}_*,\tilde{\mathbf{x}}_*|\tilde{X}) \stackrel{\text{def}}{=} d_{\mathrm{KL}}(m_*,v_{**},\mathbf{v}_*,\tilde{\mathbf{x}}_*|\tilde{X}) + \sum_{i=1}^{n} d_h^i(m_*,v_{**},\mathbf{v}_*,\tilde{\mathbf{x}}_*|\tilde{X})$$

be the objective function within $d_1(\tilde{\mathbf{x}}_*|\tilde{X})$. Within this section, d_1 shall refer to this objective function instead of its maximum. The contributions from m_* and $(\mathbf{v}_*, v_{**})$ are decoupled in this objective, so the search for the optimal m_* and $(\mathbf{v}_*, v_{**})$ can be perform separately.

Instead of using m_*, v_{**} and $\mathbf{v}_*$ as the variational parameters, we can treat μ, ν and $\boldsymbol{\nu}$ as the variational parameters, and then define m_{**}, v_{**} and $\mathbf{v}_*$ as functions of them:

$$m_* \stackrel{\text{def}}{=} \mu + \mathbf{k}_*^{\mathrm{T}}K^{-1}\mathbf{m}, \qquad v_{**} \stackrel{\text{def}}{=} \nu + \mathbf{v}_*^{\mathrm{T}}V^{-1}\mathbf{v}_*, \qquad \mathbf{v}_* \stackrel{\text{def}}{=} \boldsymbol{\nu} + VK^{-1}\mathbf{k}_*.$$

This is valid and does not change the search space of the original variational parameters. During the optimization, the positive definiteness of V_* (70) can be ensured by constraining the Schur complement ν to be positive (see Horn and Johnson 1985, Theorem 7.7.6). Under this re-parametrization, d_1 is concave in μ and ν but not necessarily concave in $\boldsymbol{\nu}$ because of the positive quadratic term within ω_i (73).

Below, we give the gradient updates for μ and ν.

D.2.1 NEWTON-RAPHSON UPDATES FOR μ

Let $g^c_{*i} \stackrel{\text{def}}{=} \bar{g}^c_i \exp \mu \boldsymbol{\kappa}_i^{\mathrm{T}} \mathbf{e}^c$, $\bar{g}^c_{*i} \stackrel{\text{def}}{=} g^c_{*i} / \sum_{c'=1}^{C} g^{c'}_{*i}$, $\bar{\mathbf{g}}_{*i} \stackrel{\text{def}}{=} \left(\bar{g}^1_{*i}, \ldots, \bar{g}^C_{*i}\right)^{\mathrm{T}}$, $\bar{\mathbf{g}}_*$ be the stacking of $\bar{\mathbf{g}}_{*1}, \ldots, \bar{\mathbf{g}}_{*n}$, $\bar{G}_*$ be the diagonal matrix with $\bar{\mathbf{g}}_*$ down its diagonal, and $\tilde{G}_*$ be a nC-by-nC block diagonal matrix where the ith block is $\bar{\mathbf{g}}_{*i}\bar{\mathbf{g}}_{*i}^{\mathrm{T}}$. The Newton-Raphson update for μ is obtained from the first and the second derivatives $\partial d_1/\partial\mu = -\mu/\kappa + \boldsymbol{\kappa}^{\mathrm{T}}\mathbf{y} - \boldsymbol{\kappa}^{\mathrm{T}}\bar{\mathbf{g}}_*$ and $\partial^2 d_1/\partial\mu^2 = -1/\kappa - \boldsymbol{\kappa}^{\mathrm{T}}(\bar{G}_* - \tilde{G}_*)\boldsymbol{\kappa}$.

D.2.2 "BEYOND" NEWTON RAPHSON UPDATES FOR ν_{**}

We give an update for ν that converges faster than the Newton-Raphson update for $\log\nu$ when the optimal value is small, using a non-quadratic local approximation (Minka, 2002):

$$\tilde{d}_1(\nu) = \text{constant} + \frac{1}{2}\log\nu + \frac{n}{2}\log(\nu + a) - \frac{b}{2}\nu,$$

where a and b are parameters in the approximation. Within the approximation, ν is constrained to be positive due to the second term. By equating the first two derivatives of $d_1(\nu)$ to those of $\tilde{d}_1(\nu)$ at a given ν, we obtain

$$a = \sqrt{\frac{n}{\sum_{i=1}^n \tau_i^2}} - \nu, \qquad\qquad b = \sqrt{n\sum_{i=1}^n \tau_i^2} + \frac{1}{\kappa} + \boldsymbol{\kappa}^{\mathrm{T}} S \boldsymbol{\kappa} - \sum_{i=1}^n \tau_i,$$

where the positive branch of the square-root for a is used so that $a+\nu$ remains positive. Fixing a and b, the update for ν is obtained by equating the gradient of $\tilde{d}_1(\nu)$ at the updated point, say ν^{bNR}, to zero. This involve a quadratic equation, and we use its positive solution

$$\nu^{\mathrm{bNR}} = \frac{-(ab-n-1) + \sqrt{(ab-n-1)^2 + 4ab}}{2b}. \tag{74}$$

We prove that this update is guaranteed to be positive in Theorem 32 below.

Lemma 31 $\tau_i \stackrel{\text{def}}{=} \boldsymbol{\kappa}_i^{\mathrm{T}} V_{\mathrm{f}i}^{-1} \boldsymbol{\kappa}_i / \omega_i < 1/\nu$.

Proof Define

$$\tilde{V}_* \stackrel{\text{def}}{=} \begin{pmatrix} V & \mathbf{v}_* \\ \mathbf{v}_*^{\mathrm{T}} & \mathbf{v}^{\mathrm{T}} V^{-1} \mathbf{v} \end{pmatrix},$$

which is positive semi-definite (Horn and Johnson, 1985, Theorem 7.7.6). Then $\tilde{V}_{\mathrm{f}*}$ below is positive definite since the covariance of the joint prior $p(\mathbf{z}_*, \mathbf{f})$ is positive definite.

$$\tilde{V}_{\mathrm{f}*} \stackrel{\text{def}}{=} K_{\mathrm{ff}} - K_{\mathrm{f}*}^{\mathrm{T}} K_*^{-1} K_{\mathrm{f}*} + K_{\mathrm{f}*}^{\mathrm{T}} K_*^{-1} \tilde{V}_* K_*^{-1} K_{\mathrm{f}*} = V_{\mathrm{f}} - (\kappa - (\chi - \nu))\boldsymbol{\kappa}\boldsymbol{\kappa}^{\mathrm{T}} + \boldsymbol{\psi}\boldsymbol{\kappa}^{\mathrm{T}} + \boldsymbol{\kappa}\boldsymbol{\psi}^{\mathrm{T}}.$$

Similarly, the ith diagonal C-by-C sub-matrix of $\tilde{V}_{\mathrm{f}*}$ given by

$$\tilde{V}_{\mathrm{f}*i} = V_{\mathrm{f}i} - (\kappa - (\chi - \nu))\boldsymbol{\kappa}_i\boldsymbol{\kappa}_i^{\mathrm{T}} + \boldsymbol{\psi}_i\boldsymbol{\kappa}_i^{\mathrm{T}} + \boldsymbol{\kappa}_i\boldsymbol{\psi}_i^{\mathrm{T}}$$

is positive definite. Using Lemma 23, we obtain $|\tilde{V}_{\mathrm{f}*i}| = \tilde{\omega}_i |V_{\mathrm{f}i}|$, where

$$\tilde{\omega}_i \stackrel{\text{def}}{=} \left(1 + \boldsymbol{\kappa}_i^{\mathrm{T}} V_{\mathrm{f}i}^{-1} \boldsymbol{\psi}_i\right)^2 - \boldsymbol{\kappa}_i^{\mathrm{T}} V_{\mathrm{f}i}^{-1} \boldsymbol{\kappa}_i \left(\kappa - (\chi - \nu) + \boldsymbol{\psi}_i^{\mathrm{T}} V_{\mathrm{f}i}^{-1} \boldsymbol{\psi}_i\right)$$

is positive because both $|\tilde{V}_{\mathrm{f}*i}|$ and $|V_{\mathrm{f}i}|$ are positive. But $\tilde{\omega}_i = \omega_i - \nu\,\boldsymbol{\kappa}_i^{\mathrm{T}} V_{\mathrm{f}i}^{-1} \boldsymbol{\kappa}_i$. Thus

$$\omega_i = \tilde{\omega}_i + \nu\,\boldsymbol{\kappa}_i^{\mathrm{T}} V_{\mathrm{f}i}^{-1} \boldsymbol{\kappa}_i > \nu\,\boldsymbol{\kappa}_i^{\mathrm{T}} V_{\mathrm{f}i}^{-1} \boldsymbol{\kappa}_i.$$

So $\tau_i < 1/\nu$. ■

Theorem 32 *Update ν^{bNR} given in Equation 74 is positive.*

Proof This update is guaranteed to be positive when a and b are both positive. Parameter b is positive because $1/\kappa + \boldsymbol{\kappa}^{\mathrm{T}} S \boldsymbol{\kappa}$ is positive and $\sum_{i=1}^{n} \tau_i \leq (n \sum_{i=1}^{n} \tau_i^2)^{1/2}$ by applying the Cauchy-Schwarz inequality. Parameter a is positive because $\tau_i < 1/\nu$ from Lemma 31. ■

Appendix E. Implementation Considerations

This appendix considers the details for an implementation of the variational bound optimization presented in this paper.

E.1 Matrix Inversion in Update for b in Sparse Approximation

For the sparse approximation, the Newton-Raphson update for $\mathbf{b}$ given in Appendix C.2.1 requires inverting $X \stackrel{\text{def}}{=} I + (\bar{G} - \tilde{G}) \left(K_{\mathrm{f}}^{\mathrm{T}} K^{-1} K_{\mathrm{f}} + S^{-1} \right)$ of order Cn-by-Cn. To avoid $O(C^3 n^3)$ computation, we apply the Woodbury's inversion lemma thrice. Let $M \stackrel{\text{def}}{=} (S + \bar{G} - \tilde{G})^{-1}$, and $\tilde{L}_K \stackrel{\text{def}}{=} K_{\mathrm{f}}^{\mathrm{T}} L_K^{-\mathrm{T}}$, where L_K is the lower Cholesky factor of K. Then

$$
\begin{aligned}
X^{-1} &= I - (\bar{G} - \tilde{G}) \left(\left(\tilde{L}_K \tilde{L}_K^{\mathrm{T}} + S^{-1} \right)^{-1} + (\bar{G} - \tilde{G}) \right)^{-1} \\
&= I - (\bar{G} - \tilde{G}) \left(M^{-1} - S\tilde{L}_K \left(I + \tilde{L}_K^{\mathrm{T}} S \tilde{L}_K \right)^{-1} \tilde{L}_K^{\mathrm{T}} S \right)^{-1} \\
&= I - (\bar{G} - \tilde{G}) \left(M + MS\tilde{L}_K \left(I + \tilde{L}_K^{\mathrm{T}} (S - SMS) \tilde{L}_K \right)^{-1} \tilde{L}_K^{\mathrm{T}} SM \right) \\
&= SM - (S - SMS) \tilde{L}_K \left(I + \tilde{L}_K^{\mathrm{T}} (S - SMS) \tilde{L}_K \right)^{-1} \tilde{L}_K^{\mathrm{T}} SM,
\end{aligned}
$$

where we have substituted $(\bar{G} - \tilde{G}) = M^{-1} - S$ to obtain the last expression.

E.2 Better Conditioned Updates for V

In this section, we give better conditioned updates for the optimization of V.

E.2.1 NON-SPARSE CASE

Equation 28 in Section 3.4 gives the fixed-point update $V^{\mathrm{fx}} = (K^{-1} + W)^{-1}$ for the variational parameter V, where W is rank deficient (see Lemma 28). We factorize $W = L_W L_W^{\mathrm{T}}$, and introduce $B \stackrel{\text{def}}{=} L_W^{\mathrm{T}} K L_W + I$ and $T \stackrel{\text{def}}{=} L_W B^{-1} L_W^{\mathrm{T}}$. Then the Woodbury's inversion lemma gives $V^{\mathrm{fx}} = K - KTK$. The optimal update is given by the best convex combination of V and V^{fx}. Let T^{old} be such that

$$
V = K - KT^{\mathrm{old}}K. \tag{75}
$$

The best convex combination is the one optimized over $\eta \in [0,1]$ in $V^{\mathrm{cc}} = K - KT^{\mathrm{cc}}K$, where $T^{\mathrm{cc}} \stackrel{\text{def}}{=} (1-\eta)T^{\mathrm{old}} + \eta T$. The update for V^{cc} implies that T^{cc} is the T^{old} for the next iteration, so (75) is always possible. Moreover, with $\Delta \stackrel{\text{def}}{=} \mathrm{d}V^{\mathrm{cc}}/\mathrm{d}\eta$, we also have $\Delta = K(T^{\mathrm{old}} - T)K$ and $K^{-1}\Delta = (T^{\mathrm{old}} - T)K$.

E.2.2 SPARSE CASE

Equation 33 in Section 4.2 gives the fixed-point update in the sparse case:

$$V^{\mathrm{fx}} = \left(K^{-1} + K^{-1}K_{\mathrm{f}}W_{\mathrm{f}}K_{\mathrm{f}}^{\mathrm{T}}K^{-1}\right)^{-1}.$$

If we were to proceed as for the non-sparse case using the Woodbury's inversion lemma, then the inversion of a Cn-by-Cn matrix would be required. However, this is to be avoided in the sparse approximation, which aims to reduce time complexity. Instead, we compute $V^{\mathrm{fx}} = L_K(I + \tilde{L}_K^{\mathrm{T}}W_{\mathrm{f}}\tilde{L}_K)^{-1}L_K^{\mathrm{T}}$, where L_K is the lower Cholesky factor of K, and $\tilde{L}_K \stackrel{\mathrm{def}}{=} K_{\mathrm{f}}^{\mathrm{T}}L_K^{-\mathrm{T}}$. This is more efficient and yet does not involve any inversion of K.

The computation of V_{f} at this fixed point requires $T \stackrel{\mathrm{def}}{=} K^{-1} - K^{-1}V^{\mathrm{fx}}K^{-1}$. This can be done with the above formula for V^{fx}:

$$T = K^{-1} - L_K^{-\mathrm{T}}\left(I + \tilde{L}_K^{\mathrm{T}}W_{\mathrm{f}}\tilde{L}_K\right)^{-1}L_K^{-1} = L_K^{-\mathrm{T}}\left(I - \left(I + \tilde{L}_K^{\mathrm{T}}W_{\mathrm{f}}\tilde{L}_K\right)^{-1}\right)L_K^{-1}.$$

Hence

$$V_{\mathrm{f}}^{\mathrm{fx}} = K_{\mathrm{ff}} - K_{\mathrm{f}}^{\mathrm{T}}TK_{\mathrm{f}} = K_{\mathrm{ff}} - \tilde{L}_K\left(I - \left(I + \tilde{L}_K^{\mathrm{T}}W_{\mathrm{f}}\tilde{L}_K\right)^{-1}\right)\tilde{L}_K^{\mathrm{T}}.$$

E.3 Initialization

Our variational lower bound (21) on the marginal likelihood is concave with respect to all the variational parameters, so the initialization of parameters does not affect the converged answer in theory. However, in practice, initialization is still important for two reasons. First, it can ensure that the matrices are better conditioned. Second, it can ensure that we start near to the converged answer, so that convergence is sooner.

For initialization, there are two cases to be considered. The easier case is during model learning when we can use the optimized variational parameters from the previous model to initialize the variational parameters of the current model. We shall omit details for this case. The more difficult case is when there is no previous model, usually when no model learning is involved or at the onset of model learning. In this section, we suggest a procedure for initialization in this case. The key idea behind our procedure is to locate the variational mean at the data and to use the same covariance at every input $\mathbf{x}_i$.

E.3.1 COVARIANCES

From Equation 20 for the analysis of the proof of Theorem 6, a parametrization of W_i that satisfies the two mentioned properties is $W_i \stackrel{\mathrm{def}}{=} M - M\mathbf{1}\mathbf{1}^{\mathrm{T}}M/\mathbf{1}^{\mathrm{T}}M\mathbf{1}$, where M is a C-by-C positive definite matrix. Although we have noted there that using a diagonal M is suboptimal, there is much appeal in such a setting for initialization because of the match with the likelihood terms. Hence we shall initialize with $W_i \stackrel{\mathrm{def}}{=} \gamma\left(I/C - \mathbf{1}^{\mathrm{T}}\mathbf{1}/C^2\right)$, for some $\gamma > 0$. The initial covariance V of the variational posterior can be computed using Woodbury's inversion lemma on the fixed point equation $(K^{-1} + W)^{-1}$, where W is the block diagonal matrix consisting or the W_is.

E.3.2 MEANS

Our initialization for the mean locate it at the data. To this end, let us recall a few invariances at the stationary point of the variational lower bound (21) on the marginal likelihood. For the ith datum,

$\mathbf{a}_i$ and W_i are the parameters for the variational posterior $r(\mathbf{f}_i|\mathbf{y}_i)$ defined in Lemma 2. For the case of non-sparse approximation, we have the invariances

$$\begin{aligned} \mathbf{y}-\mathbf{b} &= W(\mathbf{a}-\mathbf{m}), && \text{(From definition in Lemma 5)} \\ \mathbf{m} &= K(\mathbf{y}-\mathbf{b}), && \text{(Section 3.3)} \\ V &= (K^{-1}+W)^{-1}, && \text{(Section 3.4)} \end{aligned}$$

where $\mathbf{y}$ (resp. $\mathbf{b}$, $\mathbf{a}$, $\mathbf{m}$) is the stacking of the $\mathbf{y}_i$s (resp. $\mathbf{b}_i$s, $\mathbf{a}_i$s, $\mathbf{m}_i$s) for each datum. Rearranging for $\mathbf{m}$ gives

$$\mathbf{m} = (I+KW)^{-1}KW\mathbf{a} = (K^{-1}+W)^{-1}W\mathbf{a} = VW\mathbf{a}. \tag{76}$$

We initialize $\mathbf{m}$ through an appropriate value for $\mathbf{a}$. Since $\mathbf{a}_i$ is the mean of $r(\mathbf{f}_i|\mathbf{y}_i)$, we choose to set $\mathbf{a}_i = \gamma(\mathbf{y}_i + (\mathbf{y}_i - \mathbf{1})/(C-1))$, for some fixed parameter γ. For example, if $\mathbf{x}_i$ is in the first class, then $\mathbf{a}_i = (\gamma, -\gamma/(C-1), \ldots, -\gamma/(C-1))^{\mathrm{T}}$. This locates the mean of $r(\mathbf{f}_i|\mathbf{y}_i)$ to be positive for the class given by the data and uniformly negative otherwise. Let $\boldsymbol{\alpha} \stackrel{\text{def}}{=} K^{-1}\mathbf{m}$. The initialization (76) satisfies the sum-to-zero property:

$$\mathbf{1}^{\mathrm{T}}\boldsymbol{\alpha} = \mathbf{1}^{\mathrm{T}}K^{-1}VW\mathbf{a} = \mathbf{1}^{\mathrm{T}}(I+WK)^{-1}W\mathbf{a} = \mathbf{1}^{\mathrm{T}}W(I+KW)^{-1}\mathbf{a} = \mathbf{0},$$

where the third equality applies Searle's Identity, and the last equality is because $\mathbf{1}$ is in the null-space of W. With $\gamma = 1$, the setting for $\mathbf{a}_i$ is the minimizer of the loss function in a multi-class SVM under the sum-to-zero constraint (Lee et al., 2004, Lemma 1).

Similarly, for sparse approximation, we have

$$\begin{aligned} \mathbf{y}-\mathbf{b} &= W_{\mathrm{f}}(\mathbf{a}-\mathbf{m}_{\mathrm{f}}), && \text{(From definition in Lemma 5)} \\ \mathbf{m}_{\mathrm{f}} &= K_{\mathrm{f}}^{\mathrm{T}}K^{-1}\mathbf{m}, && \text{(Section 4, Equation 31)} \\ \mathbf{m} &= K_{\mathrm{f}}(\mathbf{y}-\mathbf{b}), && \text{(Section 4.2)} \\ V &= (K^{-1}+K^{-1}K_{\mathrm{f}}W_{\mathrm{f}}K_{\mathrm{f}}^{\mathrm{T}}K^{-1})^{-1}. && \text{(Section 4.2)} \end{aligned}$$

Rearranging for $\mathbf{m}$ gives

$$\mathbf{m} + K_{\mathrm{f}}W_{\mathrm{f}}K_{\mathrm{f}}^{\mathrm{T}}K^{-1}\mathbf{m} = K_{\mathrm{f}}W_{\mathrm{f}}\mathbf{a} \quad \Longleftrightarrow \quad KV^{-1}\mathbf{m} = K_{\mathrm{f}}W_{\mathrm{f}}\mathbf{a} \quad \Longleftrightarrow \quad \mathbf{m} = VK^{-1}K_{\mathrm{f}}W_{\mathrm{f}}\mathbf{a}.$$

Initialization of $\mathbf{a}$ is done as in the non-sparse case.

Appendix F. Importance Sampling

In this section, we describe how various quantities of interest can be computed using importance sampling. Let $p(\mathbf{f}|\mathbf{y})$ be the exact posterior of the latent function values at the observed data. This is obtained from Bayes' rule $p(\mathbf{f}|\mathbf{y}) = p(\mathbf{y}|\mathbf{f})p(\mathbf{f})/p(\mathbf{y})$, where $p(\mathbf{y})$ is the marginal likelihood, which is intractable to compute exactly. Let $p_s(\mathbf{f})$ be a proposal distribution. Our choice of $p_s(\mathbf{f})$ is the multivariate-t distribution (Kotz and Nadarajah, 2004) with four degrees of freedom, centered at the that mean of the optimized variational approximation to $p(\mathbf{f}|\mathbf{y})$ and with covariance twice the covariance of the prior $p(\mathbf{f})$; that is

$$p_s(\mathbf{f}) = \frac{\Gamma((\nu+p)/2)}{((\pi\nu)^{p/2}\Gamma(\nu/2)|K|^{1/2}}\left[1+\frac{1}{\nu}(\mathbf{f}-\mathbf{m}^*)^{\mathrm{T}}K^{-1}(\mathbf{f}-\mathbf{m}^*)\right]^{(\nu+p)/2},$$

where $\nu = 4$, $p = nC$ is the dimension of $\mathbf{f}$, K is the prior covariance of $\mathbf{f}$, and $\mathbf{m}^*$ is the mean of the optimized variational posterior. This choice of proposal ensures that $p(\mathbf{f}) \leq c\, p_s(\mathbf{f})$ for all $\mathbf{f}$ for some finite constant $c > 0$, which is a desideratum for importance samplers. It also locates the proposal at the estimated mean of the posterior.

Let $\mathbf{f}^{(s)}$ be a sample from the proposal, indexed by s over n_s samples. Its unnormalized weight $w^{(s)}$ is

$$w^{(s)} \stackrel{\text{def}}{=} \frac{p(\mathbf{y})p(\mathbf{f}^{(s)}|\mathbf{y})}{p_s(\mathbf{f}^{(s)})} = \frac{p(\mathbf{y}|\mathbf{f}^{(s)})p(\mathbf{f}^{(s)})}{p_s(\mathbf{f}^{(s)})},$$

which can be computed exactly for the multinomial logistic likelihood. A Monte Carlo estimate of $p(\mathbf{y})$ is $\hat{p}(\mathbf{y}) \stackrel{\text{def}}{=} \sum_s w^{(s)}/n_s$, which is the sample mean of the $w^{(s)}$s, because

$$p(\mathbf{y}) \stackrel{\text{def}}{=} \int p(\mathbf{y}|\mathbf{f})p(\mathbf{f})\mathrm{d}\mathbf{f} = \int \frac{p(\mathbf{y}|\mathbf{f})p(\mathbf{f})}{p_s(\mathbf{f})} p_s(\mathbf{f})\mathrm{d}\mathbf{f} \approx \frac{1}{n_s}\sum_s w^{(s)}.$$

The strong law of large numbers says that $\hat{p}(\mathbf{y})$ converges to $p(\mathbf{y})$ almost surely as n_s approaches infinity (Geweke, 2005, Theorem 4.2.2). The rate of convergence is given by the Lindeberg-Lévy central limit theorem (Geweke, 2005, Theorem 4.2.2)

$$\sqrt{n_s}\,(p(\mathbf{y}) - \hat{p}(\mathbf{y})) \xrightarrow{\text{d}} \mathcal{N}(0, \sigma^2),$$

where σ^2 is the true variance of unnormalized weights. This variance exists for our choice of the proposal distribution because $p(\mathbf{f}) \leq c\, p_s(\mathbf{f})$ and the likelihood is bounded. This variance can be estimated from the samples $w^{(s)}$s. We use this convergence in distribution to compute a high probability upper bound to $p(\mathbf{y})$ based on the samples. Since, the weights and $p(\mathbf{y})$ are positive, one might be concerned that skewness has not been factored into the approximation. Then, one might consider using the χ^2 approximation (Hall, 1983). However, our calculations have shown this to have negligible effect on the upper bound estimate because we have used $n_s = 100,000$ samples.

F.1 Prediction

The normalized weight $\tilde{w}^{(s)}$ of $\mathbf{f}^{(s)}$ is estimated with

$$\tilde{w}^{(s)} \stackrel{\text{def}}{=} \frac{1}{n_s}\frac{p(\mathbf{f}^{(s)}|\mathbf{y})}{p_s(\mathbf{f}^{(s)})} = \frac{1}{n_s}\frac{w^{(s)}}{p(\mathbf{y})} \approx \frac{w^{(s)}}{\sum_{s'} w^{(s')}}.$$

For prediction at $\mathbf{x}_*$, the exact joint posterior of $(\mathbf{f}, \mathbf{f}_*)$ is $p(\mathbf{f}, \mathbf{f}_*|\mathbf{y}) = p(\mathbf{f}|\mathbf{y})p(\mathbf{f}_*|\mathbf{f})$. For the proposal distribution, we use $p_s(\mathbf{f}, \mathbf{f}_*) = p_s(\mathbf{f})p(\mathbf{f}_*|\mathbf{f})$, and a draw from the proposal follows this generative model. The normalized weight of sample $(\mathbf{f}, \mathbf{f}_*)^{(s)} \stackrel{\text{def}}{=} (\mathbf{f}^{(s)}, \mathbf{f}_*^{(s)})$ is

$$\tilde{w}_*^{(s)} \stackrel{\text{def}}{=} \frac{1}{n_s}\frac{p(\mathbf{f}^{(s)}|\mathbf{y})p(\mathbf{f}_*^{(s)}|\mathbf{f}^{(s)})}{p_s(\mathbf{f}^{(s)})p(\mathbf{f}_*^{(s)}|\mathbf{f}^{(s)})} = \tilde{w}^{(s)}.$$

The predictive probability is

$$p(\mathbf{y}_*|\mathbf{y}) = \int p(\mathbf{y}_*|\mathbf{f}_*)\, p(\mathbf{f}, \mathbf{f}_*|\mathbf{y})\, \mathrm{d}\mathbf{f}\mathrm{d}\mathbf{f}_* = \int p(\mathbf{y}_*|\mathbf{f}_*) \frac{p(\mathbf{f}, \mathbf{f}_*|\mathbf{y})}{p_s(\mathbf{f}, \mathbf{f}_*)} p_s(\mathbf{f}, \mathbf{f}_*)\, \mathrm{d}\mathbf{f}\mathrm{d}\mathbf{f}_* \approx \sum_s \tilde{w}_*^{(s)} p(\mathbf{y}_*|\mathbf{f}_*^{(s)}).$$

For the multinomial logistic likelihood, $p(\mathbf{y}_i|\mathbf{f}_i)$ and $p(\mathbf{y}_*|\mathbf{f}_*)$ can be computed readily. For the multinomial probit likelihood, we use the sampling approach (Girolami and Rogers, 2006) with twenty samples, which is sufficient when n_s is large.

References

T. Ando. Concavity of certain maps on positive definite matrices and applications to Hadamard products. *Linear Algebra and Its Applications*, 26:203–241, 1979.

D. Angluin and P. Laird. Learning from noisy examples. *Machine Learning*, 2:343–370, 1988.

A. Banerjee. On Bayesian bounds. In *International Conference on Machine Learning*, 2006.

E. V. Bonilla, K. M. A. Chai, and C. K. I. Williams. Multi-task Gaussian process prediction. In J. Platt, D. Koller, Y. Singer, and S. Roweis, editors, *Advances in Neural Information Processing Systems*, volume 20, pages 153–160. Curran Associates, Inc., 2008.

E. J. Bredensteiner and K. P. Bennett. Multicategory classification by support vector machines. *Computational Optimization Applications*, 12:53–79, January 1999.

J. S. Bridle. Probabilistic interpretation of feedforward classification network outputs with relationships to statistical pattern recognition. In *Neuro-computing: Algorithms, Architectures and Applications*, NATO ASI Series in Systems and Computer Science. Springer, 1989.

E. Challis and D. Barber. Concave Gaussian variational approximations for inference in large-scale Bayesian linear models. In D. Dunson and M. Dudk, editors, *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, volume 15 of *JMLR: Workshop and Conference Proceedings Series*, 2011.

K. Crammer and Y. Singer. On the algorithmic implementation of multiclass kernel-based vector machines. *Journal of Machine Learning Research*, 2:265–292, 2001.

L. Csató and M. Opper. Sparse online Gaussian processes. *Neural Computation*, 14:641–668, 2002.

A. Frank and A. Asuncion. UCI machine learning repository, 2010. URL `http://archive.ics.uci.edu/ml`.

J. Geweke. *Contemporary Bayesian Econometrics and Statistics*. Wiley Series in Probability and Statistics. John Wiley & Sons, Inc., New Jersey, 2005.

J. Geweke, M. P. Keane, and D. Runkle. Alternative computational approaches to inference in the multinomial probit model. *The Review of Economics and Statistics*, 76(4):609–32, 1994.

Z. Ghahramani and M. J. Beal. Graphical models and variational methods. In D. Saad and M. Opper, editors, *Advanced Mean Field methods — Theory and Practice*. MIT Press, 2000a.

Z. Ghahramani and M. J. Beal. Variational inference for Bayesian mixtures of factor analysers. In S. A. Solla, T. K. Leen, and K.-R. Müller, editors, *Advances in Neural Information Processing Systems*, volume 12, pages 449–455. MIT Press, Cambridge, MA, 2000b.

M. N. Gibbs. *Bayesian Gaussian Processes for Regression and Classification*. PhD thesis, University of Cambridge, Department of Physics, 1997.

M. Girolami and S. Rogers. Variational Bayesian multinomial probit regression with Gaussian process priors. *Neural Computation*, 18(8):1790–1817, 2006.

Y. Guermeur. Combining discriminant models with new multi-class SVMs. *Pattern Analysis and Applications*, 5(2):168–179, 2002.

Y. Guermeur and E. Monfrini. A quadratic loss multi-class SVM for which a radius-margin bound applies. *Informatica*, 22(1):73–96, 2011.

P. Hall. Chi squared approximations to the distribution of a sum of independent random variables. *The Annals of Probability*, 11(4):1028–1036, 1983.

R. Henao and O. Winther. PASS-GP: Predictive active set selection for Gaussian processes. In *IEEE International Workshop on Machine Learning for Signal Processing*, pages 148–153, 2010.

R. A. Horn and C. R. Johnson. *Matrix Analysis*. Cambridge University Press, 1985.

C.-W. Hsu and C.-J. Lin. A comparison of methods for multiclass support vector machines. *IEEE Transactions on Neural Networks*, 13(2):415–425, 2002.

H.-C. Kim and Z. Ghahramani. Bayesian Gaussian process classification with the EM-EP algorithm. *IEEE Transactions on Pattern Analysis Machine Intelligence*, 28:1948–1959, 2006.

D. A. Knowles and T. P. Minka. Non-conjugate variational message passing for multinomial and binary regression. In J. Shawe-Taylor, R. Zemel, P. Bartlett, F. Pereira, and K. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 24, pages 1701–1709. 2011.

S. J. Koopman, N. Shephard, and D. Creal. Testing the assumptions behind importance sampling. *Journal of Econometrics*, 149(1):2–11, 2009.

S. Kotz and S. Nadarajah. *Multivariate t Distributions and Their Applications*. Cambridge University Press, Cambridge, UK, 2004.

F. Lauer and Y. Guermeur. MSVMpack: a multi-class support vector machine package. *Journal of Machine Learning Research*, 12:2269–2272, 2011. `http://www.loria.fr/~lauer/MSVMpack`.

N. D. Lawrence, M. Seeger, and R. Herbrich. Fast sparse Gaussian process methods: The informative vector machine. In *Advances in Neural Information Processing Systems*, volume 15, pages 609–616, Cambridge, MA, 2003. MIT Press.

M. Lázaro-Gredilla and A. Figueiras-Vidal. Inter-domain Gaussian processes for sparse inference using inducing features. In Y. Bengio, D. Schuurmans, J. Lafferty, C. K. I. Williams, and A. Culotta, editors, *Advances in Neural Information Processing Systems*, volume 22, pages 1087–1095. Curran Associates, Inc., 2009.

Y. Lee, Y. Lin, and G. Wahba. Multicategory support vector machines. *Journal of the American Statistical Association*, 99(465):67–81, 2004.

D. McFadden. Conditional logit analysis of qualitative choice behavior. In P. Zarembka, editor, *Frontiers in Econometrics*, pages 105–142. Academic Press, 1974.

T. P. Minka. Beyond Newton's method, 2002. URL `http://research.microsoft.com/en-us/um/people/minka/papers/minka-newton.pdf`.

R. M. Neal. *Bayesian Learning for Neural Networks*. Springer-Verlag New York, 1996.

R. M. Neal. Regression and classification using Gaussian process priors. In A. P. D. J. M. Bernardo, J. O. Berger and A. F. M. Smith, editors, *Bayesian Statistics*, volume 6, pages 475–501. Oxford University Press, 1998.

R. M. Neal. Annealed importance sampling. *Statistics and Computing*, 11:125–139, 2001.

H. Nickisch and C. E. Rasmussen. Approximations for binary Gaussian process classification. *Journal of Machine Learning Research*, 9:2035–2078, 2008.

J. E. Potter. Matrix quadratic solutions. *SIAM Journal on Applied Mathematics*, 14(3):496–501, May 1966.

S. Puntanen and G. P. H. Styan. Historical introduction: Issai Schur and the early development of the Schur complement. In F. Zhang, editor, *The Schur Complement and Its Applications*, Numerical Methods and Algorithms, pages 1–16. Springer, 2005.

J. Quiñonero-Candela, C. E. Rasmussen, and C. K. I. Williams. Approximation methods for Gaussian process regression. In L. Bottou, O. Chapelle, D. DeCoste, and J. Weston, editors, *Large Scale Kernel Machines*, pages 203–223. MIT Press, 2007.

C. E. Rasmussen and C. K. I. Williams. *Gaussian Processes for Machine Learning*. MIT Press, Cambridge, Massachusetts, 2006.

R. Rifkin and A. Klautau. In defense of one-vs-all classification. *Journal of Machine Learning Research*, 5:101–141, 2004.

M. Seeger. PAC-Bayesian generalisation error bounds for Gaussian process classification. *Journal of Machine Learning Research*, 3:233–269, 2002.

M. Seeger and M. I. Jordan. Sparse Gaussian process classification with multiple classes. Technical report, University of California at Berkeley, Department of Statistics, 2004.

E. Snelson and Z. Ghahramani. Sparse Gaussian processes using pseudo-inputs. In Y. Weiss, B. Schölkopf, and J. Platt, editors, *Advances in Neural Information Processing Systems*, volume 18, pages 1257–1264, Cambridge, MA, 2006. MIT Press.

M. Titsias. Variational learning of inducing variables in sparse Gaussian processes. In D. van Dyk and M. Welling, editors, *Proceedings of the Twelfth International Conference on Artificial Intelligence and Statistics*, volume 5 of *JMLR: Workshop and Conference Proceedings Series*, pages 567–574, 2009a.

M. Titsias. Variational model selection for sparse Gaussian process regression. Technical report, University of Manchester, School of Computer Science, 2009b. URL `http://www.cs.man.ac.uk/~mtitsias/papers/sparseGPv2.pdf`.

V. N. Vapnik. *Statistical Learning Theory*. Wiley-Interscience, New York, 1998.

J. M. Ver Hoef and R. P. Barry. Constructing and fitting models for cokriging and multivariable spatial prediction. *Journal of Statistical Planning and Inference*, 69(2):275–294, 1998.

G. S. Watson. Spectral decomposition of the covariance matrix of a multinomial. *Journal of the Royal Statistical Society, Series B (Methodological)*, 58(1):289–291, 1996.

J. Weston and C. Watkins. Support vector machines for multi-class pattern recognition. In *Proceedings of the 7th European Symposium on Artificial Neural Network*, 1999.

C. K. I. Williams and M. Seeger. Using the Nyström method to speed up kernel machines. In T. K. Leen, T. G. Diettrich, and V. Tresp, editors, *Advances in Neural Information Processing Systems*, volume 13, pages 682–688, Cambridge, MA, 2001. MIT Press.

C. K. Williams and D. Barber. Bayesian classification with Gaussian processes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(12):1342–1351, 1998.

Journal of Machine Learning Research 13 (2012) 1809-1837 Submitted 12/11; Published 6/12

Entropy Search for Information-Efficient Global Optimization

Philipp Hennig PHILIPP.HENNIG@TUEBINGEN.MPG.DE
Christian J. Schuler CHRISTIAN.SCHULER@TUEBINGEN.MPG.DE
Department of Empirical Inference
Max Planck Institute for Intelligent Systems
Spemannstraße
72076 Tübingen, Germany

Editor: Neil Lawrence

Abstract

Contemporary global optimization algorithms are based on local measures of utility, rather than a probability measure over location and value of the optimum. They thus attempt to collect low function values, not to learn about the optimum. The reason for the absence of probabilistic global optimizers is that the corresponding inference problem is intractable in several ways. This paper develops desiderata for probabilistic optimization algorithms, then presents a concrete algorithm which addresses each of the computational intractabilities with a sequence of approximations and explicitly addresses the decision problem of maximizing information gain from each evaluation.

Keywords: optimization, probability, information, Gaussian processes, expectation propagation

1. Introduction

Optimization problems are ubiquitous in science, engineering, and economics. Over time the requirements of many separate fields have led to a heterogeneous set of settings and algorithms. Speaking very broadly, however, there are two distinct regimes for optimization. In the first one, relatively cheap function evaluations take place on a numerical machine and the goal is to find a "good" region of low or high function values. Noise tends to be small or negligible, and derivative observations are often available at low additional cost; but the parameter space may be very high-dimensional. This is the regime of *numerical, local* or *convex* optimization, often encountered as a sub-problem of machine learning algorithms. Popular algorithms for such settings include quasi-Newton methods (Broyden, 1965; Fletcher, 1970; Goldfarb, 1970; Shanno, 1970), the conjugate gradient method (Hestenes and Stiefel, 1952), and stochastic optimization and evolutionary search methods (for example Hansen and Ostermeier, 2001), to name only a few. Since these algorithms perform local search, constraints on the solution space are often a crucial part of the problem. Thorough introductions can be found in the textbooks by Nocedal and Wright (1999) and Boyd and Vandenberghe (2004). This paper will use algorithms from this domain, but it is not its primary subject.

In the second milieu, which this paper addresses, the function itself is not known and needs to be learned during the search for its *global* minimum within some measurable (usually: bounded) domain. Here, the parameter space is often relatively low-dimensional, but evaluating the function involves a monetarily or morally expensive physical process—building a prototype, drilling a borehole, killing a rodent, treating a patient. Noise is often a nontrivial issue, and derivative obser-

vations, while potentially available, cannot be expected in general. Algorithms for such applications also need to be tractable, but their most important desideratum is efficient use of data, rather than raw computational cost. This domain is often called *global optimization*, but is also closely associated with the field of *experimental design* and related to the concept of *exploration* in reinforcement learning. The learned model of the function is also known as a *response surface* in some communities. The two contributions of this paper are a probabilistic view on this field, and a concrete algorithm for such problems.

1.1 Problem Definition

We define the problem of *probabilistic global optimization*: Let $I \subset \mathbb{R}^D$ be some bounded domain of the real vector space. There is a function $f : I \to \mathbb{R}$, and our knowledge about f is described by a probability measure $p(f)$ over the space of functions $I \to \mathbb{R}$. This induces a measure

$$p_{\min}(x) \equiv p[x = \arg\min f(x)] = \int_{f:I\to\mathbb{R}} p(f) \prod_{\substack{\tilde{x}\in I \\ \tilde{x}\neq x}} \theta[f(\tilde{x}) - f(x)] \, \mathrm{d}f, \tag{1}$$

where θ is Heaviside's step function. The exact meaning of the "infinite product" over the entire domain I in this equation should be intuitively clear, but is defined properly in the Appendix. Note that the integral is over the infinite-dimensional space of functions. We assume we can evaluate the function[1] at any point $x \in I$ within some bounded domain I, obtaining function values $y(x)$ corrupted by noise, as described by a likelihood $p(y \,|\, f(x))$. Finally, let $L(x^*, x_{\min})$ be a loss function describing the cost of naming x^* as the result of optimization if the true minimum is at $x_{\min}$. This loss function induces a loss functional $\mathcal{L}(p_{\min})$ assigning utility to the uncertain knowledge about $x_{\min}$, as

$$\mathcal{L}(p_{\min}) = \int_I [\min_{x^*} L(x^*, x_{\min})] p_{\min}(x_{\min}) \, \mathrm{d}x_{\min}.$$

The goal of global optimization is to decrease the expected loss after H function evaluations at locations $x = \{x_1, \ldots, x_H\} \subset I$. The expected loss is

$$\langle \mathcal{L} \rangle_H = \int p(y \,|\, x) \mathcal{L}(p_{\min}(x \,|\, y, x)) \, \mathrm{d}y = \iint p(y \,|\, f(x)) p(f(x) \,|\, x) \mathcal{L}(p_{\min}(x \,|\, y, x)) \, \mathrm{d}y \, \mathrm{d}f, \tag{2}$$

where $\mathcal{L}(p_{\min}(x \,|\, y, x))$ should be understood as the cost assigned to the measure $p_{\min}(x)$ induced by the posterior belief over f after observations $y = \{y_1, \ldots, y_H\} \subset \mathbb{R}$ at the locations x.

The remainder of this paper will replace the symbolic objects in this general definition with concrete measures and models to construct an algorithm we call *Entropy Search*. But it is useful to pause at this point to contrast this definition with other concepts of optimization.

1.1.1 PROBABILISTIC OPTIMIZATION

The distinctive aspect of our definition of "optimization" is Equation (1), an explicit role for the function's extremum. Previous work did not consider the extremum so directly. In fact, many frameworks do not even use a measure over the function itself. An example of optimizers that only

1. We may further consider observations of linear operations on f. This includes derivative and integral observations of any order, if they exist. Section 2.8.1 addresses this point; it is unproblematic under our chosen prior, but clutters the notation, and is thus left out elsewhere in the paper.

implicitly encode assumptions about the function are genetic algorithms (Schmitt, 2004) and evolutionary search (Hansen and Ostermeier, 2001). If such formulations feature the global minimum $x_{\min}$ at all, then only in statements about the limit behavior of the algorithm after many evaluations. Not explicitly writing out the prior over the function space can have advantages: Probabilistic analyses tend to involve intractable integrals; a less explicit formulation thus allows to construct algorithms with interesting properties that would be challenging to derive from a probabilistic viewpoint. But non-probabilistic algorithms cannot make explicit statements about the location of the minimum. At best, they may be able to provide bounds.

Fundamentally, reasoning about optimization of functions on continuous domains *after finitely many evaluations*, like any other inference task on spaces without natural measures, is impossible without prior assumptions. For intuition, consider the following thought experiment: Let (x_0, y_0) be a finite, possibly empty, set of previously collected data. For simplicity, and without loss of generality, assume there was no measurement noise, so the true function actually passes through each data point. Say we want to suggest that the minimum of f may be at $x^* \in I$. To make this argument, we propose a number of functions that pass through (x_0, y_0) and are minimized at x^*. We may even suggest an uncountably infinite set of such functions. Whatever our proposal, a critic can always suggest another uncountable set of functions that also pass through the data, and are *not* minimized at x^*. To argue with this person, we need to reason about the relative size of our set versus their set. Assigning size to infinite sets amounts to the aforementioned normalized measure over admissible functions $p(f)$, and the consistent way to reason with such measures is probability theory (Kolmogorov, 1933; Cox, 1946). Of course, this amounts to imposing assumptions on f, but this is a fundamental epistemological limitation of inference, not a special aspect of optimization.

1.1.2 RELATIONSHIP TO THE BANDIT SETTING

There is a considerable amount of prior work on continuous bandit problems, also sometimes called "global optimization" (for example Kleinberg, 2005; Grünewälder et al., 2010; Srinivas et al., 2010). The bandit concept differs from the setting defined above, and bandit regret bounds do not apply here: Bandit algorithms seek to minimize *regret*, the sum over function values at evaluation points, while probabilistic optimizers seek to infer the minimum, no matter what the function values at evaluation points. An optimizer gets to evaluate H times, then has to make one single decision regarding $\mathcal{L}(p_{\min})$. Bandit players have to make H evaluations, such that the evaluations produce low values. This forces bandits to focus their evaluation policy on function value, rather than the loss at the horizon (see also Section 3.1). In probabilistic optimization, the only quantity that counts is the quality of the belief on $p_{\min}$ under $\mathcal{L}$, after H evaluations, not the sum of the function values returned during those H steps.

1.1.3 RELATIONSHIP TO HEURISTIC GAUSSIAN PROCESS OPTIMIZATION AND RESPONSE SURFACE OPTIMIZATION

There are also a number of works employing Gaussian process measures to construct heuristics for search, also known as "Gaussian process global optimization" (Jones et al., 1998; Lizotte, 2008; Osborne et al., 2009). As in our definition, these methods explicitly infer the function from observations, constructing a Gaussian process posterior. But they then evaluate at the location maximizing a heuristic $u[p(f(x))]$ that turns the *marginal* belief over $f(x)$ at x, which is a univariate Gaussian $p(f(x)) = \mathcal{N}[f(x); \mu(x), \sigma^2(x)]$, into an ad hoc utility for evaluation, designed to have high value at

locations close to the function's minimum. Two popular heuristics are the *probability of improvement* (Lizotte, 2008)

$$u_{\mathrm{PI}}(x) = p[f(x) < \eta] = \int_{-\infty}^{\eta} \mathcal{N}(f(x); \mu(x), \sigma(x)^2)\, \mathrm{d}f(x) = \Phi\left(\frac{\eta - \mu(x)}{\sigma(x)}\right),$$

and *expected improvement* (Jones et al., 1998)

$$u_{\mathrm{EI}}(x) = \mathsf{E}[\min\{0, (\eta - f(x))\}] = (\eta - \mu)\Phi\left(\frac{\eta - \mu(x)}{\sigma(x)}\right) + \sigma\phi\left(\frac{\eta - \mu(x)}{\sigma(x)}\right),$$

where $\Phi(z) = 1/2[1 + \mathrm{erf}(z/\sqrt{2})]$ is the standard Gaussian cumulative density function, $\phi(x) = \mathcal{N}(x; 0, 1)$ is the standard Gaussian probability density function, and η is a current "best guess" for a low function value, for example the lowest evaluation so far.

These two heuristics have different units of measure: probability of improvement is a probability, expected improvement has the units of f. Both utilities differ markedly from Equation (1), $p_{\min}$, which is a probability *measure* and as such a *global* quantity. See Figure 2 for a comparison of the three concepts on an example. The advantage of the heuristic approach is that it is computationally lightweight, because the utilities have analytic form. But local measures cannot capture general decision problems of the type described above. For example, these algorithms do not capture the effect of evaluations on knowledge: A small region of high density $p_{\min}(x)$ may be less interesting to explore than a broad region of lower density, because the expected *change* in knowledge from an evaluation in the broader region may be much larger, and may thus have much stronger effect on the loss. If the goal is to infer the location of the minimum (more generally: minimize loss at the horizon), the optimal strategy is to evaluate where we expect to *learn* most about the minimum (reduce loss toward the horizon), rather then where we think the minimum *is* (recall Section 1.1.2). The former is a nonlocal problem, because evaluations affect the belief, in general, everywhere. The latter is a local problem.

2. Entropy Search

The probable reason for the absence of global optimization algorithms from the literature is a number of intractabilities in any concrete realisation of the setting of Section 1.1. This section makes some choices and constructs a series of approximations, to arrive at a tangible algorithm, which we call *Entropy Search*. The derivations evolve along the following path.

choosing $p(f)$ We commit to a Gaussian process prior on f (Section 2.1). Limitations and implications of this choice are outlined, and possible extensions suggested, in Sections 2.8.1 and 2.8.3.

discretizing $p_{\min}$ We discretize the problem of calculating $p_{\min}$, to a finite set of representer points chosen from a non-uniform measure, which deals gracefully with the curse of dimensionality. Artifacts created by this discretization are studied in the tractable one-dimensional setting (Section 2.2).

approximating $p_{\min}$ We construct an efficient approximation to $p_{\min}$, which is required because Equation (1), even for finite-dimensional Gaussian measures, is not analytically tractable, (Section 2.3). We compare the approximation to the (asymptotically exact, but more expensive) Monte Carlo solution.

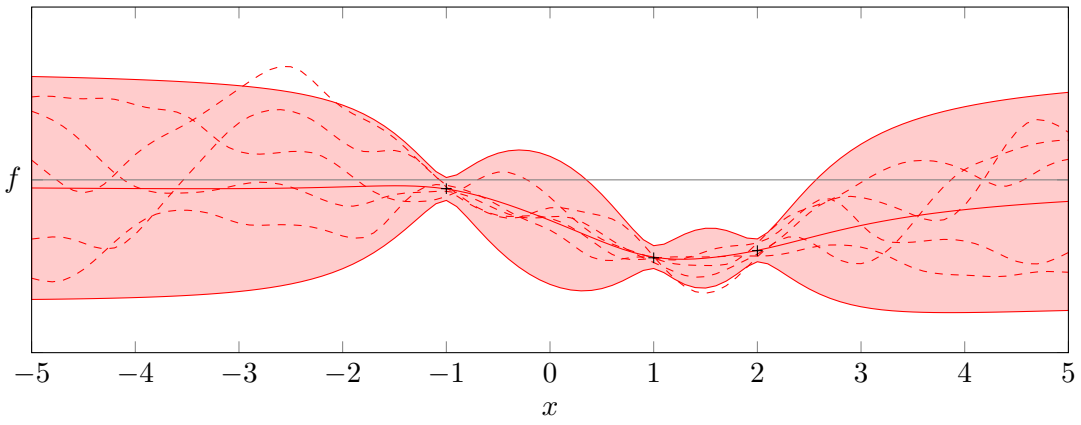


Figure 1: A Gaussian process measure (rational quadratic kernel), conditioned on three previous observations (black crosses). Mean function in solid red, marginal standard deviation at each location (two standard deviations) as light red tube. Five sampled functions from the current belief as dashed red lines. Arbitrary ordinate scale, zero in gray.

predicting change to $p_{\min}$ The Gaussian process measure affords a straightforward but rarely used analytic probabilistic formulation for the *change* of $p(f)$ as a function of the next evaluation point (Section 2.4).

choosing loss function We commit to relative *entropy* from a uniform distribution as the loss function, as this can be interpreted as a utility on gained *information* about the location of the minimum (Section 2.5).

predicting expected information gain From the predicted change, we construct a first-order expansion on $\langle \mathcal{L} \rangle$ from future evaluations and, again, compare to the asymptotically exact Monte Carlo answer (Section 2.6).

choosing greedily Faced with the exponential cost of the exact dynamic problem to the horizon H, we accept a greedy approach for the reduction of $\langle \mathcal{L} \rangle$ at every step. We illustrate the effect of this shortcut in an example setting (Section 2.7).

2.1 Gaussian Process Measure on f

The remainder of the paper commits to Gaussian process measures for $p(f)$. These are convenient for the task at hand due to their descriptive generality and their convenient analytic properties. Since this paper is aimed at readers from several communities, this section contains a very brief introduction to some relevant aspects of Gaussian processes; readers familiar with the subject can safely skip ahead. A thorough introduction can be found in a textbook of Rasmussen and Williams (2006). Some readers from other fields may find it helpful to know that more or less special cases of Gaussian process inference are elsewhere known under names like *Kriging* (Krige, 1951) and *Kolmogorov-Wiener prediction* (Wiener and Masani, 1957), but while these frameworks use essentially the same idea, the generality of their definitions varies, so restrictions of those frameworks

should not be assumed to carry over to Gaussian process inference as understood in machine learning.

A Gaussian process is an infinite-dimensional probability density, such that each linear finite-dimensional restriction is multivariate Gaussian. The infinite-dimensional space can be thought of as a space of functions, and the finite-dimensional restrictions as *values* of those functions at locations $\{x_i^*\}_{i=1,\ldots,N}$. Gaussian process beliefs are parametrized by a *mean function* $m : I \to \mathbb{R}$ and a *covariance function* $k : I \times I \to \mathbb{R}$. For our particular analysis, we restrict the domain I to finite, compact subsets of the real vector spaces $\mathbb{R}^D$. The covariance function, also known as the *kernel*, has to be positive definite, in the sense that any finite-dimensional matrix with elements $K_{ij} = k(x_i, x_j)$ has to be positive definite $\forall x_i, x_j \in I$. A number of such kernel functions are known in the literature, and different kernel functions induce different kinds of Gaussian process measures over the space of functions. Among the most widely used kernels for regression are the *squared exponential* kernel

$$k_{\mathrm{SE}}(x, x'; S, s) = s^2 \exp\left[-\frac{1}{2}(x - x')^\intercal S^{-1}(x - x')\right],$$

which induces a measure that puts nonzero mass on only smooth functions of *characteristic length-scale* S and *signal variance* s^2 (MacKay, 1998b), and the *rational quadratic* kernel (Matérn, 1960; Rasmussen and Williams, 2006)

$$k_{\mathrm{RQ}}(x, x'; S, s, \alpha) = s^2 \left(1 + \frac{1}{2\alpha}(x - x')^\intercal S^{-1}(x - x')\right)^{-\alpha},$$

which induces a belief over smooth functions whose characteristic length scales are a scale mixture over a distribution of width $1/\alpha$ and location S. Other kernels can be used to induce beliefs over non-smooth functions (Matérn, 1960), and even over non-continuous functions (Uhlenbeck and Ornstein, 1930). Experiments in this paper use the two kernels defined above, but the results apply to all kernels inducing beliefs over *continuous* functions. While there is a straightforward relationship between kernel continuity and the mean square continuity of the induced *process*, the relationship between the kernel function and the continuity of each *sample* is considerably more involved (Adler, 1981, §3). Regularity of the kernel also plays a nontrivial role in the question whether the distribution of infima of samples from the process is well-defined at all (Adler, 1990). In this work, we side-step this issue by assuming that the chosen kernel is sufficiently regular to induce a well-defined belief $p_{\min}$ as defined by Equation (8).

Kernels form a semiring: products and sums of kernels are kernels. These operations can be used to generalize the induced beliefs over the function space (Section 2.8.3). Without loss of generality, the mean function is often set to $m \equiv 0$ in theoretical analyses, and this paper will keep with this tradition, except for Section 2.8.3. Where m is nonzero, its effect is a straightforward off-set $p(f(x)) \to p(f(x) - m(x))$.

For the purpose of regression, the most important aspect of Gaussian process priors is that they are conjugate to the likelihood from finitely many observations $(X, Y) = \{x_i, y_i\}_{i=1,\ldots,N}$ of the form $y_i(x_i) = f(x_i) + \xi$ with Gaussian noise $\xi \sim \mathcal{N}(0, \sigma^2)$. The posterior is a Gaussian process with mean and covariance functions

$$\mu(x^*) = k_{x^*,X}[K_{X,X} + \sigma^2 I]^{-1} y \quad ; \quad \Sigma(x^*, x_*) = k_{x^*,x_*} - k_{x^*,X}[K_{X,X} + \sigma^2 I]^{-1} k_{X,x_*}, \tag{3}$$

where $K_{X,X}$ is the kernel Gram matrix $K_{X,X}^{(i,j)} = k(x_i, x_j)$, and other objects of the form $k_{a,b}$ are also matrices with elements $k_{a,b}^{(i,j)} = k(a_i, b_j)$. Finally, for what follows it is important to know that

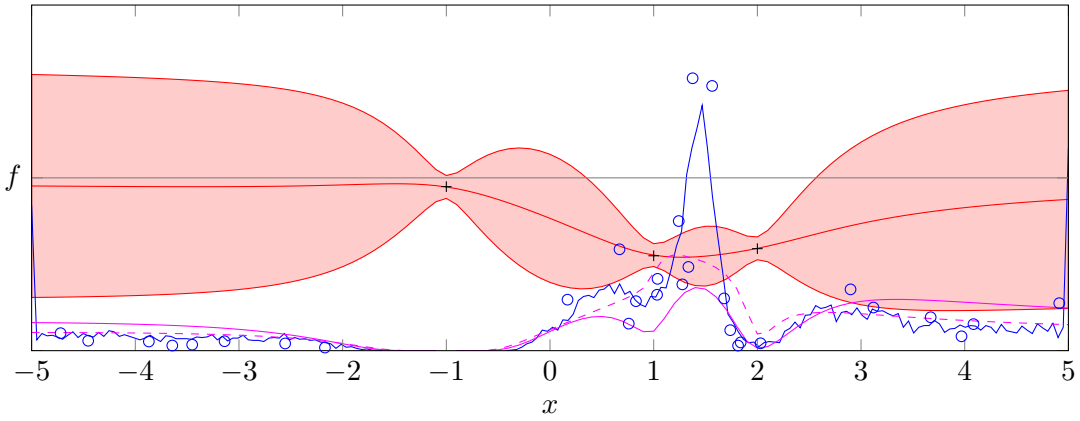


Figure 2: $p_{\min}$ induced by $p(f)$ from Figure 1. $p(f)$ repeated for reference. Blue solid line: Asymptotically exact representation of $p_{\min}$ gained from exact sampling of functions on a regular grid (artifacts due to finite sample size). For comparison, the plot also shows the local utilities *probability of improvement* (dashed magenta) and *expected improvement* (solid magenta) often used for Gaussian process global optimization. Blue circles: Approximate representation on representer points, sampled from probability of improvement measure. Stochastic error on sampled values, due to only asymptotically correct assignment of mass to samples, and varying density of points, focusing on relevant areas of $p_{\min}$. This plot uses arbitrary scales for each object: The two heuristics have different units of measure, differing from that of $p_{\min}$. Notice the interesting features of $p_{\min}$ at the boundaries of the domain: The prior belief encodes that f is smooth, and puts finite probability mass on the hypothesis that f has negative (positive) derivative at the right (left) boundary of the domain. With nonzero probability, the minimum thus lies exactly on the boundary of the domain, rather than within a Taylor radius of it.

it is straightforward to sample "functions" (point-sets of arbitrary size from I) from a Gaussian process. To sample the value of a particular sample at the M locations X^*, evaluate mean and variance function as a function of any previously collected data points, using Equation (3), draw a vector $\zeta \sim \prod^M \mathcal{N}(0,1)$ of M random numbers i.i.d. from a standard one-dimensional Gaussian distribution, then evaluate

$$\tilde{f}(X^*) = \mu(X^*) + \mathsf{C}[\Sigma(X^*, X^*)]^\intercal \zeta,$$

where the operator C denotes the Cholesky decomposition (Benoit, 1924).

2.2 Discrete Representations for Continuous Distributions

Having established a probability measure $p(f)$ on the function, we turn to constructing the belief $p_{\min}(x)$ over its minimum. Inspecting Equation (1), it becomes apparent that it is challenging in two ways: First, because it is an integral over an infinite-dimensional space, and second, because even on a finite-dimensional space it may be a hard integral for a particular $p(f)$. This section deals with the former issue, the following Section 2.3 with the latter.

It may seem daunting that $p_{\min}$ involves an infinite-dimensional integral. The crucial observation for a meaningful approximation in finite time is that regular functions can be represented meaningfully on finitely many points. If the stochastic process representing the belief over f is sufficiently regular, then Equation (1) can be approximated arbitrarily well with finitely many representer points. The discretization grid need not be regular—it may be sampled from any distribution which puts non-zero measure on every open neighborhood of I. This latter point is central to a graceful handling of the curse of dimensionality: The naïve approach of approximately solving Equation (1) on a regular grid, in a D-dimensional domain, would require $O(\exp(D))$ points to achieve any given resolution. This is obviously not efficient: Just like in other numerical quadrature problems, any given resolution can be achieved with fewer representer points if they are chosen irregularly, with higher resolution in regions of greater influence on the result of integration. We thus choose to *sample* representer points from a proposal measure u, using a Markov chain Monte Carlo sampler (our implementation uses shrinking rank slice sampling, by Thompson and Neal, 2010).

What is the effect of this stochastic discretization? A non-uniform quadrature measure $u(\tilde{x})$ for N representer locations $\{\tilde{x}_i\}_{i=1,\dots,N}$ leads to varying widths in the "steps" of the representing staircase function. As $N \to \infty$, the width of each step is approximately proportional to $(u(\tilde{x}_i)N)^{-1}$. Section 2.3 will construct a discretized $\hat{q}_{\min}(\tilde{x}_i)$ that is an approximation to the probability that $f_{\min}$ occurs within the step at $\tilde{x}_i$. So the approximate $\hat{p}_{\min}$ on this step is proportional to $\hat{q}_{\min}(\tilde{x}_i)u(\tilde{x}_i)$, and can be easily normalized numerically, to become an approximation to $p_{\min}$.

How should the measure u be chosen? Unfortunately, the result of the integration, being a density rather than a function, is itself a function of u, and the loss-function is also part of the problem. So it is nontrivial to construct an optimal quadrature measure. Intuitively, a good proposal measure for discretization points should put high resolution on regions of I where the shape of $p_{\min}$ has strong influence on the loss, and on its change. For our choice of loss function (Section 2.5), it is a good idea to choose u such that it puts high mass on regions of high value for $p_{\min}$. But for other functions, this need not always be the case.

We have experimented with a number of ad hoc choices for u, and found the aforementioned "expected improvement" and "probability of improvement" (Section 1.1.3) to lead to reasonably good performance. We use these functions for a similar reason as their original authors: Because they *tend* to have high value in regions where $p_{\min}$ is also large. To avoid confusion, however, note that we use these functions as unnormalized measures to *sample discretization points* for our *calculation* of $p_{\min}$, not as an approximation for $p_{\min}$ itself, as was done in previous work by other authors. Defects in these heuristics have weaker effect on our algorithm than in the cited works: in our case, if u is not a good proposal measure, we simply need more samples to construct a good representation of $p_{\min}$. In the limit of $N \to \infty$, all choices of u perform equally well, as long as they put nonzero mass on all open neighborhoods of the domain.

2.3 Approximating $p_{\min}$ with Expectation Propagation

The previous Section 2.2 provided a way to construct a non-uniform grid of N discrete locations $\tilde{x}_i$, $i = 1,\dots,N$. The restriction of the Gaussian process belief to these locations is a multivariate Gaussian density with mean $\tilde{\mu} \in \mathbb{R}^N$ and covariance $\tilde{\Sigma} \in \mathbb{R}^{N\times N}$. So Equation (1) reduces to a discrete probability *distribution* (as opposed to a density)

$$\hat{p}_{\min}(x_i) = \int_{f\in\mathbb{R}^N} \mathcal{N}(f;\tilde{\mu},\tilde{\Sigma}) \prod_{i\neq j}^{N} \theta(f(x_j) - f(x_i))\, \mathrm{d}f.$$

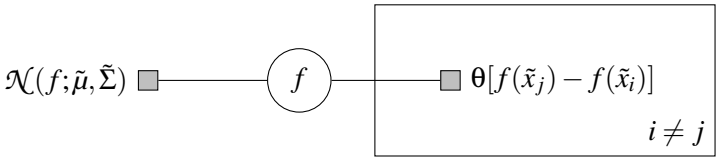


Figure 3: Graphical model providing motivation for EP approximation on $p_{\min}$. See text for details.

This is a multivariate Gaussian integral over a half-open, convex, piecewise linearly constrained integration region—a polyhedral cone. Unfortunately, such integrals are known to be intractable (Plackett, 1954; Lazard-Holly and Holly, 2003). However, it is possible to construct an effective approximation $\hat{q}_{\min}$ based on Expectation Propagation (EP) (Minka, 2001): Consider the belief $p(f(\tilde{x}))$ as a "prior message" on $f(\tilde{x})$, and each of the terms in the product as one factor providing another message. This gives the graphical model shown in Figure 3. Running EP on this graph provides an approximate Gaussian marginal, whose normalization constant $\hat{q}_{\min}(x_i)$, which EP also provides, approximates $p(f \,|\, x_{\min} = x_i)$. The EP algorithm itself is somewhat involved, and there are a number of algorithmic technicalities to take into account for this particular setting. We refer interested readers to recent work by Cunningham et al. (2011), which gives a detailed description of these aspects. The cited work also establishes that, while EP's approximations to Gaussian integrals are not always reliable, in this particular case, where there are as many constraints as dimensions to the problem, the approximation is generally of high quality (see Figure 4 for an example). An important advantage of the EP approximation over both numerical integration and Monte Carlo integration (see next Section) is that it allows analytic differentiation of $\hat{q}_{\min}$ with respect to the parameters $\tilde{\mu}$ and $\tilde{\Sigma}$ (Cunningham et al., 2011; Seeger, 2008). This fact will become important in Section 2.6.

The computational cost of this approximation is considerable: Each computation of $\hat{q}_{\min}(\tilde{x}_i)$, for a given i, involves N factor updates, which each have rank 1 and thus cost $O(N^2)$. So, overall, the cost of calculating $\hat{q}_{\min}(\tilde{x})$ is $O(N^4)$. This means N is effectively limited to well below $N = 1000$. Our implementation uses a default of $N = 50$, and can calculate next evaluation points in ~ 10 seconds. Once again, it is clear that this algorithm is not suitable for simple numerical optimization problems; but a few seconds are arguably an acceptable waiting time for physical optimization problems.

2.3.1 AN ALTERNATIVE: SAMPLING

An alternative to EP is Monte Carlo integration: sample S functions exactly from the Gaussian belief on $p(f)$, at cost $O(N^2)$ per sample, then find the minimum for each sample in $O(N)$ time. This technique was used to generate the asymptotically exact plots in Figures 2 and following. It has overall cost $O(SN^3)$, and can be implemented efficiently using Matrix-Matrix multiplications, so each evaluation of this algorithm is considerably faster than EP. It also has the advantage of asymptotic exactness. But, unfortunately, it provides no analytic derivatives, because of strong discontinuity in the step functions of Equation (1). So the choice is between a first-order expansion using EP (see Section 2.6) which is expensive, but provides a reusable, differentiable function, and repeated calls to a cheaper, asymptotically exact sampler. In our experiments, the former option appeared to be considerably faster, and of acceptable approximative quality. But for relatively

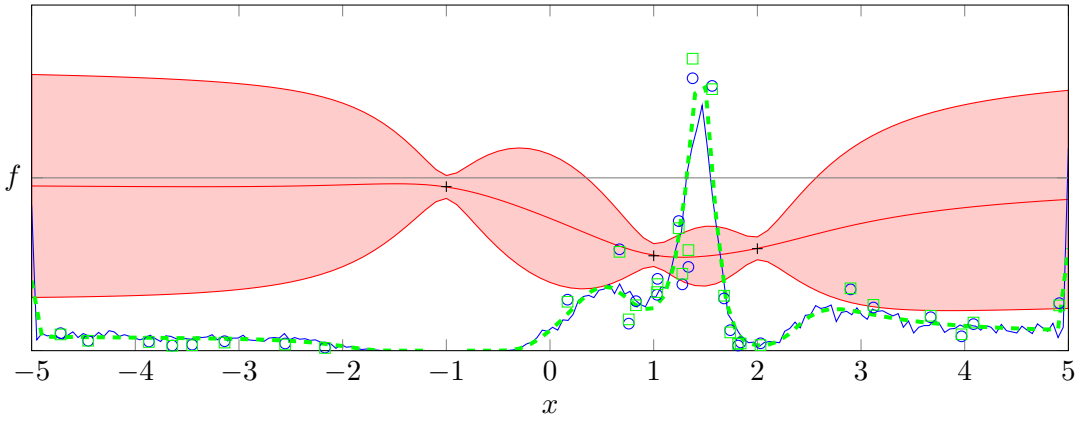


Figure 4: EP-approximation to $p_{\min}$ (dashed green). Other plots as in previous figures. EP achieves good agreement with the asymptotically exact Monte Carlo approximation to $p_{\min}$, including the point masses at the boundaries of the domain.

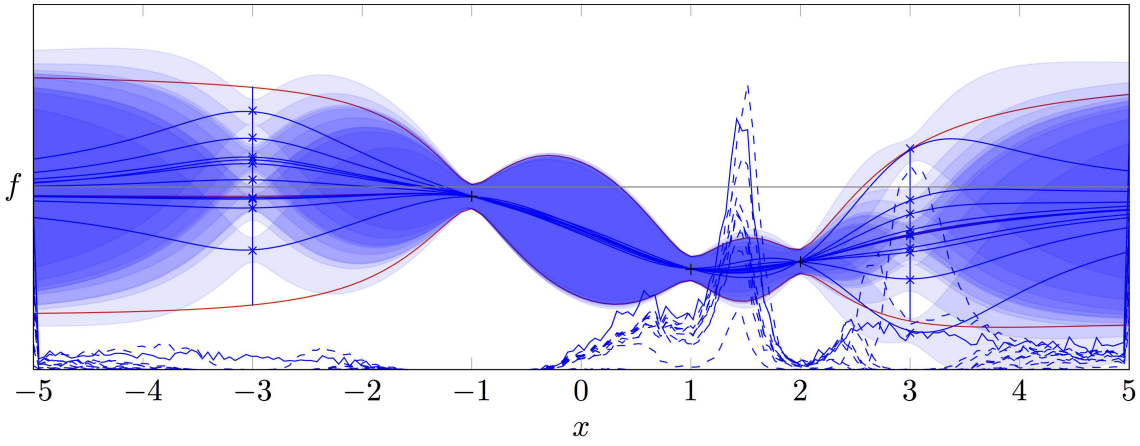


Figure 5: Innovation from two observations at $x = -3$ and $x = 3$. Current belief as red outline in background, from Figure 1. Samples from the belief over possible beliefs after observations at x in blue. For each sampled innovation, the plot also shows the induced innovated $p_{\min}$ (lower sampling resolution as previous plots). Innovations from several (here: two) observations can be sampled jointly.

high-dimensional optimization problems, where one would expect to require relatively large N for acceptable discretization, the sampling approach can be expected to scale better. [Note added in proof: It has been pointed out to us that a related approach, using sampling, was previously studied by Villemonteix et al. (2009).]

2.4 Predicting Innovation from Future Observations

As detailed in Equation (2), the optimal choice of the next H evaluations is such that the *expected* change in the loss $\langle \mathcal{L} \rangle_x$ is extremal, that is, it effects the biggest possible expected drop in loss. The loss is a function of $p_{\min}$, which in turn is a function of $p(f)$. So predicting change in loss requires predicting change in $p(f)$ as a function of the next evaluation points. It is another convenient aspect of Gaussian processes that they allow such predictions in analytic form (Hennig, 2011): Let previous observations at X_0 have yielded observations Y_0. Evaluating at locations X will give new observations Y, and the mean will be given by

$$\begin{aligned}\mu(x^*) &= [k_{x^*,X_0}, k_{x^*,X}] \begin{pmatrix} K_{X_0,X_0} & k_{X_0,X} \\ k_{X,X_0} & K_{X,X} \end{pmatrix}^{-1} \begin{pmatrix} Y_0 \\ Y \end{pmatrix} \\ &= k_{x^*,X_0} K_{X_0,X_0}^{-1} Y_0 + (k_{x^*,X} - k_{x^*,X_0} K_{X_0,X_0}^{-1} k_{X_0,X}) \times \\ &\qquad (k_{X,X} - k_{X,X_0} K_{X_0,X_0}^{-1} k_{X_0,X})^{-1} (Y - k_{X,X_0} K_{X_0,X_0}^{-1} Y_0) \\ &= \mu_0(x^*) + \Sigma_0(x^*,X)\Sigma_0^{-1}(X,X)(Y - \mu_0(X)),\end{aligned} \tag{4}$$

where $K_{a,b}^{(i,j)} = k(a_i,b_j) + \delta_{ij}\sigma^2$. The step from the first to the second line involves an application of the matrix inversion lemma, the last line uses the mean and covariance functions conditioned on the data set (X_0,Y_0) so far. Since Y is presumed to come from this very Gaussian process belief, we can write

$$Y = \mu(X) + \mathsf{C}[\Sigma(X,X)]^\intercal \Omega' + \sigma\omega = \mu(X) + \mathsf{C}[\Sigma(X,X) + \sigma^2 I_H]^\intercal \Omega \qquad \Omega, \Omega', \omega \sim \mathcal{N}(0, I_H),$$

and Equation (4) simplifies. An even simpler construction can be made for the covariance function. We find that mean and covariance function of the posterior after observations (X,Y) are mean and covariance function of the prior, incremented by the *innovations*

$$\begin{aligned}\Delta\mu_{X,\Omega}(x^*) &= \Sigma(x^*,X)\Sigma^{-1}(X,X)\,\mathsf{C}[\Sigma(X,X) + \sigma^2 I_H]\Omega \\ \Delta\Sigma_X(x^*,x_*) &= \Sigma(x^*,X)\Sigma^{-1}(X,X)\Sigma(X,x_*).\end{aligned}$$

The change to the mean function is stochastic, while the change to the covariance function is deterministic. Both innovations are functions both of X and of the evaluation points x^*. One use of this result is to sample $\langle \mathcal{L} \rangle_X$ by sampling innovations, then evaluating the innovated $p_{\min}$ for each innovation in an inner loop, as described in Section 2.3.1. An alternative, described in the next section, is to construct an analytic first order approximation to $\langle \mathcal{L} \rangle_X$ from the EP prediction constructed in Section 2.3. As mentioned above, the advantage of this latter option is that it provides an analytic function, with derivatives, which allows efficient numerical local optimization.

2.5 Information Gain—the Log Loss

To solve the decision problem of where to evaluate the function next in order to learn most about the location of the minimum, we need to say what it means to "learn". Thus, we require a loss functional that evaluates the information content of innovated beliefs $p_{\min}$. This is, of course, a core idea in information theory. The seminal paper by Shannon (1948) showed that the negative expectation of probability logarithms,

$$\mathsf{H}[p] = -\langle \log p \rangle_p = -\sum_i p_i \log p_i, \tag{5}$$

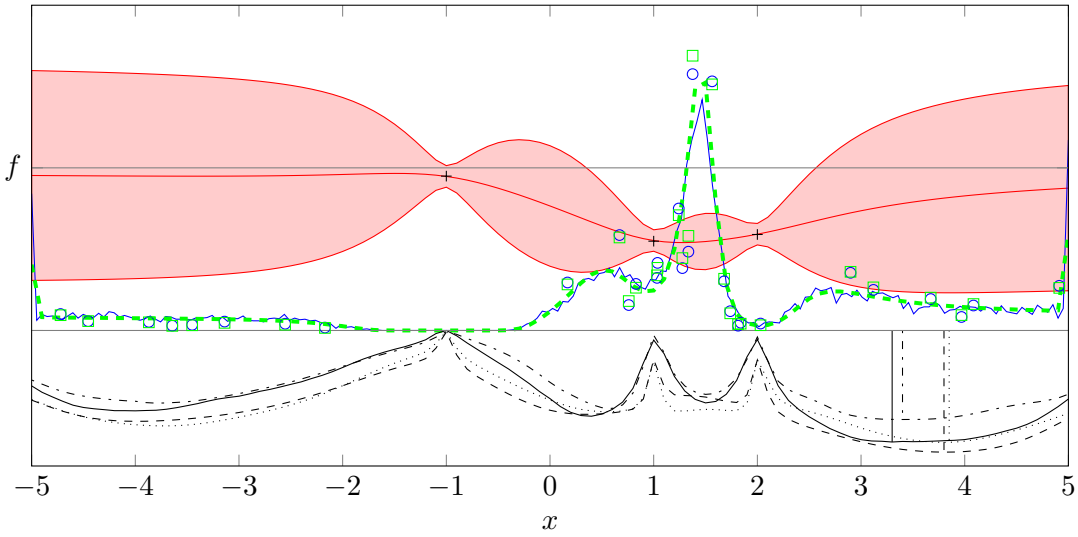


Figure 6: 1-step predicted loss improvement for the log loss (relative entropy). Upper part of plot as before, for reference. Monte Carlo prediction on regular grid as solid black line. Monte Carlo prediction from sampled irregular grid as dot-dashed black line. EP prediction on regular grid as black dashed line. EP prediction from samples as black dotted line. The minima of these functions, where the algorithm will evaluate next, are marked by vertical lines. While the predictions from the various approximations are not identical, they lead to similar next evaluation points. Note that these next evaluation points differ qualitatively from the choice of the GP optimization heuristics of Figure 2. Since each approximation is only tractable up a multiplicative constant, the scales of these plots are arbitrary, and only chosen to overlap for convenience.

known as entropy, has a number of properties that allow its interpretation as a measure of uncertainty represented by a probability distribution p. Its value can be be interpreted as the number of natural information units an optimal compression algorithm requires to encode a sample from the distribution, given knowledge of the distribution. However, it has since been pointed out repeatedly that this concept does not easily generalize to probability densities. A density $p(x)$ has a unit of measure $[x]^{-1}$, so its logarithm is not well-defined, and one cannot simply replace summation with integration in Equation (5). A functional that *is* well-defined on probability densities and preserves many of the information-content interpretations of entropy (Jaynes and Bretthorst, 2003) is *relative entropy*, also known as Kullback-Leibler (1951) divergence. We use its negative value as a loss function for information gain.

$$\mathcal{L}_{\mathrm{KL}}(p;b) = -\int p(x) \log \frac{p(x)}{b(x)} \, \mathrm{d}x.$$

As base measure b we choose the uniform measure $\mathbb{U}_I(x) = |I|^{-1}$ over I, which is well-defined because I is presumed to be bounded.[2] With this choice, the loss is maximized (at $\mathcal{L} = 0$) for a

2. Although uniform measures appeal as a natural representation of ignorance, they do encode an assumption about I being represented in a "natural" way. Under a nonlinear transformation of I, the distribution would not remain

uniform belief over the minimum, and diverges toward negative infinity if p approaches a Dirac point distribution. The resulting algorithm, Entropy Search, will thus choose evaluation points such that it expects to move away from the uniform base measure toward a Dirac distribution as quickly as possible.

The reader may wonder: What about the alternative idea of maximizing, at each evaluation, entropy relative to the *current* $p_{\min}$? This would only encourage the algorithm to attempt to change the current belief, but not necessarily in the right direction. For example, if the current belief puts very low mass on a certain region, an evaluation that has even a small chance of increasing $p_{\min}$ in this region could appear more favorable than an alternative evaluation predicted to have a large effect on regions where the current $p_{\min}$ has larger values. The point is not to just change $p_{\min}$, but to change it such that it moves away from the base measure.

Recall that we approximate the *density* $p(x)$ using a *distribution* $\hat{p}(x_i)$ on a finite set $\{x_i\}$ of representer points, which define steps of width proportional, up to stochastic error, to an unnormalized measure $\tilde{u}(x_i)$. In other words, we can approximate $p_{\min}(x)$ as

$$p_{\min}(x) \approx \frac{\hat{p}(x_i) N \tilde{u}(x_i)}{Z_u}; \qquad Z_u = \int \tilde{u}(x)\,\mathrm{d}x; \qquad x_i = \arg\min_{\{x_j\}} \|x - x_j\|.$$

We also note that after N samples, the unit element of measure has size, up to stochastic error, of $\Delta x_i \approx \frac{Z_u}{\tilde{u}(x_i)N}$. So we can approximately represent the loss

$$\begin{aligned}
\mathcal{L}_{\mathrm{KL}}(p_{\min}; b) &\approx -\textstyle\sum_i p_{\min}(x_i)\Delta x_i \log \frac{p_{\min}(x_i)}{b(x_i)} \\
&= -\textstyle\sum_i \hat{p}_{\min}(x_i) \log \frac{\hat{p}_{\min}(x_i) N \tilde{u}(x_i)}{Z_u b(x_i)} \\
&= -\textstyle\sum_i \hat{p}_{\min}(x_i) \log \frac{\hat{p}_{\min}(x_i)\tilde{u}(x_i)}{b(x_i)} + \log\left(\frac{Z_u}{N}\right) \textstyle\sum_i \hat{p}_{\min}(x_i) \\
&= \mathsf{H}[\hat{p}_{\min}] - \langle \log \tilde{u} \rangle_{\hat{p}_{\min}} + \langle \log b \rangle_{\hat{p}_{\min}} + \log Z_u - \log N,
\end{aligned}$$

which means we do not require the normalization constant Z_u for optimization of $\mathcal{L}_{\mathrm{KL}}$. For our uniform base measure, the third term in the last line is a constant, too; but other base measures would contribute nontrivially.

2.6 First-Order Approximation to $\langle \mathcal{L} \rangle$

Since EP provides analytic derivatives of $p_{\min}$ with respect to mean and covariance of the Gaussian measure over f, we can construct a first order expansion of the expected change in loss from evaluations. To do so, we consider, in turn, the effect of evaluations at X on the measure on f, the induced change in $p_{\min}$, and finally the change in $\mathcal{L}$. Since the change to the mean is Gaussian stochastic, Itō's (1951) Lemma applies. The following Equation uses the summation convention: double indices in products are summed over.

$$\begin{aligned}
\langle \Delta\mathcal{L} \rangle_X = \int \mathcal{L} \Bigg[p^0_{\min} &+ \frac{\partial p_{\min}}{\partial \Sigma(\tilde{x}_i, \tilde{x}_j)} \Delta\Sigma_X(\tilde{x}_i, \tilde{x}_j) + \frac{\partial^2 p_{\min}}{\partial \mu_i \partial \mu_j} \Delta\mu_{X,1}(\tilde{x}_i) \Delta\mu_{X,1}(\tilde{x}_j) \\
&+ \frac{\partial p_{\min}}{\partial \mu(\tilde{x}_i)} \Delta_{X,\Omega}\mu(\tilde{x}_i) + O((\Delta\mu)^2, (\Delta\Sigma)^2) \Bigg] \mathcal{N}(\Omega; 0, 1)\,\mathrm{d}\Omega - \mathcal{L}[p^0_{\min}]. \quad (6)
\end{aligned}$$

uniform. For example, uniform measures on the [0,1] simplex appear bell-shaped in the softmax basis (MacKay, 1998a). So, while b here does not represent prior knowledge on $x_{\min}$ per se, it does provide a unit of measure to information and as such is nontrivial.

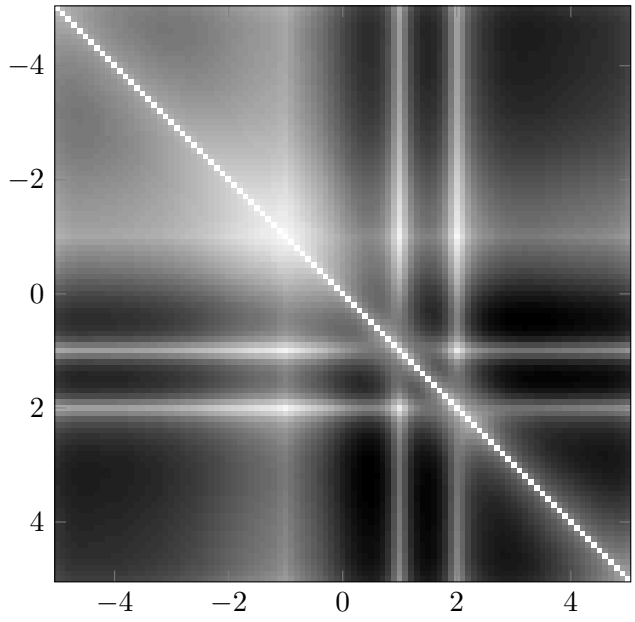


Figure 7: Expected drop in relative entropy (see Section 2.5) from two additional evaluations to the three old evaluations shown in previous plots. First new evaluation on abscissa, second new evaluation on ordinate, but due to the exchangeability of Gaussian process measures, the plot is symmetric. Diagonal elements excluded for numerical reasons. Blue regions are more beneficial than red ones. The relatively complicated structure of this plot illustrates the complexity of finding the optimal H-step evaluation locations.

The first line contains deterministic effects, the first term in the second line covers the stochastic aspect. Monte Carlo integration over the stochastic effects can be performed approximately using a small number of samples Ω. These samples should be drawn only once, at first calculation, to get a differentiable function $\langle \Delta\mathcal{L} \rangle_X$ that can be re-used in subsequent optimization steps.

The above formulation is agnostic with respect to the loss function. Hence, in principle, Entropy Search should be easy to generalize to different loss functions. But recall that the fidelity of the calculation of Equation (6) depends on the intermediate approximate steps, in particular the choice of discretization measure $\tilde{u}$. We have experimented with other loss functions and found it difficult to find a good measure $\tilde{u}$ providing good performance for many such loss functions. So this paper is limited to the specific choice of the relative entropy loss function. Generalization to other losses is future work.

2.7 Greedy Planning, and its Defects

The previous sections constructed a means to predict, approximately, the expected drop in loss from H new evaluations at locations $X = \{x_i\}_{i=1,\ldots,N}$. The remaining task is to optimize these locations. It may seem pointless to construct an optimization algorithm which itself contains an optimization problem, but note that this new optimization problem is quite different from the initial one. It is a numerical optimization problem, of the form described in Section 1: We can evaluate the utility

function numerically, without noise, with derivatives, and at hopefully relatively low cost compared to the physical process we are ultimately trying to optimize.

Nevertheless, one issue remains: Optimizing evaluations over the entire horizon H is a dynamic programming problem, which, in general, has cost exponential in H. However, this problem has a particular structure: Apart from the fact that evaluations drawn from Gaussian process measures are exchangeable, there is also other evidence that optimization problems are benign from the point of view of planning. For example, Srinivas et al. (2010) show that the information gain over the function values is submodular, so that greedy learning of the function comes close to optimal learning of the function. While is is not immediately clear whether this statement extends to our issue of learning about the function's minimum, it is obvious that the greedy choice of whatever evaluation location most reduces expected loss in the immediate next step is guaranteed to never be catastrophically wrong. In contrast to general planning, there are no "dead ends" in inference problems. At worst, a greedy algorithm may choose an evaluation point revealed as redundant by a later step. But thanks to the consistency of Bayesian inference in general, and Gaussian process priors in particular (van der Vaart and van Zanten, 2011), no decision can lead to an evaluation that somehow makes it impossible to learn the true function afterward. In our approximate algorithm, we thus adopt this greedy approach. It remains an open question for future research whether approximate planning techniques can be applied efficiently to improve performance in this planning problem.

2.8 Further Issues

This section digresses from the main line of thought to briefly touch upon some extensions and issues arising from the choices made in previous sections. For the most part, we point out well-known analytic properties and approximations that can be used to generalize the algorithm. Since they apply to Gaussian process regression rather than the optimizer itself, they will not play a role in the empirical evaluation of Section 3.

2.8.1 DERIVATIVE OBSERVATIONS

Gaussian process inference remains analytically tractable if instead of, or in addition to direct observations of f, we observe the result of any *linear* operator acting on f. This includes observations of the function's derivatives (Rasmussen and Williams, 2006, §9.4) and, with some caveats, to integral observations (Minka, 2000). The extension is pleasingly straightforward: The kernel defines covariances between function values. Covariances between the function and its derivatives are simply given by

$$\operatorname{cov}\left(\frac{\partial^n f(x)}{\prod_i \partial x_i}, \frac{\partial^m f(x')}{\prod_j \partial x'_j}\right) = \frac{\partial^{n+m} k(x,x')}{\prod_i \partial x_i \prod_j \partial x'_j},$$

so kernel evaluations simply have to be replaced with derivatives (or integrals) of the kernel where required. Obviously, this operation is only valid as long as the derivatives and integrals in question exist for the kernel in question. Hence, all results derived in previous sections for optimization from function evaluations can trivially be extended to optimization from function and derivative observations, or from only derivative observations.

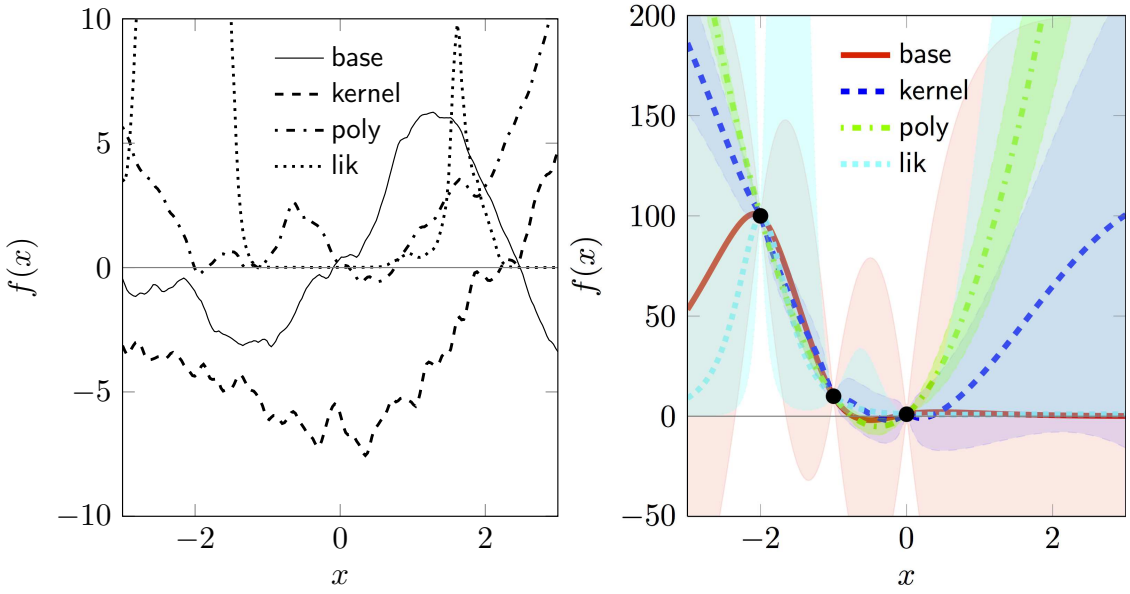


Figure 8: Generalizing GP regression. **Left:** Samples from different priors. **Right:** Posteriors (mean, two standard deviations) after observing three data points with negligible noise (kernel parameters differ between the two plots). base: standard GP regression with Matérn kernel. kernel: sum of two kernels (square exponential and rational quadratic) of different length scales and strengths. poly: polynomial (here: quadratic) mean function. lik: Non-Gaussian likelihood (here: logarithmic link function). The scales of both x and $f(x)$ are functions of kernel parameters, so the numerical values in this plot have relevance only relative to each other. Note the strong differences in both mean and covariance functions of the posteriors.

2.8.2 LEARNING HYPERPARAMETERS

Throughout this paper, we have assumed kernel and likelihood function to be given. In real applications, this will not usually be the case. In such situations, the hyperparameters defining these two functions, and if necessary a mean function, can be learned from the data, either by setting them to maximum likelihood values, or by full-scale Bayesian inference using Markov chain Monte Carlo methods. See Rasmussen and Williams (2006, §5) and Murray and Adams (2010) for details. In the latter case, the belief $p(f)$ over the function is a mixture of Gaussian processes. To still be able to use the algorithm derived so far, we approximate this belief with a single Gaussian process by calculating expected values of mean and covariance function.

Ideally, one would want to take account of this hierarchical learning process in the decision problem addressed by the optimizer. This adds another layer of computation complexity to the problem, and is outside of the scope of this paper. Here, we content ourselves with considering the uncertainty of the Gaussian process conditioned on a particular set of hyperparameters.

2.8.3 LIMITATIONS AND EXTENSIONS OF GAUSSIAN PROCESSES FOR OPTIMIZATION

Like any probability measure over functions, Gaussian process measures are not arbitrarily general. In particular, the most widely used kernels, including the two mentioned above, are *stationary*, meaning they only depend on the difference between locations, not their absolute values. Loosely speaking, the prior "looks the same everywhere". One may argue that many real optimization problems do not have this structure. For example, it may be known that the function tends to have larger functions values toward the boundaries of I or, more vaguely, that it is roughly "bowl-shaped". Fortunately, a number of extensions readily suggest themselves to address such issues (Figure 8).

Parametric Means As pointed out in Section 2.1, we are free to add any parametric general linear model as the mean function of the Gaussian process,

$$m(x) = \sum_i \phi_i(x) w_i.$$

Using Gaussian beliefs on the weights w_i of this model, this model may be learned at the same time as the Gaussian process itself (Rasmussen and Williams, 2006, §2.7). Polynomials such as the quadratic $\phi(x) = [x; xx^\intercal]$ are beguiling in this regard, but they create an explicit "origin" at the center of I, and induce strong long-range correlations between opposite ends of I. This seems pathological: In most settings, observing the function on one end of I should not tell us much about the value at the opposite end of I. But we may more generally choose any feature set for the linear model. For example, a set of radial basis functions $\phi_i(x) = \exp(\|x - c_i\|^2/\ell_i^2)$ around locations c_i at the rims of I can explain large function values in a region of width ℓ_i around such a feature, without having to predict large values at the center of I. This idea can be extended to a nonparametric version, described in the next point.

Composite Kernels Since kernels form a semiring, we may sum a kernel of large length scale and large signal variance and a kernel of short length scale and low signal variance. For example

$$k(x,x') = k_{\mathrm{SE}}(x,x';s_1,S_1) + k_{\mathrm{RQ}}(x,x',s_2,S_2,\alpha_2) \qquad s_1 \gg s_2; S_1^{ij} \gg S_2^{ij}\ \forall i,j$$

yields a kernel over functions that, within the bounded domain I, look like "rough troughs": global curvature paired with local stationary variations. A disadvantage of this prior is that it thinks "domes" just as likely as "bowls". An advantage is that it is a very flexible framework, and does not induce unwanted global correlations.

Nonlinear Likelihoods An altogether different effect can be achieved by a non-Gaussian, nonlinear likelihood function. For example, if f is known to be strictly positive, one may assume the noise model

$$p(y \mid g) = \mathcal{N}(y; \exp(g), \sigma^2); \quad f = \exp(g), \tag{7}$$

and learn g instead of f. Since the logarithm is a convex function, the minimum of the latent g is also a minium of f. Of course, this likelihood leads to a non-Gaussian posterior. To retain a tractable algorithm, approximate inference methods can be used to construct approximate Gaussian posteriors. In our example (labeled lik in Figure 8), we used a Laplace approximation: It is straightforward to show that Equation (7) implies

$$\left.\frac{\partial \log p(y \mid g)}{\partial g}\right|_{g=\hat{g}} \stackrel{!}{=} 0 \quad \Rightarrow \hat{g} = \log y \qquad \left.\frac{\partial^2 \log p(y \mid g)}{\partial^2 g}\right|_{g=\hat{g}} = \frac{y^2}{\sigma^2},$$

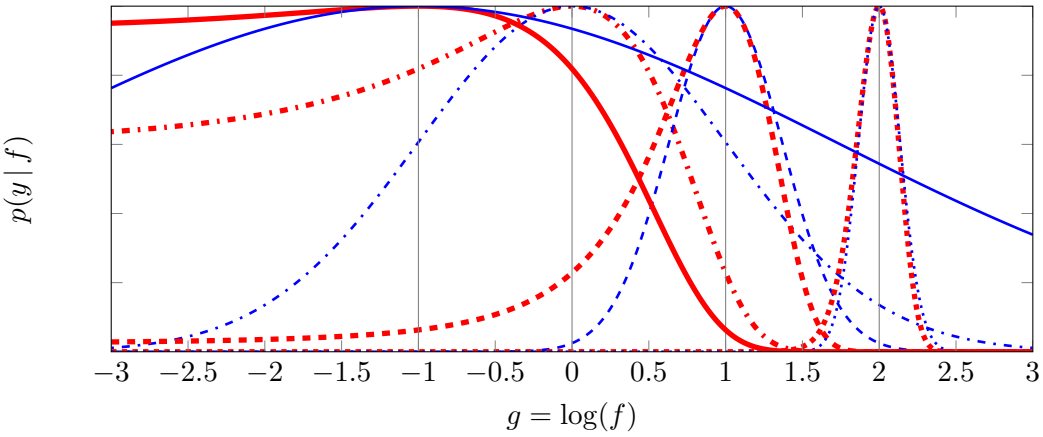


Figure 9: Laplace approximation for a logarithmic Gaussian likelihood. True likelihood in thick red, Gaussian approximation in thin blue, maximum likelihood solution marked in grey. Four log relative values $a = \log(y/\sigma)$ of sample y and noise σ (scaled in height for readability). $a = -1$ (solid); $a = 0$ (dash-dotted); $a = 1$ (dashed); $a = 2$ (dotted). The approximation is good for $a \gg 0$.

so a Laplace approximation amounts to a heteroscedastic noise model, in which an observation (y, σ^2) is incorporated into the Gaussian process as $(\log(y), (\sigma/y)^2)$. This approximation is valid if $\sigma \ll y$ (see Figure 9). For functions on logarithmic scales, however, finding minima smaller than the noise level, at logarithmic resolution, is a considerably harder problem anyway.

The right part of Figure 8 shows posteriors produced using the three approaches detailed above, and the base case of a single kernel with strong signal variance, when presented with the same three data points, with very low noise. The strong difference between the posteriors may be disappointing, but it is a fundamental aspect of inference: Different prior assumptions lead to different posteriors, and function space inference is impossible without priors. Each of the four beliefs shown in the Figure may be preferable over the others in particular situations. The polynomial mean describes functions that are almost parabolic. The exponential likelihood approximation is appropriate for functions with an intrinsic logarithmic scale. The sum kernel approach is pertinent for the search for local minima of globally stationary functions. Classic methods based on polynomial approximations are a lot more restrictive than any of the models described above.

Perhaps the most general option is to use additional prior information I giving $p(x_{\min} \mid I)$, independent of $p(f)$, to encode outside information about the location of the minimum. Unfortunately, this is intractable in general. But it may be approached through approximations. This option is outside of the scope of this paper, but will be the subject of future work.

2.9 Summary—the Entire Algorithm

Algorithm 1 shows pseudocode for Entropy Search. It takes as input the prior, described by the kernel k, and the likelihood $l = p(y \mid f(x))$, as well as the discretization measure u (which may itself

Algorithm 1 Entropy Search

procedure ENTROPYSEARCH($k, l = p(y \mid f(x)), u, H, (x,y)$)
  $\tilde{x} \sim u(x,y)$ ▷ discretize using measure u (Section 2.2)
  $[\mu, \Sigma, \Delta\mu_x, \Delta\Sigma_x] \leftarrow \mathrm{GP}(k, l, x, y)$ ▷ infer function, innovation, from GP prior (2.1)
  $[\hat{q}_{\min}(\tilde{x}), \frac{\partial \hat{q}_{\min}}{\partial \mu}, \frac{\partial^2 \hat{q}_{\min}}{\partial \mu \partial \mu}, \frac{\partial \hat{q}_{\min x}}{\partial \Sigma}] \leftarrow \mathrm{EP}(\mu, \Sigma)$ ▷ approximate $\hat{p}_{\min}$ (2.3)
  if H=0 **then**
    return $q_{\min}$ ▷ At horizon, return belief for final decision
  else
    $x' \leftarrow \arg\min \langle \mathcal{L} \rangle_x$ ▷ predict information gain; Equation (6)
    $y' \leftarrow$ EVALUATE($f(x')$) ▷ take measurement
    ENTROPYSEARCH($k, l, u, H-1, (x,y) \cup (x',y')$) ▷ move to next evaluation
  end if
end procedure

be a function of previous data, the Horizon H, and any previously collected observations (x,y). To choose where to evaluate next, we first sample discretization points from u, then calculate the current Gaussian belief over f on the discretized domain, along with its derivatives. We construct an approximation to the belief over the minimum using Expectation Propagation, again with derivatives. Finally, we construct a first order approximation on the expected information gain from an evaluation at x' and optimize numerically. We evaluate f at this location, then the cycle repeats. An example implementation in MATLAB can be downloaded from `www.probabilistic-optimization.org`.

3. Experiments

Figures in previous sections provided some intuition and anecdotal evidence for the efficacy of the various approximations used by Entropy Search. In this section, we compare the resulting algorithm to two Gaussian process global optimization heuristics: Expected Improvement, Probability of Improvement (Section 1.1.3), as well as to a continuous armed bandit algorithm: GP-UCB (Srinivas et al., 2010). For reference, we also compare to a number of numerical optimization algorithms: Trust-Region-Reflective (Coleman and Li, 1996, 1994), Active-Set (Powell, 1978b,a), interior point (Byrd et al., 1999, 2000; Waltz et al., 2006), and a naïvely projected version of the BFGS algorithm (Broyden, 1965; Fletcher, 1970; Goldfarb, 1970; Shanno, 1970). We avoid implementation bias by using a uniform code framework for the three Gaussian process-based algorithms, that is, the algorithms share code for the Gaussian process inference and only differ in the way they calculate their utility. For the local numerical algorithms, we used third party code: The projected BFGS method is based on code by Carl Rasmussen,[3] the other methods come from version 6.0 of the optimization toolbox of MATLAB.[4]

In some communities, optimization algorithms are tested on hand-crafted test functions. This runs the risk of introducing bias. Instead, we compare our algorithms on a number of functions sampled from a generative model. In the first experiment, the function is sampled from the model used by the GP algorithms themselves. This eliminates all model-mismatch issues and allows a

3. Code can be found at `http://www.gaussianprocess.org/gpml/code/matlab/util/minimize.m`, version using BFGS: personal communication.
4. Toolbox can be found at `http://www.mathworks.de/help/toolbox/optim/rn/bsqj_zi.html`.

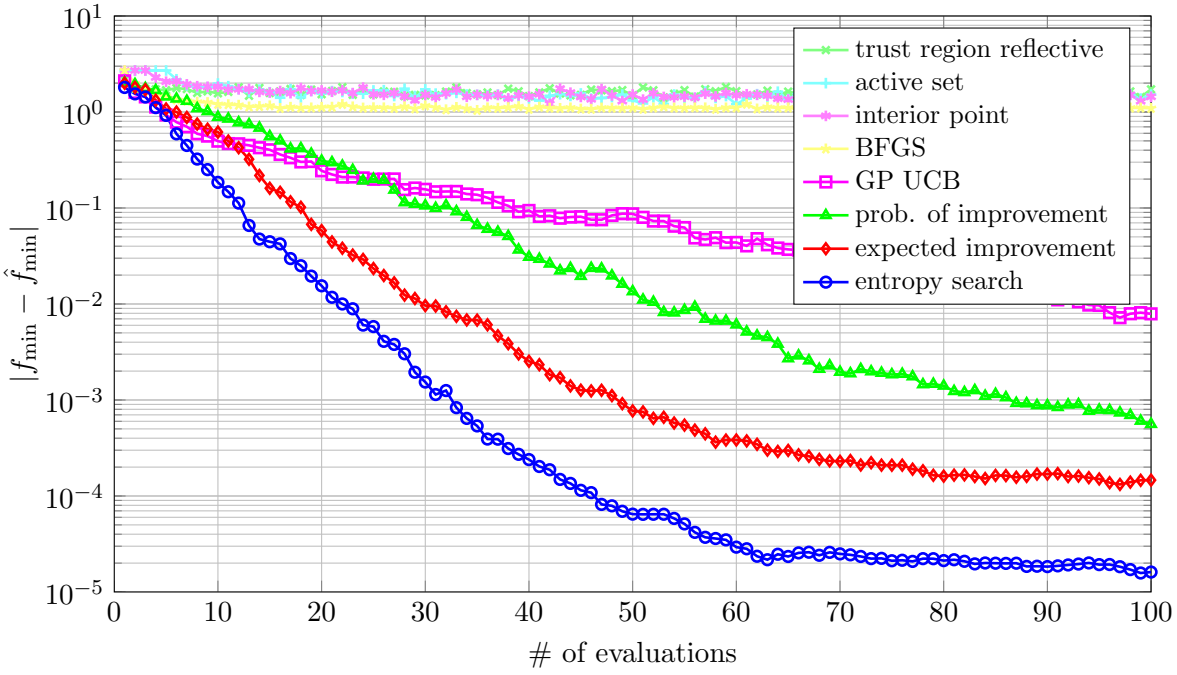


Figure 10: Distance of function value at optimizers' best guess for $x_{\min}$ from true global minimum. Log scale.

direct comparison of other GP optimizers to the probabilistic optimizer. In a second experiment, the functions were sampled from a model strictly more general than the model used by the algorithms, to show the effect of model mismatch.

3.1 Within-Model Comparison

The first experiment was carried out over the 2-dimensional unit domain $I = [0,1]^2$. To generate test functions, 1000 function values were jointly sampled from a Gaussian process with a squared-exponential covariance function of length scale $\ell = 0.1$ in each direction and unit signal variance. The resulting posterior mean was used as the test function. All algorithms had access to noisy evaluations of the test functions. For the benefit of the numerical optimizers, noise was kept relatively low: Gaussian with standard deviation $\sigma = 10^{-3}$. All algorithms were tested on the same set of 40 test functions, all Figures in this section are averages over those sets of functions. It is nontrivial to provide error bars on these average estimates, because the data sets have no parametric distribution. But the regular structure of the plots, given that individual experiments were drawn i.i.d., indicates that there is little remaining stochastic error.

After each function evaluation, the algorithms were asked to return a best guess for the minimum $x_{\min}$. For the local algorithms, this is simply the point of their next evaluation. The Gaussian process based methods returned the global minimum of the mean belief over the function (found by local optimization with random restarts). Figure 10 shows the difference between the global optimum of the function and the function value at the reported best guesses. Since the best guesses do not

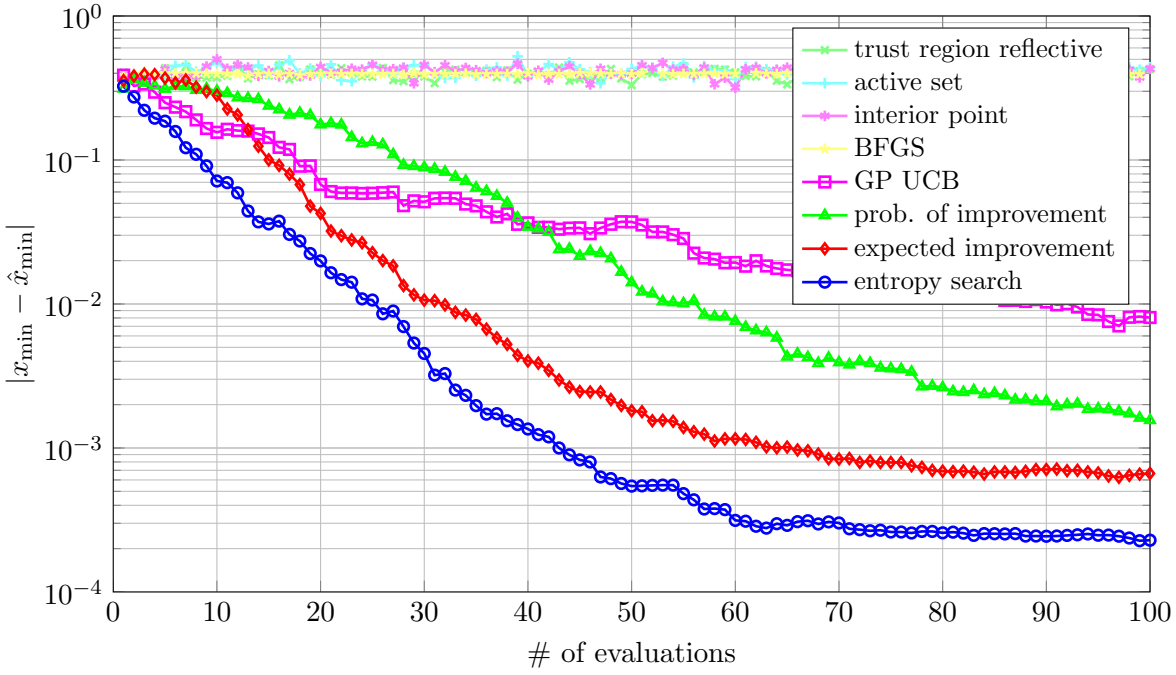


Figure 11: Euclidean distance of optimizers' best guess for $x_{\min}$ from truth. Log scale.

in general lie at a data point, their quality can actually decrease during optimization. The most obvious feature of this plot is that local optimization algorithms are not adept at finding global minima, which is not surprising, but gives an intuition for the difficulty of problems sampled from this generative model. The plot shows a clear advantage for Entropy Search over its competitors, even though the algorithm does not directly aim to optimize this particular loss function. The flattening out of the error of all three global optimizers toward the right is due to evaluation noise (recall that evaluations include Gaussian noise of standard deviation 10^{-3}). Interestingly, Entropy Search flattens out at an error almost an order of magnitude lower than that of the nearest competitor, Expected Improvement. One possible explanation for this behavior is a pathology in the classic heuristics: Both Expected Improvement and Probability of Improvement require a "current best guess" η, which has to be a point estimate, because proper marginalization over an uncertain belief is not tractable. Due to noise, it can thus happen that this best guess is overly optimistic, and the algorithm then explores too aggressively in later stages.

Figure 11 shows data from the same experiments as the previous figure, but plots Euclidean distance from the true global optimum in input space, rather than in function value space. The results from this view are qualitatively similar to those shown in Figure 10.

Since Entropy Search attempts to optimize information gain from evaluations, one would also like to compare to algorithms on the entropy loss function. However, this is challenging. First, the local optimization algorithms provide no probabilistic model of the function and can thus not provide this loss. But even for the optimization algorithms based on Gaussian process measures, it is challenging to evaluate this loss *globally* with good resolution. The only option we are aware of is to approximately calculate entropy, using the very algorithm introduced in this paper. Doing so amounts to a kind of circular experiment that Entropy Search wins by definition, so we omit it here.

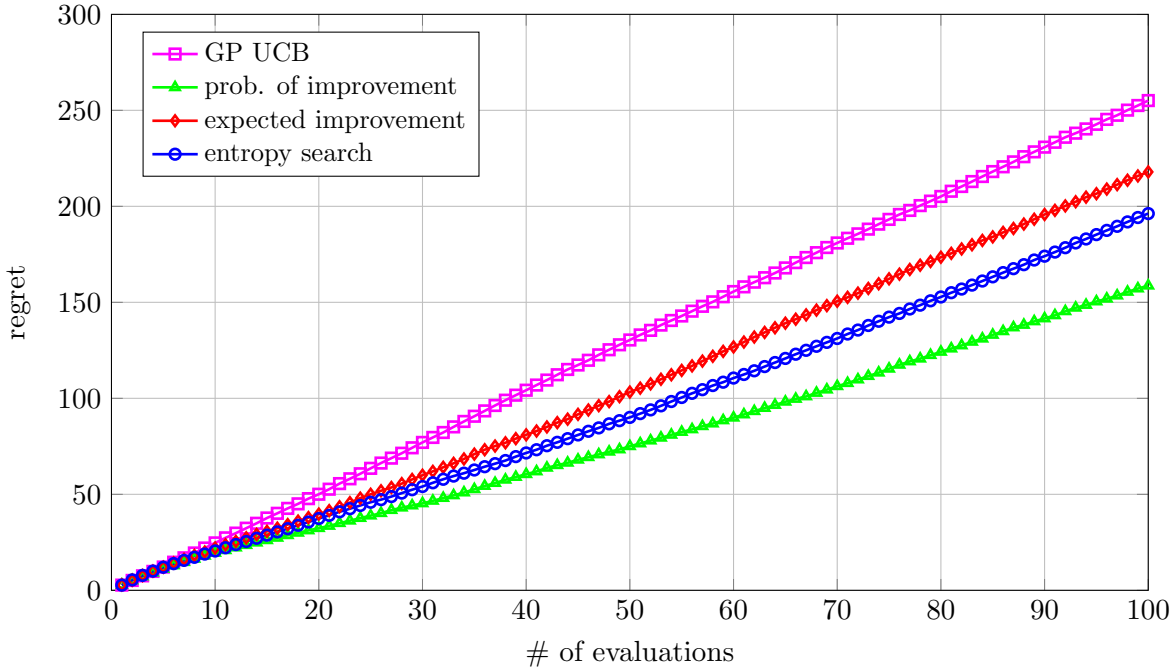


Figure 12: Regret as a function of number of evaluations.

We pointed out in Section 1.1.2 that the bandit setting differs considerably from the kind of optimization discussed in this paper, because bandit algorithms try to minimize regret, rather than improve an estimate of the function's optimum. To clarify this point further, Figure 12 shows the regret

$$r(T) = \sum_{t=1}^{T} [y_t - f_{\min}],$$

for each of the algorithms. Notice that probability of improvement, which performs worst among the global algorithms as seen from the previous two measures of performance, achieves the lowest regret. The intuition here is that this heuristic focuses evaluations on regions known to give low function values. In contrast, the actual value of the function *at the evaluation point* has no special role in Entropy Search. The utility of an evaluation point only depends on its expected effect on knowledge about the minimum of the function.

Surprisingly, the one algorithm explicitly designed to achieve low regret, GP-UCB, performs worst in this comparison. This algorithm chooses evaluation points according to (Srinivas et al., 2010)

$$x_{\text{next}} = \arg\min_x [\mu(x) - \beta^{1/2}\sigma(x)] \qquad \text{where} \qquad \beta = 4(D+1)\log T + C(k,\delta)$$

with T, the number of previous evaluations, D, the dimensionality of the input domain, and $C(k,\delta)$ is a constant that depends on some analytic properties of the kernel k and a free parameter, $0 < \delta < 1$. We found it hard to find a good setting for this δ, which clearly has influence on the algorithm's performance. The results shown here represent the best performance over a set of 4 experiments with different choices for δ. They appear to be slightly worse than, but comparable to the empirical performance reported by the original paper on this algorithm (Srinivas et al., 2010, Figure 5a).

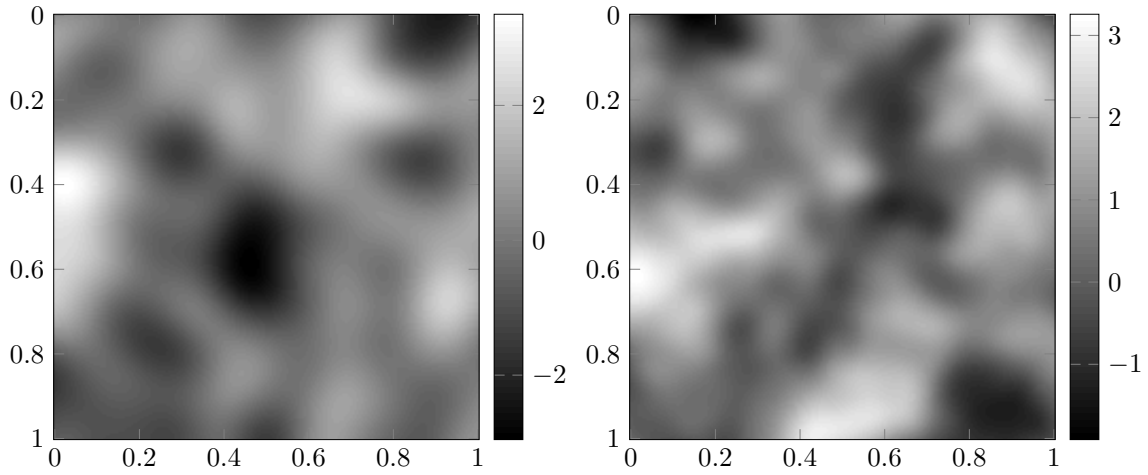


Figure 13: **Left:** A sample from the GP prior with squared exponential kernel used in the on-model experiments of Section 3.1. **Right:** Sample from prior with the rational quadratic kernel used for the out-of-model comparison of Section 3.2.

3.2 Out-of-Model Comparison

In the previous section, the algorithms attempted to find minima of functions sampled from the prior used by the algorithms themselves. In real applications, one can rarely hope to be so lucky, but hierarchical inference can be used to generalize the prior and construct a relatively general algorithm. But what if even the hierarchically extended prior class does not contain the true function? Qualitatively, it is clear that, beyond a certain point of model-mismatch, all algorithms can be made to perform arbitrarily badly. The poor performance of local optimizers (which may be interpreted as building a quadratic model) in the previous section is an example of this effect. In this section, we present results of the same kind of experiments as in the previous section, but on a set of 30 two-dimensional functions sampled from a Gaussian process prior with *rational quadratic* kernel, with the same length scale and signal variance as above, and scale mixture parameter $\alpha = 1$ (see Equation 2.1). This means samples evolve over an infinite number of different length scales, including both longer and shorter scales than those covered by the priors of the algorithms (Figure 13). Figure 14 shows error on function values, Figure 15 Euclidean error in input space, Figure 16 regret. Note the different scales for the ordinate axes relative to the corresponding previous plots: While Entropy Search still (barely) outperforms the competitors, all three algorithms perform worse than before; and their errors become more similar to each other. However, they still manage to discover good regions in the domain, demonstrating a certain robustness to model-mismatch.

4. Conclusion

This paper presented a new probabilistic paradigm for global optimization, as an inference problem on the minimum of the function, rather than the problem of collecting iteratively lower and lower function values. We argue that this description is closer to practitioners' requirements than classic response surface optimization, bandit algorithms, or other, heuristic, global optimization al-

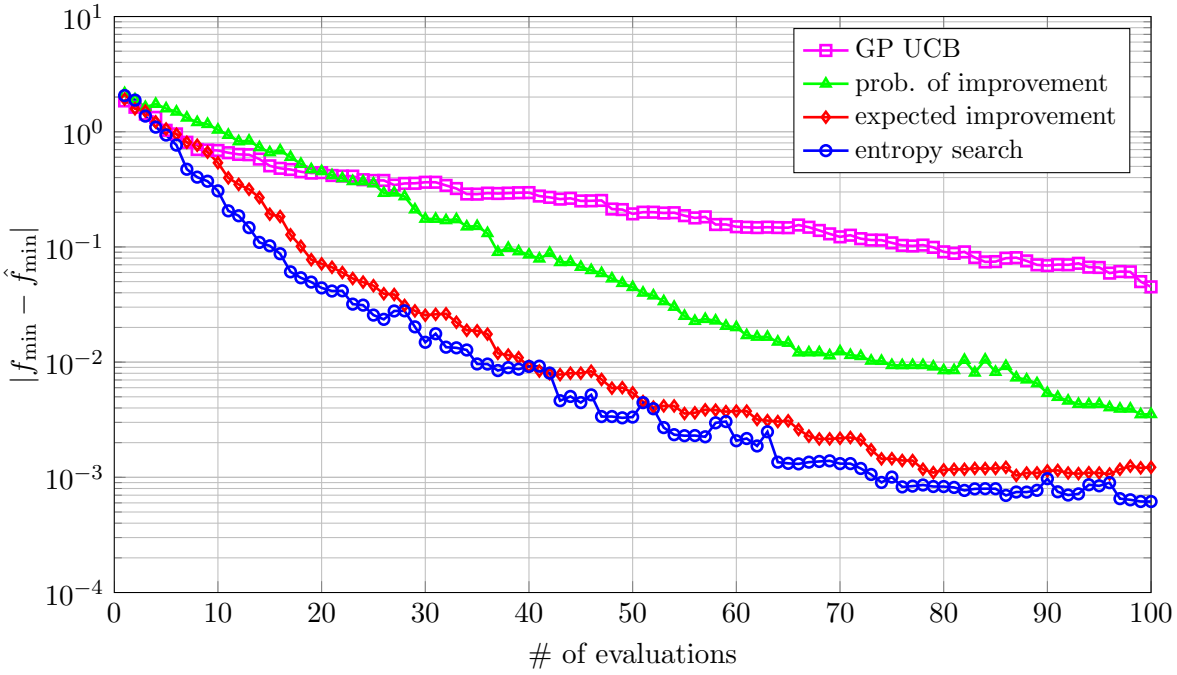


Figure 14: Function value error, off-model tasks.

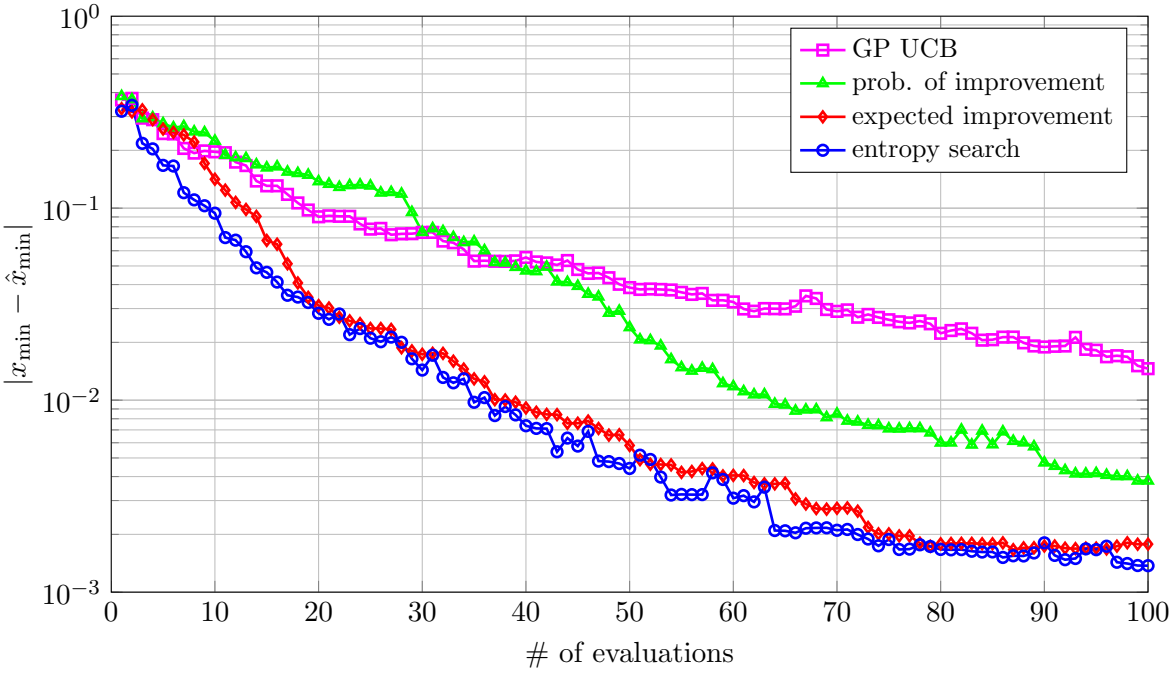


Figure 15: Error on $x_{\min}$, off-model tasks.

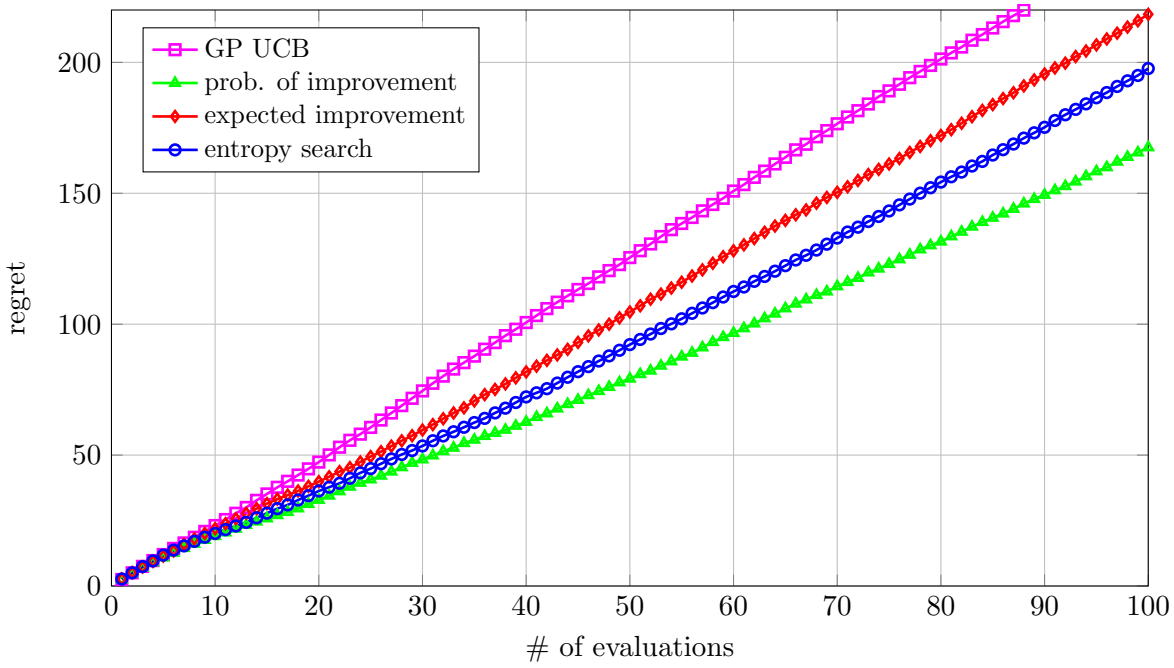


Figure 16: Regret, off-model tasks.

gorithms. In the main part of the paper, we constructed Entropy Search, a practical probabilistic global optimization algorithm, using a series of analytic assumptions and numerical approximations: A particular family of priors over functions (Gaussian processes); constructing the belief $p_{\min}$ over the location of the minimum on an irregular grid to deal with the curse of dimensionality; and using Expectation Propagation toward an efficient analytic approximation. The Gaussian belief allows analytic probabilistic predictions of the effect of future data points, from which we constructed a first-order approximation of the expected change in relative entropy of $p_{\min}$ to a base measure. For completeness, we also pointed out some already known analytic properties of Gaussian process measures that can be used to generalize this algorithm. We showed that the resulting algorithm outperforms both directly and distantly related competitors through its more elaborate, probabilistic description of the problem. This increase in performance is exchanged for somewhat increased computational cost (Entropy Search costs are a constant multiple of that of classic Gaussian process global optimizers); so this algorithm is more suited for problems where evaluating the function itself carries considerable cost. It provides a natural description of the optimization problem, by focusing on the performance under a loss function at the horizon, rather than function values returned during the optimization process. It allows the practitioner to explicitly encode prior knowledge in a flexible way, and adapts its behavior to the user's loss function.

Acknowledgments

We would like to thank Martin Kiefel for valuable feedback, as well as Tom Minka for an interesting discussion.

Appendix A. Mathematical Appendix

The notation in Equation (1) can be read, sloppily, to mean "$p_{\min}(x)$ is the probability that the value of f at x is lower than at any other $\tilde{x} \in I$". For a continuous domain, though, there are uncountably many other $\tilde{x}$. To give more precise meaning to this notation, consider the following argument. Let there be a sequence of locations $\{x_i\}_{i=1,\ldots,N}$, such that for $N \to \infty$ the density of points at each location converges to a measure $m(x)$ nonzero on every open neighborhood in I. If the stochastic process $p(f)$ is sufficiently regular to ensure samples are almost surely continuous (see footnote in Section 2.1), then almost every sample can be approximated arbitrarily well by a staircase function with steps of width $m(x_i)/N$ at the locations x_i, in the sense that $\forall \varepsilon > 0\ \exists N_0 > 0$ such that, $\forall N > N_0 : |f(x) - f(\arg\min_{x_j, j=1,\ldots,N} |x - x_j|)| < \varepsilon$, where $|\cdot|$ is a norm (all norms on finite-dimensional vector spaces are equivalent). This is the original reason why samples from sufficiently regular Gaussian processes can be plotted using finitely many points, in the way used in this paper. We now *define* the notation used in Equation (1) to mean the following limit, where it exists.

$$p_{\min}(x) = \int p(f) \prod_{\tilde{x} \neq x} \theta(f(\tilde{x}) - f(x)) \, \mathrm{d}f$$

$$\equiv \lim_{\substack{N \to \infty \\ |x_i - x_{i-1}| \cdot N \to m(x)}} \int p[f(\{x_i\}_{i=1,\ldots,N})] \prod_{i=1; i \neq j}^{N} \theta[f(x_i) - f(x_j)] \, \mathrm{d}f(\{x_i\}_{i=1,\ldots,N}) \cdot |x_i - x_{i-1}| \cdot N. \quad (8)$$

In words: The "infinite product" is meant to be the limit of finite-dimensional integrals with an increasing number of factors and dimensions, where this limit exists. In doing so, we have sidestepped the issue of whether this limit exists for any particular Gaussian process (kernel function). We do so because the theory of suprema of stochastic processes is highly nontrivial. We refer the reader to a friendly but demanding introduction to the topic by Adler (1990). From our applied standpoint, the issue of whether (8) is well defined for a particular Gaussian prior is secondary: If it is known that the true function is continuous and bounded, then it has a well-defined supremum, and the prior should reflect this knowledge by assigning sufficiently regular beliefs. If the actual prior is such that we expect the function to be discontinuous, it should be clear that optimization is extremely challenging anyway. We conjecture that the finer details of the region between these two domains have little relevance for communities interested in optimization.

References

R.J. Adler. *The Geometry of Random Fields*. Wiley, 1981.

R.J. Adler. An introduction to continuity, extrema, and related topics for general Gaussian processes. *Lecture Notes-Monograph Series*, 12:i–iii+v–vii+ix+1–55, 1990.

Benoit. Note sûre une méthode de résolution des équations normales provenant de l'application de la méthode des moindres carrés a un système d'équations linéaires en nombre inférieure a celui des inconnues. Application de la méthode a la résolution d'un système défini d'équations linéaires. (procédé du Commandant Cholesky). *Bulletin Geodesique*, 7(1):67–77, 1924.

S.P. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge Univ Press, 2004.

C.G. Broyden. A class of methods for solving nonlinear simultaneous equations. *Math. Comp.*, 19 (92):577–593, 1965.

R.H. Byrd, M.E. Hribar, and J. Nocedal. An interior point algorithm for large-scale nonlinear programming. *SIAM Journal on Optimization*, 9(4):877–900, 1999.

R.H. Byrd, J.C. Gilbert, and J. Nocedal. A trust region method based on interior point techniques for nonlinear programming. *Mathematical Programming*, 89(1):149–185, 2000.

T.F. Coleman and Y. Li. On the convergence of interior-reflective newton methods for nonlinear minimization subject to bounds. *Mathematical Programming*, 67(1):189–224, 1994.

T.F. Coleman and Y. Li. An interior trust region approach for nonlinear minimization subject to bounds. *SIAM Journal on Optimization*, 6(2):418–445, 1996.

R.T. Cox. Probability, frequency and reasonable expectation. *American Journal of Physics*, 14(1): 1–13, 1946.

J. Cunningham, P. Hennig, and S. Lacoste-Julien. Gaussian probabilities and expectation propagation. *under review. Preprint at arXiv:1111.6832 [stat.ML]*, November 2011.

R. Fletcher. A new approach to variable metric algorithms. *The Computer Journal*, 13(3):317, 1970.

D. Goldfarb. A family of variable metric updates derived by variational means. *Math. Comp.*, 24 (109):23–26, 1970.

S. Grünewälder, J.Y. Audibert, M. Opper, and J. Shawe-Taylor. Regret bounds for Gaussian process bandit problems. In *Proceedings of the 14th International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2010.

N. Hansen and A. Ostermeier. Completely derandomized self-adaptation in evolution strategies. *Evolutionary Computation*, 9(2):159–195, 2001.

P. Hennig. Optimal reinforcement learning for Gaussian systems. In *Advances in Neural Information Processing Systems*, 2011.

M.R. Hestenes and E. Stiefel. Methods of conjugate gradients for solving linear systems. *Journal of Research of the National Bureau of Standards*, 49(6):409–436, 1952.

K. Itō. On stochastic differential equations. *Memoirs of the American Mathematical Society*, 4, 1951.

E.T. Jaynes and G.L. Bretthorst. *Probability Theory: the Logic of Science*. Cambridge University Press, 2003.

D.R. Jones, M. Schonlau, and W.J. Welch. Efficient global optimization of expensive black-box functions. *Journal of Global Optimization*, 13(4):455–492, 1998.

R. Kleinberg. Nearly tight bounds for the continuum-armed bandit problem. *Advances in Neural Information Processing Systems*, 18, 2005.

A.N. Kolmogorov. Grundbegriffe der Wahrscheinlichkeitsrechnung. *Ergebnisse der Mathematik und ihrer Grenzgebiete*, 2, 1933.

D.G. Krige. A statistical approach to some basic mine valuation and allied problems at the Witwatersrand. Master's thesis, University of Witwatersrand, 1951.

S. Kullback and R.A. Leibler. On information and sufficiency. *Annals of Mathematical Statistics*, 22(1):79–86, 1951.

H. Lazard-Holly and A. Holly. Computation of the probability that a d-dimensional normal variable belongs to a polyhedral cone with arbitrary vertex. Technical report, Mimeo, 2003.

D.J. Lizotte. *Practical Bayesian Optimization*. PhD thesis, University of Alberta, 2008.

D.J.C. MacKay. Choice of basis for Laplace approximation. *Machine Learning*, 33(1):77–86, 1998a.

D.J.C. MacKay. Introduction to Gaussian processes. *NATO ASI Series F Computer and Systems Sciences*, 168:133–166, 1998b.

B. Matérn. Spatial variation. *Meddelanden fran Statens Skogsforskningsinstitut*, 49(5), 1960.

T.P. Minka. Deriving quadrature rules from Gaussian processes. Technical report, Statistics Department, Carnegie Mellon University, 2000.

T.P. Minka. Expectation Propagation for approximate Bayesian inference. In *Proceedings of the 17th Conference in Uncertainty in Artificial Intelligence*, pages 362–369, San Francisco, CA, USA, 2001. Morgan Kaufmann. ISBN 1-55860-800-1.

I. Murray and R.P. Adams. Slice sampling covariance hyperparameters of latent Gaussian models. *arXiv:1006.0868*, 2010.

J. Nocedal and S.J. Wright. *Numerical Optimization*. Springer Verlag, 1999.

M.A. Osborne, R. Garnett, and S.J. Roberts. Gaussian processes for global optimization. In *3rd International Conference on Learning and Intelligent Optimization (LION3)*, 2009.

R.L. Plackett. A reduction formula for normal multivariate integrals. *Biometrika*, 41(3-4):351, 1954.

M.J.D. Powell. The convergence of variable metric methods for nonlinearly constrained optimization calculations. *Nonlinear Programming*, 3(0):27–63, 1978a.

M.J.D. Powell. A fast algorithm for nonlinearly constrained optimization calculations. *Numerical Analysis*, pages 144–157, 1978b.

C.E. Rasmussen and C.K.I. Williams. *Gaussian Processes for Machine Learning*. MIT Press, 2006.

L.M. Schmitt. Theory of genetic algorithms II: models for genetic operators over the string-tensor representation of populations and convergence to global optima for arbitrary fitness function under scaling. *Theoretical Computer Science*, 310(1-3):181–231, 2004.

M. Seeger. Expectation propagation for exponential families. Technical report, U.C. Berkeley, 2008.

D.F. Shanno. Conditioning of quasi-Newton methods for function minimization. *Math. Comp.*, 24 (111):647–656, 1970.

C.E. Shannon. A mathematical theory of communication. *Bell System Technical Journal*, 27, 1948.

N. Srinivas, A. Krause, S. Kakade, and M. Seeger. Gaussian process optimization in the bandit setting: No regret and experimental design. In *International Conference on Machine Learning*, 2010.

M.B. Thompson and R.M. Neal. Slice sampling with adaptive multivariate steps: The shrinking-rank method. *preprint at arXiv:1011.4722*, 2010.

G.E. Uhlenbeck and L.S. Ornstein. On the theory of the Brownian motion. *Physical Review*, 36(5): 823, 1930.

A.W. van der Vaart and J.H. van Zanten. Information rates of nonparametric Gaussian process methods. *Journal of Machine Learning Research*, 12:2095–2119, 2011.

J. Villemonteix, E. Vazquez, and E. Walter. An informational approach to the global optimization of expensive-to-evaluate functions. *Journal of Global Optimization*, 44(4):509–534, 2009.

R.A. Waltz, J.L. Morales, J. Nocedal, and D. Orban. An interior algorithm for nonlinear optimization that combines line search and trust region steps. *Mathematical Programming*, 107(3): 391–408, 2006.

N. Wiener and P. Masani. The prediction theory of multivariate stochastic processes. *Acta Mathematica*, 98(1):111–150, 1957.

Journal of Machine Learning Research 13 (2012) 1839-1864 Submitted 1/11; Revised 12/11; Published 6/12

Estimation and Selection via Absolute Penalized Convex Minimization And Its Multistage Adaptive Applications

Jian Huang JIAN-HUANG@UIOWA.EDU
Department of Statistics and Actuarial Science
University of Iowa
Iowa City, IA 52242, USA

Cun-Hui Zhang CZHANG@STAT.RUTGERS.EDU
Department of Statistics and Biostatistics
Rutgers University
Piscataway, New Jersey 08854, USA

Editor: Hui Zhou

Abstract

The ℓ_1-penalized method, or the Lasso, has emerged as an important tool for the analysis of large data sets. Many important results have been obtained for the Lasso in linear regression which have led to a deeper understanding of high-dimensional statistical problems. In this article, we consider a class of weighted ℓ_1-penalized estimators for convex loss functions of a general form, including the generalized linear models. We study the estimation, prediction, selection and sparsity properties of the weighted ℓ_1-penalized estimator in sparse, high-dimensional settings where the number of predictors p can be much larger than the sample size n. Adaptive Lasso is considered as a special case. A multistage method is developed to approximate concave regularized estimation by applying an adaptive Lasso recursively. We provide prediction and estimation oracle inequalities for single- and multi-stage estimators, a general selection consistency theorem, and an upper bound for the dimension of the Lasso estimator. Important models including the linear regression, logistic regression and log-linear models are used throughout to illustrate the applications of the general results.

Keywords: variable selection, penalized estimation, oracle inequality, generalized linear models, selection consistency, sparsity

1. Introduction

High-dimensional data arise in many diverse fields of scientific research. For example, in genetic and genomic studies, more and more large data sets are being generated with rapid advances in biotechnology, where the total number of variables p is larger than the sample size n. Fortunately, statistical analysis is still possible for a substantial subset of such problems with a sparse underlying model where the number of important variables is much smaller than the sample size. A fundamental problem in the analysis of such data is to find reasonably accurate sparse solutions that are easy to interpret and can be used for the prediction and estimation of covariable effects. The ℓ_1-penalized method, or the Lasso (Tibshirani, 1996; Chen et al., 1998), has emerged as an important approach to finding such solutions in sparse, high-dimensional statistical problems.

In the last few years, considerable progress has been made in understanding the theoretical properties of the Lasso in $p \gg n$ settings. Most results have been obtained for linear regression

models with a quadratic loss. Greenshtein and Ritov (2004) studied the prediction performance of the Lasso in high-dimensional least squares regression. Meinshausen and Bühlmann (2006) showed that, for neighborhood selection in the Gaussian graphical models, under a neighborhood stability condition on the design matrix and certain additional regularity conditions, the Lasso is selection consistent even when $p \to \infty$ at a rate faster than n. Zhao and Yu (2006) formalized the neighborhood stability condition in the context of linear regression as a strong irrepresentable condition. Candes and Tao (2007) derived an upper bound for the ℓ_2 loss of a closely related Dantzig selector in the estimation of regression coefficients under a condition on the number of nonzero coefficients and a uniform uncertainty principle on the design matrix. Similar results have been obtained for the Lasso. For example, upper bounds for the ℓ_q loss of the Lasso estimator has being established by Bunea et al. (2007) for $q = 1$, Zhang and Huang (2008) for $q \in [1;2]$, Meinshausen and Yu (2009) for $q = 2$, Bickel et al. (2009) for $q \in [1;2]$, and Zhang (2009) and Ye and Zhang (2010) for general $q \geq 1$. For convex minimization methods beyond linear regression, van de Geer (2008) studied the Lasso in high-dimensional generalized linear models (GLM) and obtained prediction and ℓ_1 estimation error bounds. Negahban et al. (2010) studied penalized M-estimators with a general class of regularizers, including an ℓ_2 error bound for the Lasso in GLM under a restricted strong convexity and other regularity conditions.

Theoretical studies of the Lasso have revealed that it may not perform well for the purpose of variable selection, since its required irrepresentable condition is not properly scaled in the number of relevant variables. In a number of simulation studies, the Lasso has shown weakness in variable selection when the number of nonzero regression coefficients increases. As a remedy, a number of proposals have been introduced in the literature and proven to be variable selection consistent under regularity conditions of milder forms, including concave penalized LSE (Fan and Li, 2001; Zhang, 2010a), adaptive Lasso (Zou, 2006; Meier and Bühlmann, 2007; Huang et al., 2008), stepwise regression (Zhang, 2011a), and multi-stage methods (Hunter and Li, 2005; Zou and Li, 2008; Zhang, 2010b, 2011b).

In this article, we study a class of weighted ℓ_1-penalized estimators with a convex loss function. This class includes the Lasso, adaptive Lasso and multistage recursive application of adaptive Lasso in generalized linear models as special cases. We study prediction, estimation, selection and sparsity properties of the weighted ℓ_1-penalized estimator based on a convex loss in sparse, high-dimensional settings where the number of predictors p can be much larger than the sample size n. The main contributions of this work are as follows.

- We extend the existing theory for the unweighted Lasso from linear regression to more general convex loss function.
- We develop a multistage method to approximate concave regularized convex minimization with recursive application of adaptive Lasso, and provide sharper risk bounds for this concave regularization approach in the general setting.
- We apply our results to a number of important special cases, including the linear, logistic and log-linear regression models.

This article is organized as follows. In Section 2 we describe a general formulation of the absolute penalized minimization problem with a convex loss, along with two basic inequalities and a number of examples. In Section 3 we develop oracle inequalities for the weighted Lasso estimator for general quasi star-shaped loss functions and an ℓ_2 bound on the prediction error. In

Section 4 we develop multistage recursive applications of adaptive Lasso as an approximate concave regularization method and provide sharper oracle inequalities for this approach. In Section 5 we derive sufficient conditions for selection consistency. In Section 6 we provide an upper bound on the dimension of the Lasso estimator. Concluding remarks are given in Section 7. All proofs are provided in an appendix.

2. Absolute Penalized Convex Minimization

In this section, we define the weighted Lasso for a convex loss function and characterize its solutions via the KKT conditions. We then derive some basic inequalities for the weighted Lasso solutions in terms of the symmetrized Bregman divergence (Bregman, 1967; Nielsen and Nock, 2007). We also illustrate the applications of the basic inequalities in several important examples.

2.1 Definition and the KKT Conditions

We consider a general convex loss function of the form

$$\ell(\beta) = \psi(\beta) - \langle \beta, z \rangle, \tag{1}$$

where $\psi(\beta)$ is a known convex function, z is observed, and β is unknown. Unless otherwise stated, the inner product space is $\mathbb{R}^p$, so that $\{z, \beta\} \subset \mathbb{R}^p$ and $\langle \beta, z \rangle = \beta' z$. Our analysis of (1) requires certain smoothness of the function $\psi(\beta)$ in terms of its differentiability. In what follows, such smoothness assumptions are always explicitly described by invoking the derivative of ψ. For any $v = (v_1, \ldots, v_p)'$, we use $\|v\|$ to denote a general norm of v and $|v|_q$ the ℓ_q norm $(\sum_j |v_j|^q)^{1/q}$, with $|v|_\infty = \max_j |v_j|$. Let $\widehat{w} \in \mathbb{R}^p$ be a (possibly estimated) weight vector with nonnegative elements $\widehat{w}_j, 1 \le j \le p$, and $\widehat{W} = \mathrm{diag}(\widehat{w})$. The weighted absolute penalized estimator, or weighted Lasso, is defined as

$$\widehat{\beta} = \arg\min_{\beta} \left\{ \ell(\beta) + \lambda |\widehat{W}\beta|_1 \right\}. \tag{2}$$

Here we focus on the case where $\widehat{W}$ is diagonal. In linear regression, Tibshirani and Taylor (2011) considered non-diagonal, predetermined $\widehat{W}$ and derived an algorithm for computing the solution paths.

A vector $\widehat{\beta}$ is a global minimizer in (2) if and only if the negative gradient at $\widehat{\beta}$ satisfies the Karush-Kuhn-Tucker (KKT) conditions,

$$g = -\dot{\ell}(\widehat{\beta}) = z - \dot{\psi}(\widehat{\beta}), \ \begin{cases} g_j = \widehat{w}_j \lambda \, \mathrm{sgn}(\widehat{\beta}_j) & \text{if } \widehat{\beta}_j \neq 0 \\ g_j \in \widehat{w}_j \lambda [-1, 1] & \text{all } j, \end{cases} \tag{3}$$

where $\dot{\ell}(\beta) = (\partial/\partial\beta)\ell(\beta)$ and $\dot{\psi}(\beta) = (\partial/\partial\beta)\psi(\beta)$. Since the KKT conditions are necessary and sufficient for (2), results on the performance of $\widehat{\beta}$ can be viewed as analytical consequences of (3).

The estimator (2) includes the ℓ_1-penalized estimator, or the Lasso, with the choice $\widehat{w}_j = 1, 1 \le j \le p$. A careful study of the (unweighted) Lasso in general convex minimization (1) is by itself an interesting and important problem. Our work includes the Lasso as a special case since $\widehat{w}_j = 1$ is allowed in our theorems.

In practice, unequal $\widehat{w}_j$ arise in many ways. In adaptive Lasso (Zou, 2006), a decreasing function of a certain initial estimator of β_j is used as the weight $\widehat{w}_j$ to remove the bias of the Lasso. In Zou and Li (2008) and Zhang (2010b), the weights $\widehat{w}_j$ are computed iteratively with $\widehat{w}_j = \dot{\rho}_\lambda(\widehat{\beta}_j)$, where $\dot{\rho}_\lambda(t) = (d/dt)\rho_\lambda(t)$ with a suitable concave penalty function $\rho_\lambda(t)$. This is also designed to remove the bias of the Lasso, since the concavity of $\rho_\lambda(t)$ guarantees smaller weight for larger $\widehat{\beta}_j$. In Section 4, we provide results on the improvements of this weighted Lasso over the standard Lasso. In linear regression, Zhang (2010b) gave sufficient conditions under which this iterative algorithm provides smaller weights $\widehat{w}_j$ for most large β_j. Such nearly unbiased methods are expected to produce better results than the Lasso when a significant fraction of nonzero $|\beta_j|$ are of the order λ or larger. Regardless of the computational methods, the results in this paper demonstrate the benefits of using data dependent weights in a general class of problems with convex losses.

Unequal weights may also arise for computational reasons. The Lasso with $\widehat{w}_j = 1$ is expected to perform similarly to weighted Lasso with data dependent $1 \le \widehat{w}_j \le C_0$, with a fixed C_0. However, the weighted Lasso is easier to compute since $\widehat{w}_j$ can be determined as a part of an iterative algorithm. For example, in a gradient descent algorithm, one may take larger steps and stop the computation as soon as the KKT conditions (3) are attained for any weights satisfying $1 \le \widehat{w}_j \le C_0$.

The weight function $\widehat{w}_j$ can be also used to standardize the penalty level, for example with $\widehat{w}_j = \{\ddot{\psi}_{jj}(\widehat{\beta})\}^{1/2}$, where $\ddot{\psi}_{jj}(\beta)$ is the j-th diagonal element of the Hessian matrix of $\psi(\beta)$. When $\psi(\beta)$ is quadratic, for example in linear regression, $\widehat{w}_j = \{\ddot{\psi}_{jj}(\widehat{\beta})\}^{1/2}$ does not depend on $\widehat{\beta}$. However, in other convex minimization problems, such weights need to be computed iteratively.

Finally, in certain applications, the effects of a certain set S_* of variables are of primary interest, so that penalization of $\widehat{\beta}_{S_*}$, and thus the resulting bias, should be avoided. This leads to "semi-penalized" estimators with $\widehat{w}_j = 0$ for $j \in S_*$, for example, with weights $\widehat{w}_i = I\{j \notin S_*\}$.

2.2 Basic Inequalities, Prediction, and Bregman Divergence

Let β^* denote a target vector for β. In high-dimensional models, the performance of an estimator $\widehat{\beta}$ is typically measured by its proximity to a target under conditions on the sparsity of β^* and the size of the negative gradient $-\dot{\ell}(\beta^*) = z - \dot{\psi}(\beta^*)$. For ℓ_1-penalized estimators, such results are often derived from the KKT conditions (3) via certain basic inequalities, which are direct consequences of the KKT conditions and have appeared in different forms in the literature, for example, in the papers cited in Section 1. Let $D(\beta, \beta^*) = \ell(\beta) - \ell(\beta^*) - \langle \dot{\ell}(\beta^*), \beta - \beta^* \rangle$ be the Bregman divergence (Bregman, 1967) and consider its symmetrized version (Nielsen and Nock, 2007)

$$\Delta(\beta, \beta^*) = D(\beta, \beta^*) + D(\beta^*, \beta) = \big\langle \beta - \beta^*, \dot{\psi}(\beta) - \dot{\psi}(\beta^*) \big\rangle. \tag{4}$$

Since ψ is convex, $\Delta(\beta, \beta^*) \ge 0$. Two basic inequalities below provide upper bounds for the symmetrized Bregman divergence $\Delta(\widehat{\beta}, \beta^*)$. The sparsity of β^* is measured by a weighted ℓ_1 norm of β^* in the first one and by a sparse set in the second one.

Let S be any set of indices satisfying $S \supseteq \{j : \beta^*_j \ne 0\}$ and let S^c be the complement of S in $\{1, \ldots, p\}$. We shall refer to S as the sparse set. Let $W = \mathrm{diag}(w)$ for a possibly unknown vector $w \in \mathbb{R}^p$ with elements $w_j \ge 0$. Define

$$z_0^* = |\{z - \dot{\psi}(\beta^*)\}_S|_\infty,\ z_1^* = |W_{S^c}^{-1}\{z - \dot{\psi}(\beta^*)\}_{S^c}|_\infty, \tag{5}$$

$$\Omega_0 = \big\{\widehat{w}_j \le w_j\ \forall j \in S\big\} \cap \big\{w_j \le \widehat{w}_j\ \forall j \in S^c\big\}, \tag{6}$$

where for any p-vector v and set A, $v_A = (v_j : j \in A)'$. Here and in the sequel M_{AB} denotes the $A \times B$ subblock of a matrix M and $M_A = M_{AA}$.

Lemma 1 *(i) Let β^* be a target vector. In the event $\Omega_0 \cap \{|(z - \psi(\beta^*))_j| \le \widehat{w}_j \lambda \ \forall j\}$,*

$$\Delta(\widehat{\beta}, \beta^*) \le 2\lambda |\widehat{W}\beta^*|_1 \le 2\lambda |W\beta^*|_1. \tag{7}$$

(ii) For any target vector β^* and $S \supseteq \{j : \beta^*_j \neq 0\}$, the error $h = \widehat{\beta} - \beta^*$ satisfies

$$\begin{aligned} \Delta(\beta^* + h, \beta^*) + (\lambda - z_1^*)|W_{S^c} h_{S^c}|_1 &\le \langle h_S, g_S - \{z - \psi(\beta^*)\}_S \rangle \\ &\le (|w_S|_\infty \lambda + z_0^*)|h_S|_1 \end{aligned} \tag{8}$$

in Ω_0 for a certain negative gradient vector g satisfying $|g_j| \le \widehat{w}_j \lambda$. Consequently, in $\Omega_0 \cap \{(|w_S|_\infty \lambda + z_0^*)/(\lambda - z_1^*) \le \xi\}$, $h \neq 0$ belongs to the sign-restricted cone $\mathscr{C}_-(\xi, S) = \{b \in \mathscr{C}(\xi, S) : b_j(\psi(\beta + b) - \psi(\beta))_j \le 0 \ \forall j \in S^c\}$, where

$$\mathscr{C}(\xi, S) = \left\{ b \in \mathbb{R}^p : |W_{S^c} b_{S^c}|_1 \le \xi |b_S|_1 \neq 0 \right\}. \tag{9}$$

Remark 2 *Sufficient conditions are given in Subsection 3.2 for $\{|(z - \psi(\beta^*))_j| \le \widehat{w}_j \lambda \ \forall j\}$ to hold with high probability in generalized linear models. See Lemma 8, Remarks 10 and 11 and Examples 7, 8, and 9.*

A useful feature of Lemma 1 is the explicit statements of the monotonicity of the basic inequality in the weights. By Lemma 1 (ii), it suffices to study the analytical properties of the penalized criterion with the error $h = \widehat{\beta} - \beta^*$ in the sign-restricted cone, provided that the event $(|w_S|_\infty \lambda + z_0^*)/(\lambda - z_1^*) \le \xi$ has large probability. However, unless $\mathscr{C}_-(\xi, S)$ is specified, we will consider the larger cone in (9) in order to simplify the analysis. The choices of the target vector β^*, the sparse set $S \supseteq \{j : \beta^*_j \neq 0\}$, weight vector $\widehat{w}$ and its bound w are quite flexible. The main requirement is that $\{|S|, z_0^*, z_1^*\}$ should be small. In linear regression or generalized linear models, we may conveniently consider β^* as the vector of true regression coefficients under a probability measure P_{β^*}. However, β^* can also be a sparse version of a true β, for example, $\beta^*_j = \beta_j I\{|\beta_j| \ge \tau\}$ for a threshold value τ under P_β.

The upper bound in Lemma 1 (i) gives the so called "slow rate" of convergence for the Bregman divergence. In Section 3, we provide "fast rate" of convergence for the Bregman divergence via oracle inequalities for $|h_S|_1$ in (8).

The symmetrized Bregman divergence $\Delta(\widehat{\beta}, \beta^*)$ has the interpretations as the regret in prediction error in linear regression, the symmetrized Kullback-Leibler (KL) divergence in generalized linear models (GLM) and density estimation, and a spectrum loss for the graphical Lasso, as shown in examples below. These quantities can be all viewed as the size of the prediction error since they measure distances between a target density of the observations and an estimated density.

Example 1 ***(Linear regression)*** *Consider the linear regression model*

$$y_i = \sum_{j=1}^p x_{ij} \beta_j + \varepsilon_i, \ \ i = 1, \ldots, n, \tag{10}$$

where y_i is the response variable, x_{ij} are predictors or design variables, and ε_i is the error term. Let $y = (y_1, \ldots, y_n)'$ and let X be the design matrix whose ith row is $x^i = (x_{i1}, \ldots, x_{ip})$. The estimator (2) can be written as a weighted Lasso with $\psi(\beta) = |X\beta|_2^2/(2n)$ and $z = X'y/n$ in (1). For predicting a vector $\widetilde{y}$ with $\mathrm{E}_{\beta^*}[\widetilde{y}|X,y] = X\beta^*$,

$$n\Delta(\widehat{\beta}, \beta^*) = |X\widehat{\beta} - X\beta^*|_2^2 = \mathrm{E}_{\beta^*}\big[|\widetilde{y} - X\widehat{\beta}|_2^2 \big| X, y\big] - \min_{\delta(X,y)} \mathrm{E}_{\beta^*}\big[|\widetilde{y} - \delta(X,y)|_2^2 \big| X, y\big]$$

is the regret of using the linear predictor $X\widehat{\beta}$ compared with the optimal predictor. See Greenshtein and Ritov (2004) for several implications of (7).

Example 2 ***(Logistic regression)*** *We observe $(X,y) \in \mathbb{R}^{n\times(p+1)}$ with independent rows (x^i, y_i), where $y_i \in \{0,1\}$ are binary response variables with*

$$\mathrm{P}_\beta(y_i = 1|x^i) = \pi_i(\beta) = \exp(x^i\beta)/(1+\exp(x^i\beta)),\ 1 \le i \le n. \tag{11}$$

The loss function (1) is the average negative log-likelihood:

$$\ell(\beta) = \psi(\beta) - z'\beta \ \ \textit{with} \ \ \psi(\beta) = \sum_{i=1}^n \frac{\log(1+\exp(x^i\beta))}{n},\ z = X'y/n. \tag{12}$$

Thus, (2) is a weighted ℓ_1 penalized MLE. For probabilities $\{\pi', \pi''\} \subset (0,1)$, the KL information is $K(\pi', \pi'') = \pi' \log(\pi'/\pi'') + (1-\pi')\log\{(1-\pi')/(1-\pi'')\}$. Since $\dot{\psi}(\beta) = \sum_{i=1}^n x^i \pi_i(\beta)/n$ and $logit(\pi_i(\beta^*)) - logit(\pi_i(\beta)) = x^i(\beta^* - \beta)$, (4) gives

$$\Delta(\beta, \beta^*) = \frac{1}{n}\sum_{i=1}^n \Big\{ K\big(\pi_i(\beta^*), \pi_i(\beta)\big) + K\big(\pi_i(\beta), \pi_i(\beta^*)\big)\Big\}.$$

Thus, the symmetrized Bregman divergence $\Delta(\beta^*, \beta)$ is the symmetrised KL-divergence.

Example 3 ***(GLM).*** *The GLM contains the linear and logistic regression models as special cases. We observe $(X,y) \in \mathbb{R}^{n\times(p+1)}$ with rows (x^i, y_i). Suppose that conditionally on X, y_i are independent under P_β with*

$$y_i \sim f(y_i|\theta_i) = \exp\Big(\frac{\theta_i y_i - \psi_0(\theta_i)}{\sigma^2} + \frac{c(y_i,\sigma)}{\sigma^2}\Big),\ \theta_i = x^i\beta. \tag{13}$$

Let $f_{(n)}(y|X,\beta) = \prod_{i=1}^n f(y_i|x^i\beta)$. The loss function can be written as a normalized negative likelihood $\ell(\beta) = (\sigma^2/n)\log f_{(n)}(y|X,\beta)$ with $\psi(\beta) = \sum_{i=1}^n \{\psi_0(x^i\beta) + c(y_i,\sigma)\}/n$ and $z = X'y/n$. The KL divergence is

$$D\Big(f_n(\cdot|X,\beta^*)\Big\|f_n(\cdot|X,\beta)\Big) = \mathrm{E}_{\beta^*}\log\Big(\frac{f_{(n)}(y|X,\beta^*)}{f_{(n)}(y|X,\beta)}\Big).$$

The symmetrized Bregman divergence can be written as

$$\Delta(\widehat{\beta}, \beta^*) = \frac{\sigma^2}{n}\Big\{D\Big(f_{(n)}(\cdot|X,\beta^*)\Big\|f_{(n)}(\cdot|X,\widehat{\beta})\Big) + D\Big(f_{(n)}(\cdot|X,\widehat{\beta})\Big\|f_{(n)}(\cdot|X,\beta^*)\Big)\Big\}.$$

Example 4 ***(Nonparametric density estimation)*** *Although the focus of this paper is on regression models, here we illustrate that $\Delta(\widehat{\beta},\beta^*)$ is the symmetrised KL divergence in the context of nonparametric density estimation. Suppose the observations $y=(y_1,\ldots,y_n)'$ are iid from $f(\cdot|\beta)=\exp\{\langle\beta,T(\cdot)\rangle-\psi(\beta)\}$ under P_β, where $T(\cdot)=(u_j(\cdot),j\le p)'$ with certain basis functions $u_j(\cdot)$. Let the loss function $\ell(\beta)$ in (1) be the average negative log-likelihood $n^{-1}\sum_{i=1}^n\log f(y_i|\beta)$ with $z=n^{-1}\sum_{i=1}^n T(y_i)$. Since $E_\beta T(y_i)=\dot\psi(\beta)$, the KL divergence is*

$$D\Big(f(\cdot|\beta^*)\Big\|f(\cdot|\beta)\Big)=E_{\beta^*}\log\Big(\frac{f(y_i|\beta^*)}{f(y_i|\beta)}\Big)=\psi(\beta)-\psi(\beta^*)-\langle\beta-\beta^*,\dot\psi(\beta^*)\rangle.$$

Again, the symmetrized Bregman divergence is the symmetrised KL divergence between the target density $f(\cdot|\beta^*)$ and the estimated density $f(\cdot|\widehat{\beta})$:

$$\Delta(\beta,\beta^*)=D\Big(f(\cdot|\beta^*)\Big\|f(\cdot|\widehat{\beta})\Big)+D\Big(f(\cdot|\widehat{\beta})\Big\|f(\cdot|\beta^*)\Big).$$

van de Geer (2008) pointed out that for this example, the natural choices of the basis functions u_j and weights w_j satisfy $\int u_j d\nu=0$ and $w_k^2=\int u_k^2 d\nu$.

Example 5 ***(Graphical Lasso)*** *Suppose we observe $X\in\mathbb{R}^{n\times p}$ and would like to estimate the precision matrix $\beta=(EX'X/n)^{-1}\in\mathbb{R}^{p\times p}$. In the graphical Lasso, (1) is the length normalized negative likelihood with $\psi(\beta)=-\log\det\beta$, $z=-X'X/n$, and $\langle\beta,z\rangle=-\mathrm{trace}(\beta z)$. Since the gradient of ψ is $\dot\psi(\beta)=E_\beta z=-\beta^{-1}$, we find*

$$\Delta(\beta,\beta^*)=\mathrm{trace}\big((\widehat{\beta}-\beta^*)((\beta^*)^{-1}-\widehat{\beta}^{-1}\big)=\sum_{j=1}^p(\lambda_j-1)^2/\lambda_j,$$

where $(\lambda_1,\ldots,\lambda_p)$ are the eigenvalues of $(\beta^*)^{-1/2}\widehat{\beta}(\beta^*)^{-1/2}$. In graphical Lasso, the diagonal elements are typically not penalized. Consider $\widehat{w}_{jk}=I\{j\ne k\}$, so that the penalty for the off-diagonal elements are uniformly weighted. Since Lemma 1 requires $|(z-\dot\psi(\beta^*))_{jk}|\le\widehat{w}_{jk}\lambda$, β^* is taken to match $X'X/n$ on the diagonal and the true β in correlations. Let $S=\{(j,k):\beta_{jk}\ne0,j\ne k\}$. In the event $\max_{j\ne k}|z_{jk}-\beta^*_{jk}|\le\lambda$, Lemma 1 (i) gives

$$|S|\lambda\max_{j\ne k}|\beta^*_{jk}|=o(1)\ \Rightarrow\ \|(\beta^*)^{-1/2}\widehat{\beta}(\beta^*)^{-1/2}-I_{p\times p}\|_2=o(1)$$

where $\|\cdot\|_2$ is the spectrum norm. Rothman et al. (2008) proved the consistency of the graphical Lasso under similar conditions with a different analysis.

3. Oracle Inequalities

In this section, we extract upper bounds for the estimation error $\widehat{\beta}-\beta^*$ from the basic inequality (8). Since (8) is monotone in the weights, the oracle inequalities are sharper when the weights $\widehat{w}_j$ are smaller in $S\supseteq\{j:\beta^*_j\ne0\}$ and larger in S^c.

We say that a function $\phi(b)$ defined in $\mathbb{R}^p$ is quasi star-shaped if $\phi(tb)$ is continuous and non-decreasing in $t\in[0,\infty)$ for all $b\in\mathbb{R}^p$ and $\lim_{b\to0}\phi(b)=0$. All seminorms are quasi star-shaped. The sublevel sets $\{b:\phi(b)\le t\}$ of a quasi star-shaped function are all star-shaped. Constant factors of the following form play a crucial role in our analysis.

Definition 3 *For $0 \le \eta^* \le 1$ and any pair of quasi star-shaped functions $\phi_0(b)$ and $\phi(b)$, define a general invertibility factor (GIF) over the cone (9) as follows:*

$$F(\xi, S; \phi_0, \phi) = \inf\Big\{ \frac{\Delta(\beta^* + b, \beta^*) e^{\phi_0(b)}}{|b_S|_1 \phi(b)} : b \in \mathscr{C}(\xi, S), \phi_0(b) \le \eta^* \Big\}, \tag{14}$$

where $\Delta(\beta, \beta^*)$ is as in (4).

The GIF extends the squared compatibility constant (van de Geer and Bühlmann, 2009) and the weak and sign-restricted cone invertibility factors (Ye and Zhang, 2010) from the linear regression model with $\phi_0(\cdot) = 0$ to the general model (1) and from ℓ_q norms to general $\phi(\cdot)$. They are all closely related to the restricted eigenvalues (Bickel et al., 2009; Koltchinskii, 2009) as we will discuss in Subsection 3.1.

The basic inequality (8) implies that the symmetrized Bregman divergence $\Delta(\widehat{\beta}, \beta^*)$ is no greater than a linear function of $|h_S|_1$, where $h = \widehat{\beta} - \beta^*$. If $\Delta(\widehat{\beta}, \beta^*)$ is no smaller than a linear function of the product $|h_S|_1 \phi(h)$, then an upper bound for $\phi(h)$ exists. Since the symmetrized Bregman divergence (4) is approximately quadratic, $\Delta(\widehat{\beta}, \beta^*) \approx \langle h, \ddot{\psi}(\beta^*) h \rangle$, in a neighborhood of β^*, this is reasonable when $h = \widehat{\beta} - \beta^*$ is not too large and $\ddot{\psi}(\beta^*)$ is invertible in the cone. A suitable factor $e^{\phi_0(b)}$ in (14) forces the computation of this lower bound in a proper neighborhood of β^*.

We first provide a set of general oracle inequalities.

Theorem 4 *Let $\{z_0^*, z_1^*\}$ be as in (5) with $S \supseteq \{j : \beta_j^* \ne 0\}$, Ω_0 in (6), $0 \le \eta \le \eta^* \le 1$, and $\{\phi_0(b), \phi(b)\}$ be a pair of quasi star-shaped functions. Then, in the event*

$$\Omega_1 = \Omega_0 \cap \Big\{ \frac{|w_S|_\infty \lambda + z_0^*}{(\lambda - z_1^*)_+} \le \xi, \ \frac{|w_S|_\infty \lambda + z_0^*}{F(\xi, S; \phi_0, \phi_0)} \le \eta e^{-\eta} \Big\}, \tag{15}$$

the following oracle inequalities hold:

$$\phi_0(\widehat{\beta} - \beta^*) \le \eta, \quad \phi(\widehat{\beta} - \beta^*) \le \frac{e^\eta (|w_S|_\infty \lambda + z_0^*)}{F(\xi, S; \phi_0, \phi)}, \tag{16}$$

and with $\phi_{1,S}(b) = |b_S|_1 / |S|$

$$\Delta(\widehat{\beta}, \beta^*) + (\lambda - z_1^*) |W_{S^c}(\widehat{\beta} - \beta^*)_{S^c}|_1 \le \frac{e^\eta (|w_S|_\infty \lambda + z_0^*)^2 |S|}{F(\xi, S; \phi_0, \phi_{1,S})}. \tag{17}$$

Remark 5 *Sufficient conditions are given in Subsection 3.2 for (15) to hold with high probability. See Lemma 8, Remarks 10 and 11 and Examples 7, 8, and 9.*

The oracle inequalities in Theorem 4 control both the estimation error in terms of $\phi(\widehat{\beta} - \beta^*)$ and the prediction error in terms of the symmetrized Bregman divergence $\Delta(\widehat{\beta}, \beta^*)$ discussed in Section 2. Since they are based on the GIF (14) in the intersection of the cone and the unit ball $\{b : \phi_0(b) \le 1/e\}$, they are different from typical results in a small-ball analysis based on the Taylor expansion of $\psi(\beta)$ at $\beta = \beta^*$. An important feature of Theorem 4 is that its regularity condition is imposed only on the GIF (14) evaluated at the target β^*; The uniformity of the order of $\Delta(\beta + b, \beta)$ in β is not required. Theorem 4 does allow $\phi_0(\cdot) = 0$ with $F(\xi, S; \phi_0, \phi_0) = \infty$ and $\eta = 0$ in linear regression.

3.1 The Hessian and Related Quantities

In this subsection we describe the relationship between the GIF (14) and the Hessian of the convex function $\psi(\cdot)$ in (1) and examine cases where the quasi star-shaped functions $\phi_0(\cdot)$ and $\phi(\cdot)$ are familiar seminorms. Throughout, we assume that $\psi(\beta)$ is twice differentiable. Let $\ddot{\psi}(\beta)$ be the Hessian of $\psi(\beta)$ and $\Sigma^* = \ddot{\psi}(\beta^*)$.

The GIF (14) can be simplified under the following condition.

Definition 6 *Given a nonnegative-definite matrix Σ and constant $\eta^* > 0$, the symmetriized Bregman divergence $\Delta(\beta, \beta^*)$ satisfies the ϕ_0-relaxed convexity (ϕ_0-RC) condition if*

$$\Delta(\beta^* + b, \beta^*) e^{\phi_0(b)} \geq \langle b, \Sigma b \rangle, \ \forall\, b \in \mathscr{C}(\xi, S), \ \phi_0(b) \leq \eta^*. \tag{18}$$

The ϕ_0-RC condition is related to the restricted strong convexity condition for the Bregman divergence (Negahban et al., 2010): $\ell(\beta^* + b) - \ell(\beta^*) - \langle \dot{\ell}(\beta^*), b \rangle \geq \widetilde{\kappa} \|b\|^2$ with a certain restriction $b \in \mathscr{S}$ and a loss function $\|b\|$. It actually implies the restricted strong convexity of the symmetrized Bregman divergence with $\widetilde{\kappa} = e^{-\eta^*}$ and loss $\|b\|_* = \langle b, \Sigma b \rangle^{1/2}$. However, (18) is used in our analysis mainly to find a quadratic form as a media for the eventual comparison of $\Delta(\beta^* + b, \beta^*)$ with $|b_S|_1 \phi(b)$ in (14), where $\phi(b)$ is the loss function. In fact, in our examples, we find quasi star-shaped functions ϕ_0 for which (18) holds for unrestricted b ($\eta^* = \xi = \infty$). In such cases, the ϕ_0-RC condition is a smoothness condition on the Hessian operator $\ddot{\psi}(\beta) = \ddot{\ell}(\beta)$, since $\Delta(\beta^* + h, \beta^*) = \int_0^1 \langle h, \ddot{\psi}(\beta^* + th) h \rangle dt$ by (4).

In what follows, $\Sigma = \Sigma^* = \ddot{\psi}(\beta^*)$ is allowed in all statements unless otherwise stated. Under the ϕ_0-RC (18), the GIF (14) is bounded from below by the following simple GIF:

$$F_0(\xi, S; \phi) = \inf_{b \in \mathscr{C}(\xi, S)} \frac{\langle b, \Sigma b \rangle}{|b_S|_1 \phi(b)}. \tag{19}$$

In linear regression, $F_0(\xi, S; \phi)$ is the square of the compatibility factor for $\phi(b) = \phi_{1,S}(b) = |b_S|_1/|S|$ (van de Geer, 2007) and the weak cone invertibility factor for $\phi(b) = \phi_q(b) = |b|_q/|S|^{1/q}$ (Ye and Zhang, 2010). They are both closely related to the restricted isometry property (RIP) (Candes and Tao, 2005), the sparse Rieze condition (SRC) (Zhang and Huang, 2008), and the restricted eigenvalue (Bickel et al., 2009). Extensive discussion of these quantities can be found in Bickel et al. (2009), van de Geer and Bühlmann (2009) and Ye and Zhang (2010). The following corollary is an extension of an oracle inequality of Ye and Zhang (2010) from linear regression to the general convex minimization problem (1).

Corollary 7 *Let $\eta \leq \eta^* \leq 1$. Suppose the ϕ_0-RC condition (18). Then, in the event*

$$\Omega_0 \cap \big\{ |w_S|_\infty \lambda + z_0^* \leq \min\big(\xi(\lambda - z_1^*), \eta e^{-\eta} F_0(\xi, S; \phi_0)\big) \big\},$$

the oracle inequalities (16) and (17) in Theorem 4 hold with the GIF $F(\xi, S; \phi_0, \phi)$ replaced by the simple GIF $F_0(\xi, S; \phi)$ in (19). In particular, in the same event,

$$\phi_0(h) \leq \eta, \ |h|_q \leq \frac{e^\eta (|w_S|_\infty \lambda + z_0^*) |S|^{1/q}}{F_0(\xi, S; \phi_q)}, \ \forall\, q > 0,$$

with $\phi_q(b) = |b|_q/|S|^{1/q}$ and $h = \widehat{\beta} - \beta^*$, and with $\phi_{1,S}(b) = |b_S|_1/|S|$,

$$e^{-\eta} \langle h, \Sigma h \rangle \leq \Delta(\widehat{\beta}, \beta^*) \leq \frac{e^\eta (|w_S|_\infty \lambda + z_0^*)^2 |S|}{F_0(\xi, S; \phi_{1,S})} - (\lambda - z_1^*) |W_{S^c} h_{S^c}|_1.$$

Here the only differences between the general model (1) and linear regression ($\phi_0(b) = 0$) are the extra factor e^{η} with $\eta \leq 1$, the extra constraint $|w_S|_\infty \lambda + z_0^* \leq \eta e^{-\eta} F_0(\xi, S; \phi_0)$, and the extra ϕ_0-RC condition (18). Moreover, the simple GIF (19) explicitly expresses all conditions on $F_0(\xi, S; \phi)$ as properties of a fixed matrix Σ.

Example 6 ***(Linear regression: oracle inequalities).*** *For $\psi(\beta) = |Xb|_2^2/(2n)$ and $\Sigma = X'X/n$, $F_0(\xi, S; \phi_q)$ is the weak cone invertibility factor for $q \in [1, \infty]$ (Ye and Zhang, 2010), where a sharper version is defined as the sign restricted invertibility factor (SCIF):*

$$\mathrm{SCIF}_q(\xi, S) = \inf_{b \in \mathscr{C}_-(\xi, S)} |\Sigma b|_\infty / \phi_q(b), \ \phi_q = |b|_q / |S|^{1/q}.$$

For $q = 1$, $F_0^{1/2}(\xi, S; \phi_{1,S})$ is the compatibility constant (van de Geer, 2007)

$$\kappa_*(\xi, S) = \inf_{b \in \mathscr{C}(\xi, S)} \frac{|S|^{1/2} |Xb|_2}{|b_S|_1 n^{1/2}} = \inf_{b \in \mathscr{C}(\xi, S)} \Big(\frac{b' \Sigma b}{|b_S|_1^2 / |S|} \Big)^{1/2}. \tag{20}$$

They are all closely related to the ℓ_2 restricted eigenvalues

$$\mathrm{RE}_2(\xi, S) = \inf_{b \in \mathscr{C}(\xi, S)} \frac{|Xb|_2}{|b|_2 n^{1/2}} = \inf_{b \in \mathscr{C}(\xi, S)} \Big(\frac{b' \Sigma b}{|b|_2^2} \Big)^{1/2}$$

(Bickel et al., 2009; Koltchinskii, 2009). Since $|b_S|_1^2 \leq |b|_2^2 |S|$, $\kappa_*(\xi, S) \geq \mathrm{RE}_2(\xi, S)$ (van de Geer and Bühlmann, 2009). For the Lasso with $\widehat{w}_j = 1$,

$$|\widehat{\beta} - \beta^*|_2 \leq \frac{|S|^{1/2} (\lambda + z_0^*)}{\mathrm{SCIF}_2(\xi, S)} \leq \frac{|S|^{1/2} (\lambda + z_0^*)}{F_0(\xi, S; \phi_2)} \leq \frac{|S|^{1/2} (\lambda + z_0^*)}{\kappa_*(\xi, S) \mathrm{RE}_2(\xi, S)} \tag{21}$$

in the event $\lambda + z_0^* \leq \xi(\lambda - z_1^*)$ (Ye and Zhang, 2010). Thus, cone and general invertibility factors yield sharper ℓ_2 oracle inequalities.

The factors in the oracle inequalities in (21) do not always have the same order for large $|S|$. Although the oracle inequality based on $\mathrm{SCIF}_2(\xi, S)$ is the sharpest among them, it seems not to lead to a simple extension to the general convex minimization in (1). Thus, we settle with extensions of the second sharpest oracle inequality in (21) with $F_0(\xi, S; \cdot)$.

3.2 Oracle Inequalities for the Lasso in GLM

An important special case of the general formulation is the ℓ_1-penalized estimator in a generalized linear model (GLM) (McCullagh and Nelder, 1989). This is Example 3 in Subsection 2.2, where we set up the notation in (13) and gave the KL divergence interpretation to (4). The ℓ_1 penalized, normalized negative likelihood is

$$\ell(\beta) = \psi(\beta) - z'\beta, \ \text{ with } \ \psi(\beta) = \frac{1}{n} \sum_{i=1}^n \big\{ \psi_0(x^i \beta) - c(y_i, \sigma) \big\} \ \text{ and } \ z = \frac{X'y}{n}. \tag{22}$$

Assume that ψ_0 is twice differentiable. Denote the first and second derivatives of ψ_0 by $\dot{\psi}_0$ and $\ddot{\psi}_0$, respectively. The gradient and Hessian are

$$\dot{\psi}(\beta) = X' \dot{\psi}_0(\theta)/n \ \text{ and } \ \ddot{\psi}(\beta) = X' \mathrm{diag}(\ddot{\psi}_0(\theta)) X/n, \tag{23}$$

where $\theta = X\beta$ and $\dot{\psi}_0$ and $\ddot{\psi}_0$ are applied to the individual components of θ.

A crucial condition in our analysis of the Lasso in GLM is the Lipschitz condition

$$\max_{i\le n}\Big|\log\big(\ddot{\psi}_0(x^i\beta^*+t)\big)-\log\big(\ddot{\psi}_0(x^i\beta^*)\big)\Big|\le M_1|t|,\ \forall M_1|t|\le\eta^*, \tag{24}$$

where M_1 and η^* are constants determined by ψ_0. This condition gives

$$\Delta(\beta^*+b,\beta^*)=\int_0^1\langle b,\ddot{\psi}(\beta^*+tb)b\rangle dt\ge\int_0^1\sum_{tM_1|x^ib|\le\eta^*}\frac{\ddot{\psi}_0(x^i\beta^*)(x^ib)^2}{ne^{tM_1|x^ib|}}dt,$$

which implies the following lower bound for the GIF in (14):

$$F(\xi,S;\phi_0,\phi)\ge\inf_{b\in\mathscr{C}(\xi,S),\phi_0(b)\le\eta^*}\sum_{i=1}^n\frac{\ddot{\psi}_0(x^i\beta^*)(x^ib)^2}{n|b_S|_1\phi(b)}\int_0^1 I\{tM_1|x^ib|\le\phi_0(b)\}dt.$$

For seminorms ϕ_0 and ϕ, the infimum above can be taken over $\phi_0(b)=M_2$ due to scale invariance. Thus, for $\phi_0(b)=M_2|b|_2$ and seminorms ϕ, this lower bound is

$$F^*(\xi,S;\phi)=\inf_{b\in\mathscr{C}(\xi,S),|b|_2=1}\sum_{i=1}^n\frac{M_2\ddot{\psi}_0(x^i\beta^*)}{n|b_S|_1\phi(b)}\min\Big(\frac{|x^ib|}{M_1},\frac{(x^ib)^2}{M_2}\Big), \tag{25}$$

due to $(x^ib)^2\int_0^1 I\{tM_1|x^ib|\le M_2\}dt=M_2\min\{|x^ib|/M_1,(x^ib)^2/M_2\}$.

If (24) holds with $\eta^*=\infty$, $\Delta(\beta^*+b,\beta^*)\ge n^{-1}\int_0^1\sum_i\ddot{\psi}_0(x^i\beta^*)(x^ib)^2e^{-tM_1|x^ib|}dt$, so that by the Jensen inequality (18) holds with $\Sigma=\Sigma^*=\ddot{\psi}(\beta^*)$ and

$$\phi_0(b)=\frac{M_1\sum_{i=1}^n\ddot{\psi}_0(x^i\beta^*)|x^ib|^3}{\sum_{i=1}^n\ddot{\psi}_0(x^i\beta^*)(x^ib)^2}\le M_1|Xb|_\infty. \tag{26}$$

This gives a special $F_0(\xi,S;\phi_0)$ as

$$F_*(\xi,S)=\inf_{b\in\mathscr{C}(\xi,S)}\frac{n\langle b,\Sigma^*b\rangle^2/(M_1|b_S|_1)}{\sum_{i=1}^n\ddot{\psi}_0(x^i\beta^*)|x^ib|^3}. \tag{27}$$

Since $|Xb|_\infty\le|X_S|_\infty|b_S|_1+|X_{S^c}W_{S^c}^{-1}|_\infty|W_{S^c}b_{S^c}|\le\{|X_S|_\infty+\xi|X_{S^c}W_{S^c}^{-1}|_\infty\}|b_S|$ in the cone $\mathscr{C}(\xi,S)$ in (9), for $\phi_0(b)=M_3|b_S|_1$ with $M_3=M_1\{|X_S|_\infty+\xi|X_{S^c}W_{S^c}^{-1}|_\infty\}$, the ϕ_0-RC condition (18) automatically implies the stronger

$$e^{-\phi_0(b)}\langle b,\Sigma^*b\rangle\le\Delta(\beta^*+b,\beta^*)\le e^{\phi_0(b)}\langle b,\Sigma^*b\rangle,\ \forall\, b\in\mathscr{C}(\xi,S),\ \phi_0(b)<\infty. \tag{28}$$

Under the Lipschitz condition (24), we may also use the following large deviation inequalities to find explicit penalty levels to guarantee the noise bound (15).

Lemma 8 *(i) Suppose the model conditions (13) and (24) with certain $\{M_1,\eta^*\}$. Let x_j be the columns of X, Σ^*_{ij} be the elements of $\Sigma^*=\ddot{\psi}(\beta^*)$. For penalty levels $\{\lambda_0,\lambda_1\}$ define $t_j=\lambda_0 I\{j\in S\}+w_j\lambda_1 I\{j\notin S\}$. Suppose the bounds w_j in (6) are deterministic and*

$$M_1\max_{j\le p}\big(|x_j|_\infty|t_j/\Sigma^*_{jj}\big)\le\eta_0e^{\eta_0}\ \ \textit{and}\ \ \sum_{j=1}^p\exp\Big\{-\frac{nt_j^2e^{-\eta_0}}{2\sigma^2\Sigma^*_{jj}}\Big\}\le\frac{\varepsilon_0}{2} \tag{29}$$

for certain constants $\eta_0 \le \eta^*$ *and* $\varepsilon_0 > 0$. *Then,* $\mathrm{P}_{\beta^*}\Big\{z_0^* \le \lambda_0, z_1^* \le \lambda_1\Big\} \ge 1-\varepsilon_0$.
(ii) If $c_0 = \max_t \ddot{\psi}(t)$, *then part (i) is still valid if (24) and (29) are replaced by*

$$\sum_{j=1}^{p} \exp\Big\{-\frac{n^2 t_j^2}{2\sigma^2 c_0 |x_j|_2^2}\Big\} \le \frac{\varepsilon_0}{2}. \tag{30}$$

In particular, if $|x_j|_2^2 = n, 1 \le j \le p, w_j = 1, j \notin S$ *and* $\lambda_0 = \lambda_1 = \lambda$ *(so* $t_j = \lambda$*), then part (i) still holds if* $\lambda \ge \sigma\sqrt{(2c_0/n)\log(2p/\varepsilon_0)}$.

The following theorem is a consequence of Theorem 4, Corollary 7 and Lemma 8.

Theorem 9 *(i) Let* $\widehat{\beta}$ *be the weighted Lasso estimator in (2) with GLM loss function in (22). Let* β^* *be a target vector and* $h = \widehat{\beta} - \beta^*$. *Suppose that the data follows the GML model (13) satisfying the Lipschitz condition (24) with certain* $\{M_1, \eta^*\}$. *Let* $F^*(\xi, S; \phi)$ *be as in (25) with* $S \supseteq \{j : \beta_j^* \ne 0\}$ *and a constant* M_2. *Let* $\eta \le 1 \wedge \eta^*$ *and* $\{\lambda, \lambda_0, \lambda_1\}$ *satisfy*

$$|w_S|_\infty \lambda + \lambda_0 \le \min\big\{\xi(\lambda - \lambda_1), \eta e^{-\eta} F^*(\xi, S; M_2|\cdot|_2)\big\}. \tag{31}$$

Then, in the event $\Omega_0 \cap \big\{\max_{k=0,1}\big(z_k^*/\lambda_k\big) \le 1\big\}$ *with the* z_k^* *in (5) and* Ω_0 *in (6),*

$$\Delta(\beta^* + h, \beta^*) \le \frac{e^\eta(|w_S|_\infty\lambda + \lambda_0)^2|S|}{F^*(\xi, S; \phi_{1,S})}, \quad \phi(h) \le \frac{e^\eta(|w_S|_\infty\lambda + \lambda_0)}{F^*(\xi, S; \phi)} \tag{32}$$

for any seminorm ϕ *as the estimation loss. In particular, for* $\phi(b) = M_2|b|_2$, *(32) gives* $|h|_2 \le \eta/M_2$. *Moreover, if either (29) or (30) holds for the penalty level* $\{\lambda_0, \lambda_1\}$ *and the weight bounds* w_j *in (6) are deterministic, then*

$$\mathrm{P}_{\beta^*}\big\{\textit{(32) holds for all seminorms } \phi\big\} \ge \mathrm{P}_{\beta^*}(\Omega_0) - \varepsilon_0.$$

(ii) Suppose $\eta^* = \infty$ *and (31) holds with* $F^*(\xi, S; M_2|\cdot|_2)$ *replaced by the special simple GIF* $F_*(\xi, S)$ *in (27) for the* ϕ_0 *in (26). Then, the conclusions of part (i) hold with* $F^*(\xi, S; \cdot)$ *replaced by the simple GIF* $F_0(\xi, S; \cdot)$ *in (19). Moreover,* $\phi_0(h) \le \eta$ *and (32) can be strengthened with the lower bound* $\Delta(\beta^* + h, \beta^*) \ge e^{-\eta}\langle h, \Sigma^* h\rangle$.
(iii) For any $\eta^* > 0$, *the conclusions of part (ii) hold for the* $\phi_0(b) = M_3|b_S|_1$ *in (28), if* $F_*(\xi, S)$ *is replaced by* $\kappa_*^2(\xi, S)/(M_3|S|)$ *in (31), where* $\kappa_*(\xi, S)$ *is the compatibility constant in (20).*

Remark 10 *If either (29) or (30) holds for the penalty levels* $\{\lambda_0, \lambda_1\}$ *and the bounds* w_j *in (6) are deterministic, then (32) implies* $\mathrm{P}_{\beta^*}\{\textit{the noise bound (15) holds}\} \ge \mathrm{P}_{\beta^*}(\Omega_0) - \varepsilon_0$.

Remark 11 *Suppose that* $\max_{j\notin S} 1/w_j$, $\max_j 1/\Sigma_{jj}^*$, $\max_{j\in S} w_j$, $\max_j \Sigma_{jj}^*$, *and* M_1 *are all bounded, and that* $\{1 + F_*^2(\xi, S)\}(\log p)/n \to 0$. *Then, (29) holds with the penalty level* $\lambda_0 = \lambda_1 = a\sigma\sqrt{(2/n)\log(p/\varepsilon_0)}$ *for certain* $a \le (1 + o(1))\max_j(\Sigma^*)_{jj}^{1/2}/w_j$, *due to* $\max\{\lambda_0, \eta, \eta_0\} \to 0+$. *Again, the conditions and conclusions of Theorem 9 "converge" to those for the linear regression as if the Gram matrix is* Σ^*.

Remark 12 *In Theorem 9, the key condition (31) is weaker in parts (i) and (ii) than part (iii), although part (ii) requires* $\eta^* = \infty$. *For* $\Sigma = \Sigma^*$ *and* $M_1 = M_2 \le M_3/(1+\xi)$,

$$\kappa_*^2(\xi,S)/(M_3|S|) \le \min\Big\{F_*(\xi,S), F^*(\xi,S;M_2|\cdot|_2)\Big\},$$

since $n^{-1}\sum_{i=1}^n \ddot\psi_0(x^i\beta^*)|x^ib|^3/\langle b,\Sigma^* b\rangle \le |Xb|_\infty \le |b_S|_1 M_3/M_1$ *as in the derivation of (28) and* $|b|_2 \le |b|_1 \le (1+\xi)|b_S|_1$ *in the cone (9). For the more familiar* $\kappa_*^2(\xi,S)/(M_3|S|)$ *with the compatibility constant, (31) essentially requires a small* $|S|\sqrt{(\log p)/n}$. *The sharper Theorem 9 (i) and (ii) provides conditions to relax the requirement to a small* $|S|(\log p)/n$.

Remark 13 *For* $\widehat w_j = 1$, *Negahban et al. (2010) considered M-estimators under the restricted strong convexity condition discussed below Definition 6. For the GLM, they considered iid sub-Gaussian* x^i *and used empirical process theory to bound the ratio* $\Delta(\beta^*+b,\beta^*)/\{|b|_2(|b|_2 - c_0|b|_1\}$ *from below over the cone (9) with a small* c_0. *Their result extends the* ℓ_2 *error bound* $|S|^{1/2}(\lambda + z_0^*)/\mathrm{RE}_2^2(\xi,S)$ *of Bickel et al. (2009), while Theorem 9 extends the sharper (21) with the factor* $F_0(\xi,S;\phi_2)$. *Theorem 9 applies to both deterministic and random designs. Similar to Negahban et al. (2010), for iid sub-Gaussian* x^i, *empirical process theory can be applied to the lower bound (25) for the GIF to verify the key condition (31) with* $F^*(\xi,S;M_2|\cdot|_2) \gtrsim |S|^{-1/2}$, *provided that* $|S|(\log p)/n$ *is small.*

Example 7 ***(Linear regression: oracle inequalities, continuation)*** *For the linear regression model (10) with quadratic loss,* $\psi_0(\theta) = \theta^2/2$, *so that (24) holds with* $M_1 = 0$ *and* $\eta^* = \infty$. *It follows that* $F^*(\xi,S;M_2|\cdot|_2) = \infty$ *and (31) has the interpretation with* $\eta = 0+$ *and* $\eta e^{-\eta}F^*(\xi,S;M_2|\cdot|_2) = \infty$. *Moreover, since* $M_1 = 0$, $\eta_0 = 0+$ *in (29). Thus, the conditions and conclusions of Theorem 9 "converge" to the case of linear regression as* $M_1 \to 0+$. *Suppose iid* $\varepsilon_i \sim N(0,\sigma^2)$ *as in (13). For* $\widehat w_j = w_j = 1$ *and* $\Sigma_{jj}^* = \sum_{i=1}^n x_{ij}^2/n = 1$, *(29) holds with* $\lambda_0 = \lambda_1 = \sigma\sqrt{(2/n)\log(p/\varepsilon_0)}$ *and (31) holds with* $\lambda = \lambda_0(1+\xi)/(1-\xi)$. *The value of* σ *can be estimated iteratively using the mean residual squares (Städler et al., 2010; Sun and Zhang, 2011). Alternatively, cross-validation can be used to pick* λ. *For* $\phi(b) = \phi_2(b) = |b|_2/|S|^{1/2}$, *(32) matches the risk bound in (21) with the factor* $F_0(\xi,S;\phi_2)$.

Example 8 ***(Logistic regression: oracle inequalities)*** *The model and loss function are given in (11) and (12) respectively. Here we verify the conditions of Theorem 9. The Lipschitz condition (24) holds with* $M_1 = 1$ *and* $\eta^* = \infty$ *since* $\psi_0(t) = \log(1+e^t)$ *provides*

$$\frac{\ddot\psi_0(\theta+t)}{\ddot\psi_0(\theta)} = \frac{e^t(1+e^\theta)^2}{(1+e^{\theta+t})^2} \ge \begin{cases} e^{-|t|} & t<0 \\ e^{-t}(1+e^\theta)^2/(e^{-t}+e^\theta)^2 \ge e^{-|t|} & t>0. \end{cases}$$

Since $\max_t \ddot\psi(t) = c_0 = 1/4$ *we can apply (30). In particular, if* $\widehat w_j = w_j = 1 = |x_j|_2^2/n$, $\lambda = \{(\xi+1)/(\xi-1)\}\sqrt{(\log(p/\varepsilon_0))/(2n)}$ *and* $\lambda\{2\xi/(\xi+1)\}/F_*(\xi,S) \le \eta e^{-\eta}$, *then (32) holds with at least probability* $1-\varepsilon_0$ *under* P_{β^*}. *For such deterministic* $\widehat W$ *and* X, *an adaptive choice of the penalty level is* $\lambda = \widehat\sigma\sqrt{(2/n)\log p}$ *with* $\widehat\sigma^2 = \sum_{i=1}^n \pi_i(\widehat\beta)\{1-\pi_i(\widehat\beta)\}/n$, *where* $\pi_i(\beta)$ *is as in Example 2.*

Example 9 ***(Log-linear models: oracle inequalities)*** *Consider counting data with* $y_i \in \{0,1,2,\ldots\}$. *In log-linear models, it is assume that*

$$\mathrm{E}_\beta(y_i) = e^{\theta_i},\ \theta_i = x^i\beta,\ 1 \le i \le n.$$

This becomes a GLM with the average negative Poisson log-likelihood function

$$\ell(\beta) = \psi(\beta) - z'\beta,\ \psi(\beta) = \sum_{i=1}^{n} \frac{\exp(x^i\beta) - \log(y_i!)}{n},\ z = X'y/n.$$

In this model, $\psi_0(t) = e^t$*, so that the Lipschitz condition (24) holds with* $M_1 = 1$ *and* $\eta^* = \infty$*. Although (30) is not useful with* $c_0 = \infty$*, (29) can be used in Theorem 9.*

4. Adaptive and Multistage Methods

We consider in this section an adaptive Lasso and its repeated applications, with weights recursively generated from a concave penalty function. This approach appears to provide the most appealing choice of weights both from heuristic and theoretical standpoints. The analysis here uses the results in Section 3 and an idea in Zhang (2010b).

We first consider adaptive Lasso and provide conditions under which it improves upon its initial estimator. Let $\rho_\lambda(t)$ be a concave penalty function with $\dot\rho_\lambda(0+) = \lambda$, where $\dot\rho_\lambda(t) = (\partial/\partial t)\rho_\lambda(t)$. The maximum concavity of the penalty is

$$\kappa = \sup_{0<t_1<t_2} \frac{|\dot\rho_\lambda(t_2) - \dot\rho_\lambda(t_1)|}{t_2 - t_1}. \tag{33}$$

Let $\mathscr{C}(\xi, S)$ be the cone in (9). Let $\phi_0(b)$ be a quasi star-shaped function and define

$$F_2(\xi, S; \phi_0) = \inf\Big\{ \frac{e^{\phi_0(b)}\Delta(\beta^* + b, \beta^*)}{|b_S|_2|b|_2} : 0 \neq b \in \mathscr{C}(\xi, S), \phi_0(b) \le \eta^* \Big\}. \tag{34}$$

This quantity is an ℓ_2 version of the GIF in (14). The analysis in Section 3 can be used to find lower bounds for (34) in the same way simply by taking $\phi(b) = |b|_2$ and replacing $|b_S|_1$ with $|b_S|_2$. For example, in generalized linear models (13) satisfying the Lipschitz condition (24), the derivation of (25) yields

$$F_2(\xi, S; M|\cdot|_2) \ge \inf_{b\in\mathscr{C}(\xi,S),|b|_2=1} \sum_{i=1}^{n} \frac{M_2\ddot\psi_0(x^i\beta^*)}{n|b_S|_2} \min\Big(\frac{|x^ib|}{M_1}, \frac{(x^ib)^2}{M_2}\Big).$$

Given $0 < \varepsilon_0 < 1$, the components of the error vector $z - \psi(\beta^*)$ are sub-Gaussian if for all $0 \le t \le \sigma\sqrt{(2/n)\log(4p/\varepsilon_0)}$,

$$\mathrm{P}_{\beta^*}\Big\{ |(z - \dot\psi(\beta^*))_j| \ge t \Big\} \le 2e^{-nt^2/(2\sigma^2)}. \tag{35}$$

This condition holds for all GLM when the components of $X\beta^*$ are uniformly in the interior of the natural parameter space for the exponential family.

Theorem 14 *Let* κ *be as in (33),* $S_0 = \{j : \beta^*_j \neq 0\}$*,* $\lambda_0 > 0$*,* $0 < \eta < 1$*,* $0 < \gamma_0 < 1/\kappa$*,* $A > 1$*, and* $\xi \ge (A + 1 - \kappa\gamma_0)/(A - 1)$*. Let* ϕ_0 *be a quasi star-shaped function,* $F(\xi, S; \phi_0, \phi_0)$ *be the GIF in (14), and* $F_2(\xi, S; \phi_0)$ *its* ℓ_2*-version in (34). Suppose*

$$\lambda_0\{1 + A/(1 - \kappa\gamma_0)\} \le F(\xi, S; \phi_0, \phi_0)\eta e^{-\eta},\ F_* \le F_2(\xi, S; \phi_0), \tag{36}$$

for all $S \supseteq S_0$ *with* $|S \setminus S_0| \le \ell^*$. *Let* $\widetilde{\beta}$ *be an initial estimator of* β *and* $\widehat{\beta}$ *be the weighted Lasso in (2) with weights* $\widehat{w}_j = \dot{\rho}_\lambda(|\widetilde{\beta}_j|)/\lambda$ *and penalty level* $\lambda = A\lambda_0/(1-\kappa\gamma_0)$. *Then,*

$$|\widehat{\beta}-\beta^*|_2 \le \frac{e^\eta}{F_*}\Big\{|\dot{\rho}_\lambda(|\beta^*_{S_0}|)|_2 + |\{z-\dot{\psi}(\beta^*)\}_{S_0}|_2 + \Big(\kappa + \frac{1}{\gamma_0 A} - \frac{\kappa}{A}\Big)|\widetilde{\beta}-\beta^*|_2\Big\}$$

in the event $\{|(\widetilde{\beta}-\beta)_{S_0^c}|_2^2 \le \gamma_0^2\lambda^2\ell^*\} \cap \{|z-\dot{\psi}(\beta^*)|_\infty \le \lambda_0\}$. *Moreover, if (35) holds and* $\lambda_0 = \sigma\sqrt{(2/n)\log(2p/\varepsilon_0)}$ *with* $0 < \varepsilon_0 < 1$, *then* $\mathrm{P}_{\beta^*}\{|z-\dot{\psi}(\beta^*)| \ge \lambda_0\} \le \varepsilon_0$.

Theorem 14 raises the possibility that $\widehat{\beta}$ improves $\widetilde{\beta}$ under proper conditions. Thus it is desirable to repeatedly apply this adaptive Lasso in the following way,

$$\widehat{\beta}^{(k+1)} = \arg\min_\beta \Big\{\ell(\beta) + \sum_{j=1}^p \dot{\rho}_\lambda(\widehat{\beta}_j^{(k)})|\beta_j|\Big\},\ k = 0,1,\ldots. \tag{37}$$

Such multistage algorithms have been considered in the literature (Fan and Li, 2001; Zou and Li, 2008; Zhang, 2010b). As discussed in Remark 16 below, it is beneficial to use a concave penalty ρ_λ in (37). Natural choices of ρ_λ include the smoothly clipped absolute deviation and minimax concave penalties (Fan and Li, 2001; Zhang, 2010a).

Theorem 15 *Let* $\{\kappa, S_0, \lambda_0, \eta, \gamma_0, A, \xi, \ell^*, \lambda\}$ *be the same as Theorem 14. Let* $\widehat{\beta}^{(0)}$ *be the unweighted Lasso with* $\widehat{w}_j = 1$ *in (2) and* $\widehat{\beta}^{(\ell)}$ *be the* ℓ*-th iteration of the recursion (37) initialized with* $\widehat{\beta}^{(0)}$. *Let* $\xi_0 = (\lambda+\lambda_0)/(\lambda-\lambda_0)$. *Suppose (36) holds and*

$$e^\eta\{1+(1-\kappa\gamma_0)/A\}/F(\xi_0, S_0; \phi_0, |\cdot|_2) \le \gamma_0\sqrt{\ell^*}. \tag{38}$$

Define $r_0 = (e^\eta/F_*)\{\kappa + 1/(\gamma_0 A) - \kappa/A\}$. *Suppose* $r_0 < 1$. *Then,*

$$|\widehat{\beta}^{(\ell)}-\beta^*|_2 \le \frac{|\dot{\rho}_\lambda(|\beta^*_{S_0}|)|_2 + |\{z-\dot{\psi}(\beta^*)\}_{S_0}|_2}{e^{-\eta}F_*(1-r_0)/(1-r_0^\ell)} + \frac{r_0^\ell e^\eta\lambda\{1+(1-\kappa\gamma_0)/A\}}{F(\xi_0, S_0; \phi_0, |\cdot|_2)} \tag{39}$$

in the event

$$\Big\{|z-\dot{\psi}(\beta^*)|_\infty \le \lambda_0\Big\} \cap \Big\{\frac{|\dot{\rho}_\lambda(|\beta^*_{S_0}|)|_2 + |\{z-\dot{\psi}(\beta^*)\}_{S_0}|_2}{e^{-\eta}F_*(1-r_0)} \le \gamma_0\lambda\sqrt{\ell^*}\Big\}. \tag{40}$$

Moreover, if (35) holds and $\lambda_0 = \sigma\sqrt{(2/n)\log(4p/\varepsilon_0)}$ *with* $0 < \varepsilon_0 < 1$, *then the intersection of the events (40) and* $\{|\{z-\dot{\psi}(\beta^*)\}_{S_0}|_2 \le n^{-1/2}\sigma\sqrt{2|S_0|\log(4|S_0|/\varepsilon_0)}\}$ *happens with at least* P_{β^*} *probability* $1-\varepsilon_0$, *provided that*

$$\frac{|\dot{\rho}_\lambda(|\beta^*_{S_0}|)|_2 + n^{-1/2}\sigma\sqrt{2|S_0|\log(4|S_0|/\varepsilon_0)\}}}{e^{-\eta}F_*(1-r_0)} \le \frac{\gamma_0 A\lambda_0\sqrt{\ell^*}}{1-\kappa\gamma_0}.$$

Remark 16 *Define* $R^{(0)} = \lambda e^\eta\{1+(1-\kappa\gamma_0)/A\}/F(\xi_0, S_0; \phi_0, |\cdot|_2)$ *and*

$$R^{(\infty)} = \frac{|\dot{\rho}_\lambda(|\beta^*_{S_0}|)|_2 + |\{z-\dot{\psi}(\beta^*)\}_{S_0}|_2}{e^{-\eta}F_*(1-r_0)},\ R^{(\ell)} = (1-r_0^\ell)R^{(\infty)} + r_0^\ell R^{(0)}.$$

It follows from (39) that $R^{(\ell)}$ is an upper bound of $|\widehat{\beta}^{(\ell)}-\beta^*|_2$ under proper conditions. This implies $|\widehat{\beta}^{(\ell)}-\beta^*| \le 2R^{(\infty)}$ after $\ell = |\log r_0|^{-1}\log(R^{(\infty)}/R^{(0)})$ iterations of the recursion (37). Under condition (35),

$$\mathrm{E}_{\beta^*}R^{(\infty)} \le \{|\dot\rho_\lambda(|\beta^*_{S_0}|)|_2 + 2\sigma\sqrt{|S_0|/n}\}e^\eta/\{F_*(1-r_0)\}.$$

Since $\rho_\lambda(t)$ is concave in t, $|\dot\rho_\lambda(|\beta^*_{S_0}|)|_2 \le \dot\rho_\lambda(0+)|S_0|^{1/2} = \lambda|S_0|^{1/2}$. This component of $\mathrm{E}_{\beta^*}R^{(\infty)}$ matches the noise inflation due to model selection uncertainty since $\lambda \asymp \lambda_0 = \sigma\sqrt{(2/n)\log(p/\varepsilon_0)}$. This noise inflation diminishes when $\min_{j\in S_0}|\beta^*_j| \ge \gamma\lambda$ and $\dot\rho_\lambda(t)=0$ for $|t|\ge\gamma\lambda$, yielding the super-efficient $\mathrm{E}_{\beta^*}R^{(\infty)} \le \{2\sigma\sqrt{|S_0|/n}\}e^\eta/\{F_*(1-r_0)\}$ without the $\log p$ factor. The risk bound $R^{(\infty)}$ is comparable with those for concave penalized least squares in linear regression (Zhang, 2010a).

Remark 17 *For $\log(p/n) \asymp \log p$, the penalty level λ in Theorems 14 and 15 are comparable with the best proven results and of the smallest possible order in linear regression. For $\log(p/n) \ll \log p$, the proper penalty level is expected to be of the order $\sigma\sqrt{(2/n)\log(p/|S_0|)}$ under a vectorized sub-Gaussian condition which is slightly stronger than (35). This refinement for $\log(p/n) \ll \log p$ is beyond the scope of this paper.*

5. Selection Consistency

In this section, we provide a selection consistency theorem for the ℓ_1 penalized convex minimization estimator, including both the weighted and unweighted cases. Let $\|M\|_\infty = \max_{|u|_\infty\le 1}|Mu|_\infty$ be the ℓ_∞-to-ℓ_∞ operator norm of a matrix M.

Theorem 18 *Let $\ddot\psi(\beta) = \ddot\ell(\beta)$ be the Hessain of the loss in (1), $\widehat\beta$ be as in (2), β^* a target vector, z^*_k be as in (5), Ω_0 in (6), $S \supseteq \{j:\beta^*_j\ne 0\}$ and $F(\xi,S;\phi_0,\phi)$ as in (14).*
(i) Let $0<\eta\le\eta^*\le 1$, $\mathscr{B}^*_0 = \{\beta:\phi_0(\beta-\beta^*)\le\eta, \beta_{S^c}=0\}$ and $S_\beta = \{j:\beta_j\ne 0\}$. Suppose

$$\sup_{\beta\in\mathscr{B}^*_0}\big|\widehat W^{-1}_{S^c}\ddot\psi_{S^c,S_\beta}(\beta)\{\ddot\psi_{S_\beta}(\beta)\}^{-1}\widehat W_{S_\beta}\mathrm{sgn}(\beta_{S_\beta})\big|_\infty \le \kappa_0 < 1 \tag{41}$$

$$\sup_{\beta\in\mathscr{B}^*_0}\big\|\widehat W^{-1}_{S^c}\ddot\psi_{S^c,S_\beta}(\beta)\{\ddot\psi_{S_\beta}(\beta)\}^{-1}\big\|_\infty \le \kappa_1. \tag{42}$$

Then, $\{j:\widehat\beta_j\ne 0\}\subseteq S$ in the event

$$\Omega^*_1 = \Omega_0\cap\Big\{|\widehat w_S|_\infty\lambda + z^*_0 \le \eta e^{-\eta}F(0,S;\phi_0,\phi_0),\ \kappa_1 z^*_0 + z^*_1 < (1-\kappa_0)\lambda\Big\}. \tag{43}$$

(ii) Let $0<\eta\le\eta^*\le 1$ and $\mathscr{B}_0 = \{\beta:\phi_0(\beta-\beta^*)\le\eta, \mathrm{sgn}(\beta)=\mathrm{sgn}(\beta^*)\}$. Suppose (41) and (42) hold with $\mathscr{B}^*_0$ replaced by $\mathscr{B}_0$. Then, $\mathrm{sgn}(\widehat\beta)=\mathrm{sgn}(\beta^*)$ in the event

$$\Omega^*_1\cap\Big\{\sup_{\beta\in\mathscr{B}_0}\big\|\{\ddot\psi_S(\beta)\}^{-1}\big\|_\infty\big(|\widehat w_S|_\infty\lambda + z^*_0\big) < \min_{j\in S}|\beta^*_j|\Big\}. \tag{44}$$

(iii) Suppose conditions of Theorem 9 hold for the GLM. Then, the conclusions of (i) and (ii) hold under the respective conditions if $F(0,S;\phi_0,\phi_0)$ is replaced by $F^*(\xi,S;M_2|\cdot|_2)$ or $F_*(\xi,S)$ or $\kappa^2_*(\xi,S)/(M_3|S|)$ with the respective ϕ_0 in Theorem 9.

For $\widehat{w}_j = 1$, this result is somewhat more specific in the radius η for the uniform irrepresentable condition (41), compared with a similar extension of the selection consistency theory to the graphical Lasso by Ravikumar et al. (2008). In linear regression (10), $\ddot{\psi}(\beta) = \Sigma = X'X/n$ does not depend on β, so that Theorem 18 with the special $w_j = 1$ matches the existing selection consistency theory for the unweighted Lasso (Meinshausen and Bühlmann, 2006; Tropp, 2006; Zhao and Yu, 2006; Wainwright, 2009). We discuss below the ℓ_1 penalized logistic regression as a specific example.

Example 10 ***(Logistic regression: selection consistency)*** *Suppose $w_j = 1 = |x_j|_2^2/n$ where x_j are the columns of X. If (43) and (44) hold with z_0^* and z_1^* replaced by $\sqrt{(\log(p/\varepsilon_0))/(2n)}$, then the respective conclusions of Theorem 18 hold with at least probability $1-\varepsilon_0$ in P_{β^*}.*

6. The Sparsity of the Lasso and SRC

The results in Sections 2, 3, and 4 are concerned with prediction and estimation properties of $\widehat{\beta}$, but not dimension reduction. Theorem 18 (i) and (iii) provide dimension reduction under ℓ_∞-type conditions (41) and (42). In this section, we provide upper bounds for the dimension of $\widehat{\beta}$ under conditions of a weaker ℓ_2 type. For this purpose, we introduce

$$\kappa_+(m) = \sup_{|B|=m} \Big\{ \lambda_{\max}\Big(W_B^{-2} \int_0^1 \ddot{\psi}_B(\beta^* + tb)dt \Big) : B \cap S = \emptyset, b \in \mathscr{C}(\xi, S), \phi_0(b) \le \eta^* \Big\} \tag{45}$$

as a restricted upper eigenvalue, where $\lambda_{\max}(M)$ is the largest eigenvalue of matrix M, $B \subseteq \{1,\ldots,p\}$, and $\ddot{\psi}_B(\beta)$ and W_B are the restrictions of the Hessian of (1) and the weight operator $W = \mathrm{diag}(w_1,\ldots,w_p)$ to $\mathbb{R}^B$.

Theorem 19 *Let β^* be a target vector, $S \supseteq \{j : \beta_j^* \neq 0\}$, $\widehat{\beta}$ be the weighted Lasso estimator (2), and z_k^* be the ℓ_∞-noise level as in (5). Let $0 \le \eta^* \le 1$, $\phi_{1,S}(b) = |b_S|_1/|S|$, ϕ_0 be a quasi star-shaped function, and $F(\xi, S; \phi_0, \phi)$ be the GIF in (14). Then, in the event (15),*

$$\#\{j : \widehat{\beta}_j \neq 0, j \notin S\} < d_1 = \min\Big\{ m \ge 1 : \frac{m}{\kappa_+(m)} > \frac{e^\eta \xi^2 |S|}{F(\xi, S; \phi_0, \phi_{1,S})} \Big\}.$$

It follows from the Cauchy-Schwarz inequality that $\kappa_+(m)$ is sub-additive, $\kappa_+(m_1+m_2) \le \kappa_+(m_1) + \kappa_+(m_2)$, so that $m/\kappa_+(m)$ is non-decreasing in m. For GLM, lower bounds for the GIF and probability upper bounds for z_k^* can be found in Subsection 3.2. For $S = \{j : \beta_j^* \neq 0\}$. Theorem 19 gives an upper bound for the false negative.

In linear regression, upper bounds for the false negative of the Lasso or concave penalized LSE can be found in Zhang and Huang (2008) and Zhang (2010a) under a sparse Riesz condition (SRC). We now extend their results to the Lasso for the more general convex minimization problem (1). For this purpose, we strengthen (18) to

$$e^{-\phi_0(b)}\Sigma^* \le \ddot{\psi}(\beta^* + b) \le e^{\phi_0(b)}\Sigma^*, \quad \forall\, b \in \mathscr{C}(\xi, S),\ \phi_0(b) \le \eta^*, \tag{46}$$

and assume the following SRC: for certain constants $\{c_*, c^*\}$, integer d^*, $0 < \alpha < 1$, $0 < \eta \le \eta^* \le 1$, all $A \supset S$ with $|A| = d^*$, and all $u \in \mathbb{R}^A$ with $|u| = 1$,

$$c_* \le \langle u, \ddot{\psi}_A(\beta^*)u \rangle \le c^*,\ \frac{|S|}{2(1-\alpha)}\Big(\frac{e^{2\eta}c^*}{c_*} + 1 - 2\alpha \Big) \le d^*. \tag{47}$$

Theorem 20 *Let $\widehat{\beta}$ be the Lasso estimator (2) with $w_j = 1$ for all j, β^* be a target vector, $S \supseteq \{j : \beta^*_j \neq 0\}$, and z^*_k be the ℓ_∞-noise level as in (5). Let ϕ_0 be a quasi star-shaped function, and $F(\xi, S; \phi_0, \phi)$ be the GIF in (14). Suppose (46) and (47) hold. Let d_1 be the integer satisfying $d_1 - 1 \leq |S|(e^{2\eta}c^*/c_* - 1)/(2-2\alpha) < d_1$. Then,*

$$\#\{j : \widehat{\beta}_j \neq 0, j \notin S\} < d_1$$

when $z^*_0 + \xi z^*_1 \leq (\xi - 1)\lambda$, $\lambda + z^*_0 \leq \eta e^{-\eta} F(\xi, S; \phi_0, \phi_0)$, and

$$\max_{A \supset S, |A| \leq d_1} |(\Sigma^*)^{-1/2}_A \dot{\ell}_A(\beta^*)|_2 \leq e^{-\eta}\alpha\lambda\sqrt{d_1/c^*}.$$

Theorems 19 and 20 use different sets of conditions to derive dimension bounds since different analytical approaches are used. These sets of conditions do not imply each other. In the most optimistic case, the SRC (47) allows $d^* = d_1 + |S|$ to be arbitrary close to $|S|$ when $e^{2\eta}c^*/c_* \approx 1$, while Theorem 19 requires $d_1 \geq |S|$ when $\kappa_+(m) \geq 1$ and $F(\xi, S; \phi_0, \phi_{1,S}) \leq 1$ (always true for Σ^* with 1 in the diagonal).

7. Discussion

In this paper, we studied the estimation, prediction, selection and sparsity properties of the weighted and adaptive ℓ_1-penalized estimators in a general convex loss formulation. We also studied concave regularization in the form of recursive application of adaptive ℓ_1-penalized estimators.

We applied our general results to several important statistical models, including linear regression and generalized linear models. For linear regression, we extend the existing results to weighted and adaptive Lasso. For the GLMs, the $\ell_q, q \geq 1$ error bounds for a general $q \geq 1$ for the GLMs are not available in the literature, although ℓ_1 and ℓ_2 bounds have been obtained under different sets of conditions respectively in van de Geer (2008) and]citeNegahbanRWY10. Our fixed-sample analysis provides explicit definition of constant factors in an explicit neighborhood of a target. Our oracle inequalities yields even sharper results for multistage recursive application of adaptive Lasso based on a suitable concave penalty. The results on the sparsity of the solution to the ℓ_1-penalized convex minimization problem is based on a new approach.

An interesting aspect of the approach taken in this paper in dealing with general convex losses such as those for the GLM is that the conditions imposed on the Hessian naturally "converge" to those for the linear regression as the convex loss "converges" to a quadratic form.

A key quantity used in the derivation of the results is the generalized invertibility factor (14), which grow out of the idea of the ℓ_2 restricted eigenvalue but improves upon it. The use of GIF yields sharper bounds on the estimation and prediction errors. This was discussed in detail in the context of linear regression in Ye and Zhang (2010).

We assume that the convex function $\psi(\cdot)$ is twice differentiable. Although this assumption is satisfied in many important and widely used statistical models, it would be interesting to extend the results obtained in this paper to models with less smooth loss functions, such as those in quantile regression and support vector machine.

Acknowledgments

The work of Jian Huang is supported in part by the National Institutes of Health (NIH Grants R01CA120988 and R01CA142774) and the National Science Foundation (NSF Grant DMS-08-05670). The work of Cun-Hui Zhang is supported in part by the National Science Foundation (NSF Grants DMS-0906420 and DMS-1106753) and the National Security Agency (NSA Grant H98230-11-1-0205).

Appendix A.

Proof of Lemma 1. Since $\dot\psi(\widehat\beta)-\dot\psi(\beta^*)=z-\dot\psi(\beta^*)-g$, (3) implies

$$\Delta(\widehat\beta,\beta^*)=\big\langle\widehat\beta,z-\dot\psi(\beta^*)\big\rangle-\lambda|\widehat W\widehat\beta|_1-\big\langle\beta^*,z-\dot\psi(\beta^*)-g\big\rangle$$

and $|g_j|\le\widehat w_j\lambda$. Thus, (7) follows from $|(z-\dot\psi(\beta^*)_j|\le\widehat w_j\lambda$ and $\widehat w_j\le w_j$ in S in Ω_0.

For (8), we have $h_{S^c}=\widehat\beta_{S^c}$ and $\beta^*_{S^c}=0$, so that in Ω_0 (3) gives

$$\begin{aligned}\Delta(\widehat\beta,\beta^*) &= \big\langle\widehat\beta_{S^c},\{z-\dot\psi(\beta^*)\}_{S^c}\big\rangle-\lambda|\widehat W_{S^c}\widehat\beta_{S^c}|_1-\big\langle h_S,\{z-\dot\psi(\beta^*)-g\}_S\big\rangle\\ &\le |W_{S^c}\widehat\beta_{S^c}|_1(z_1^*-\lambda)+\big\langle h_S,g_S-\{z-\dot\psi(\beta^*)\}_S\big\rangle\\ &\le |W_{S^c}\widehat\beta_{S^c}|_1(z_1^*-\lambda)+|h_S|_1(z_0^*+|w_S|_\infty\lambda).\end{aligned}$$

This gives (8). Since $\Delta(\widehat\beta,\beta^*)>0$, $h\in\mathscr{C}(\xi,S)$ when $(|w_S|_\infty\lambda+z_0^*)/(\lambda-z_1^*)\le\xi$. For $j\notin S$, $h_j(\dot\psi(\beta+h)-\dot\psi(\beta))_j=\widehat\beta_j(z-\dot\psi(\beta^*)-g)_j\le|\widehat\beta_j|(w_j\lambda-g_j)\le0$. □

Proof of Theorem 4. Let $h=\widehat\beta-\beta^*$. Since $\psi(\beta)$ is a convex function,

$$t^{-1}\Delta(\beta^*+th,\beta^*)=\frac{\partial}{\partial t}\Big\{\psi(\beta^*+th)-t\big\langle h,\dot\psi(\beta^*)\big\rangle\Big\}$$

is an increasing function of t. For $0\le t\le1$ and in the event Ω_1, (8) implies

$$t^{-1}\Delta(\beta^*+th,\beta^*)\le\Delta(h+\beta^*,\beta^*)<(|w_S|_\infty\lambda+z_0^*)|h_S|_1.$$

By (9) and (14), $F(\xi,S;\phi_0,\phi_0)\le\Delta(\beta^*+th,\beta^*)e^{\phi_0(th)}/\{t|h_S|_1\phi_0(th)\}$ for $\phi_0(th)\le\eta^*$. Thus, for $\phi_0(th)\le\min\{\eta^*,\phi_0(h)\}$ and in the event Ω_1,

$$\phi_0(th)e^{-\phi_0(th)}\le\frac{\Delta(\beta^*+th,\beta^*)}{t|h_S|_1F(\xi,S;\phi_0,\phi_0)}<\frac{|w_S|_\infty\lambda+z_0^*}{F(\xi,S;\phi_0,\phi_0)}\le\eta e^{-\eta}.$$

If $\eta^*<\phi_0(h)$, the above inequality at $\phi_0(th)=\eta^*$ would give $\eta^*e^{-\eta^*}<\eta e^{-\eta}$, which contradicts to $\eta\le\eta^*\le1$. Thus, $\eta^*\ge\phi_0(h)$ and $\phi_0(th)e^{-\phi_0(th)}\le\eta e^{-\eta}$ for all $0\le t\le1$. This implies $\phi_0(h)\le\eta\le\eta^*$. Another application of (8) yields

$$\phi(h)\le\frac{\Delta(\beta^*+h,\beta^*)e^{\phi_0(h)}}{F(\xi,S;\phi_0,\phi)|h_S|_1}\le\frac{(|w_S|_\infty\lambda+z_0^*)e^\eta}{F(\xi,S;\phi_0,\phi)}.$$

We obtain (17) by applying (16) with $\phi=\phi_{1,S}$ to the right-hand side of (8). □

Proof of Lemma 8. (i) Since $\dot\psi(\beta)=\sum_{i=1}^n x^i\dot\psi_0(x^i\beta)/n$ by (23),

$$\begin{aligned}\mathrm{E}_\beta\exp\Big\{\frac{n}{\sigma^2}\langle b,z-\dot\psi(\beta)\rangle\Big\} &= \exp\Big[\sum_{i=1}^n\frac{\psi_0(x^i(\beta+b))-\psi_0(x^i\beta)-(x^ib)\dot\psi_0(x^i\beta)}{\sigma^2}\Big]\\ &= \exp\Big[\sum_{i=1}^n\int_0^1\frac{(x^ib)^2\ddot\psi_0(x^i(\beta+tb))}{\sigma^2}(1-t)dt\Big].\end{aligned}\tag{48}$$

This and (24) imply that for $M_1|Xb|_\infty\le\eta_0$,

$$\mathrm{E}_{\beta^*}\exp\Big\{\frac{n}{\sigma^2}\langle b,z-\dot\psi(\beta^*)\rangle\Big\}\le\exp\Big[\frac{ne^{\eta_0}\langle b,\Sigma^*b\rangle}{2\sigma^2}\Big].\tag{49}$$

Since $\max_{k=0,1}z_k^*/\lambda_k=\max_j t_j^{-1}|z_j-\dot\psi_j(\beta^*)|$ by (5),

$$\begin{aligned}\mathrm{P}_{\beta^*}\Big\{\max_{k=0,1}z_k^*/\lambda_k>1\Big\} &\le \sum_{j=1}^p\mathrm{P}_{\beta^*}\Big\{|z_j-\dot\psi_j(\beta^*)|>t_j\Big\}\\ &\le \sum_{j=1}^p\mathrm{E}_{\beta^*}\exp\Big\{\frac{n}{\sigma^2}b_j|z_j-\dot\psi_j(\beta^*)|-\frac{n}{\sigma^2}b_jt_j\Big\}\end{aligned}$$

with $b_j=e^{-\eta_0}t_j/\Sigma^*_{jj}$. Since $M_1\max_{ij}|x_{ij}|b_j\le\eta_0$, (49) gives

$$\mathrm{P}_{\beta^*}\Big\{\max_{k=0,1}z_k^*/\lambda_k>1\Big\}\le\sum_{j=1}^p 2\exp\Big(-\frac{ne^{-\eta_0}t_j^2}{2\sigma^2\Sigma^*_{jj}}\Big).$$

(ii) If (30) holds, we simply replace $\ddot\psi_0(x^i(\beta+tb))$ by c_0 in (48). The rest is simpler and omitted. □

Proof of Theorem 9. (i) Since $F^*(\xi,S;\phi)$ in (25) is a lower bound of $F(\xi,S;\phi_0,\phi)$ in (14), (32) follows from Theorem 4 with $\phi_0(b)=M_2|b|_2$. The probability statement follows from Lemma 8. (ii) Since (18) holds for the $\phi_0(b)$ in (26), we are allowed to use $F_*(\xi,S)=F_0(\xi,S;\phi_0)$ in Corollary 7. The condition $\eta^*=\infty$ is used since $\phi_0(b)$ does not control $M_1|Xb|_\infty$. (iii) We are also allowed to use $\phi_0(b)=M_3|b_S|_1$ in (28) due to $M_1|Xb|_\infty\le\phi_0(b)$. □

Proof of Theorem 14. Let $h=\widehat\beta-\beta^*$, $w_j=\widehat w_j$ and $S=\{j:|\widehat\beta_j|>\gamma_0\lambda\}\cup S_0$. For $j\notin S$, $w_j=\dot\rho_\lambda(|\widetilde\beta_j|)/\lambda\ge\{\dot\rho_\lambda(0+)-\kappa\gamma_0\lambda\}/\lambda=1-\kappa\gamma_0$, so that $z_1^*=|W_{S^c}^{-1}\{z-\dot\psi(\beta^*)\}_{S^c}|_\infty\le\lambda_0/(1-\kappa\gamma_0)=\lambda/A$. We also have $z_0^*\le|z-\dot\psi(\beta^*)|_\infty\le\lambda_0=(1-\kappa\gamma_0)\lambda/A$. Since $|\widehat w|_\infty\le1$, these bounds for z_0^* and z_1^* yield

$$\frac{|\widehat w_S|_\infty\lambda+z_0^*}{\lambda-z_1^*}\le\frac{\lambda+(1-\kappa\gamma_0)\lambda/A}{\lambda-\lambda/A}=\frac{A+1-\kappa\gamma_0}{A-1}\le\xi.$$

Thus, since $|g_j|\le\widehat w_j\lambda$ in (8), Lemma 1 provides

$$h\in\mathscr{C}(\xi,S),\ \Delta(\beta^*+h,\beta^*)\le|h_S|_2\big(|\widehat w_S|_2\lambda+|\{z-\dot\psi(\beta^*)\}_S|_2\big)$$

Since $|S\setminus S_0|\le|(\widetilde\beta-\beta^*)_{S_0^c}|_2^2/(\gamma_0^2\lambda^2)\le\ell^*$, we have by (36)

$$|w_S|_\infty\lambda+z_0^*\le\lambda+\lambda_0=\lambda_0\{1+A/(1-\kappa\gamma_0)\}\le F(\xi,S;\phi_0,\phi_0)\eta e^{-\eta}.$$

Thus, $\phi_0(h) \le \eta$ by (16), so that by (34) and (36) ,

$$e^{-\eta} F_* |h_S|_2 |h|_2 \le \Delta(\beta^* + h, \beta^*) \le |h_S|_2 \big(|\widehat{w}_S|_2 \lambda + |\{z - \dot\psi(\beta^*)\}_S|_2 \big).$$

Since $|h_S| = 0$ implies $h = 0$ for $h \in \mathscr{C}(\xi, S)$, we find

$$e^{-\eta} F_* |h|_2 \le |\widehat{w}_S|_2 \lambda + |\{z - \dot\psi(\beta^*)\}_S|_2. \tag{50}$$

Since $\widehat{w}_j \lambda = \dot\rho_\lambda(|\widetilde\beta_j|) \le \dot\rho_\lambda(|\beta^*_j|) + \kappa |\widetilde\beta_j - \beta^*_j|$, we have

$$|\widehat{w}_S|_2 \lambda \le |\dot\rho_\lambda(|\beta^*_{S_0}|)|_2 + \kappa |\widetilde\beta - \beta^*|_2.$$

Since $|z - \dot\psi(\beta^*)|_\infty \le \lambda_0 = (1 - \kappa\gamma_0)\lambda/A$ and $|\widetilde\beta_j - \beta^*_j| = |\widetilde\beta_j| \ge \gamma_0 \lambda$ for $j \in S \setminus S_0$,

$$\begin{aligned} |\{z - \dot\psi(\beta^*)\}_S|_2 &\le |\{z - \dot\psi(\beta^*)\}_{S_0}|_2 + |S \setminus S_0|^{1/2} (1 - \kappa\gamma_0)\lambda/A \\ &\le |\{z - \dot\psi(\beta^*)\}_{S_0}|_2 + |\widetilde\beta - \beta^*|_2 (1 - \kappa\gamma_0)/(\gamma_0 A). \end{aligned}$$

Inserting the above inequalities into (50), we find that

$$e^{-\eta} F_* |\widehat\beta - \beta^*|_2 \le |\dot\rho_\lambda(|\beta^*_{S_0}|)|_2 + |\{z - \dot\psi(\beta^*)\}_{S_0}|_2 + \Big(\kappa + \frac{1}{\gamma_0 A} - \frac{\kappa}{A} \Big) |\widetilde\beta - \beta^*|_2.$$

The probability statement follows directly from (35) with the union bound. □

Proof of Theorem 15. Let $R^{(\ell)}$ be as in Remark 16. For $|z - \dot\psi(\beta^*)|_\infty \le \lambda_0$, (16) of Theorem 4 gives $|\widehat\beta^{(0)} - \beta^*|_2 \le e^\eta (\lambda + \lambda_0)/F(\xi_0, S_0; \phi_0, |\cdot|_2) = R^{(0)}$. Under conditions (38) and (40), we have $R^{(\ell)} \le \gamma_0 \lambda \sqrt{\ell^*}$ for all $\ell \ge 0$. We prove (39) by induction. We have already proved (39) for $\ell = 0$. For $\ell \ge 1$, we let $\widetilde\beta = \widehat\beta^{(\ell-1)}$ and apply Theorem 14: $|\widehat\beta^{(\ell)} - \beta^*|_2 \le (1 - r_0) R^{(\infty)} + r_0 R^{(\ell-1)} = R^{(\ell)}$. The probability statement follows directly from (35) with the union bound. □

Proof of Theorem 18. Let $\widetilde z = z - \dot\psi(\beta^*)$ and λ be fixed. Consider

$$\widehat\beta(\lambda, t) = \arg\min_\beta \Big\{ \psi(\beta) - \langle \beta, \dot\psi(\beta^*) + t\widetilde z \rangle + t\lambda \sum_{j=1}^p \widehat w_j |\beta_j| : \beta_{S^c} = 0 \Big\} \tag{51}$$

as an artificial path for $0 \le t \le 1$. For each t, the KKT conditions for $\widehat\beta(\lambda, t)$ are

$$g_S(\lambda, t) = t\lambda \widehat W_S u_S(\lambda, t),\ u_j(\lambda, t) \begin{cases} = \operatorname{sgn}(\widehat\beta_j(\lambda, t)) & \forall \widehat\beta_j(\lambda, t) \ne 0 \\ \in [-1, 1], & \forall j \in S, \end{cases}$$

where $g(\lambda, t) = -\dot\psi(\widehat\beta(\lambda, t)) + \dot\psi(\beta^*) + t\widetilde z$. Since (51) is constrained to $\beta_{S^c} = 0$ and both the error $\widetilde z$ and the penalty level λ are scaled with t, Theorem 4 with $\xi = 0$ yields

$$\phi_0\big(\widehat\beta(\lambda, t) - \beta^*\big) \le \eta_t \to 0 \ \text{ with } \ \eta_t e^{-\eta_t} = t\eta e^{-\eta},\ \forall\, 0 < t \le 1. \tag{52}$$

Let $S_t = \{ j : \widehat\beta_j(\lambda, t) \ne 0 \}$. Applying the differentiation operator $D = (\partial/\partial t)$ to the KKT conditions, we find that almost everywhere in t,

$$(Dg)_{S_t}(\lambda, t) = \widetilde z_{S_t} - \ddot\psi_{S_t}(\widehat\beta(\lambda, t)) \{ (D\widehat\beta)(\lambda, t) \}_{S_t} = \lambda \widehat W_{S_t} u_{S_t}(\lambda, t).$$

It follows that

$$(D\widehat{\beta})_{S_t}(\lambda,t) = \{\ddot{\psi}_{S_t}(\widehat{\beta}(\lambda,t))\}^{-1}\{\widetilde{z}_{S_t} - \lambda\widehat{W}_{S_t}u_{S_t}(\lambda,t)\} \tag{53}$$

and with an application of the chain rule,

$$(Dg)_{S^c}(\lambda,t) = \widetilde{z}_{S^c} - \ddot{\psi}_{S^c,S_t}(\widehat{\beta}(\lambda,t))\{\ddot{\psi}_{S_t}(\widehat{\beta}(\lambda,t))\}^{-1}\{\widetilde{z}_{S_t} - \lambda\widehat{W}_{S_t}u_{S_t}(\lambda,t)\}.$$

Since $g(\lambda,t)$ is almost differentiabe and $\widehat{\beta}(\lambda,0+) = \beta^*$, we have $g(\lambda,0+) = 0$ and

$$g_{S^c}(\lambda,1-) = \int_0^1 \Big[\widetilde{z}_{S^c} - \ddot{\psi}_{S^c,S_t}(\widehat{\beta}(\lambda,t))\{\ddot{\psi}_{S_t}(\widehat{\beta}(\lambda,t))\}^{-1}\{\widetilde{z}_{S_t} - \lambda\widehat{W}_{S_t}u_{S_t}(\lambda,t)\}\Big]dt.$$

Thus, (52), (41), and (42) imply

$$|\widehat{W}_{S^c}^{-1}g_{S^c}(\lambda,1-)|_\infty \le |\widehat{W}_{S^c}^{-1}\widetilde{z}_{S^c}|_\infty + \kappa_1|\widetilde{z}_S|_\infty + \kappa_0\lambda|u_{S_t}(\lambda,t)|_\infty,$$

which is smaller than λ in the event in (43). Thus, since $\ddot{\psi}_S(\widehat{\beta}(\lambda,1-))$ is of full rank, $\widehat{\beta}(\lambda,1-)$ is the unique solution of the KKT conditions (3) for $\widehat{\beta}$. This completes the proof of part (i).

For part (ii), we observe that (44) implies $S = \{j : \beta_j^* \neq 0\}$. Since $\widehat{\beta}(\lambda,0+) = \beta^*$, there exists $t_1 > 0$ such that $u_S(\lambda,t) = \mathrm{sgn}(\beta_S^*)$ for all $0 < t < t_1$. By (52), $\widehat{\beta}(\lambda,t) \in \mathscr{B}_0$ for $0 < t < t_1$. It follows from (53) and (44) that

$$|(D\widehat{\beta})_S(\lambda,t)|_\infty \le \big\|\{\ddot{\psi}_{S_t}(\widehat{\beta}(\lambda,t))\}^{-1}\big\|_\infty|\widetilde{z}_S - \lambda\widehat{W}_S\,\mathrm{sgn}(\beta_S^*)|_\infty < \min_{j\in S}|\beta_j^*| - \varepsilon_1$$

for $0 < t < t_1$ and some $\varepsilon_1 > 0$. Since $\widehat{\beta}(\lambda,0+) = \beta^*$, this implies $|\widehat{\beta}_S(\lambda,t) - \beta_S^*|_\infty < \min_{j\in S}|\beta_j^*| - \varepsilon_1$ for all $0 < t < t_1 \wedge 1$. It follows that $\mathrm{sgn}(\widehat{\beta}(\lambda,t)) = \mathrm{sgn}(\beta^*)$ for $0 < t \le 1$ by the continuity of $\widehat{\beta}(\lambda,t)$ in t, that is, $t_1 = 1$. Consequently, conditions (41), and (42) are only needed for the smaller class $\mathscr{B}_0$ in the proof of part (i). This gives $\widehat{\beta}(\lambda,1) = \widehat{\beta}$ and completes the proof of part (ii).

Finally, in part (iii), $F_0(\xi,S;\phi_0,\phi_0)$ is simply replaced by its lower bounds with the respective ϕ_0. □

Proof of Theorem 19. Suppose the event Ω_1 in (15) happens, so that $\widehat{w}_j \ge w_j$ for $j \notin S$ and the conclusion of Theorem 4 hold. Let $h = \widehat{\beta} - \beta^*$ and $\widehat{\Sigma} = \int_0^1 \ddot{\psi}(\beta^* + xh)dx$. It follows from (1) that $\widehat{\Sigma}h = \dot{\psi}(\beta^* + h) - \dot{\psi}(\beta^*) = \dot{\ell}(\widehat{\beta}) - \dot{\ell}(\beta^*)$. By the KKT conditions (3),

$$|(\widehat{\Sigma}h)_j| = |(\dot{\ell}(\widehat{\beta}) - \dot{\ell}(\beta^*))_j| \ge \widehat{w}_j\lambda - z_j \ge w_j(\lambda - z_1^*) > 0, \quad j \notin S.$$

Let $B \subseteq \{j \notin S : \widehat{\beta}_j \neq 0\}$ with $|B| \le d_1$. It follows from Theorem 4 that $\phi_0(h) \le \eta \le \eta^*$, so that (45) implies $\max_{|u|_2=1}|(W^{-1}\widehat{\Sigma}^{1/2}u)_B|_2^2 = \lambda_{\max}(W_B^{-2}\widehat{\Sigma}_B) \le \kappa_+(d_1)$. Thus, by the definition of $\Delta(\beta,\beta^*)$ in (4),

$$(\lambda - z_1^*)^2|B| \le |(W^{-1}\widehat{\Sigma}h)_B|_2^2 \le \kappa_+(d_1)\langle h,\widehat{\Sigma}h\rangle = \kappa_+(d_1)\Delta(\beta^* + h,\beta^*).$$

This and the prediction bound in Theorem 4 yield

$$|B| \le \frac{\kappa_+(d_1)\Delta(\beta^* + h,\beta^*)}{(\lambda - z_1^*)^2} \le \frac{\kappa_+(d_1)e^\eta(|w_S|_\infty\lambda + z_0^*)^2|S|}{(\lambda - z_1^*)^2F(\xi,S;\phi_0,\phi_{1,S})} \le \frac{\kappa_+(d_1)e^\eta\xi^2|S|}{F(\xi,S;\phi_0,\phi_{1,S})} < d_1.$$

Since all subsets $B \subseteq \{j \notin S : \widehat{\beta}_j \neq 0\}$ with $|B| \leq d_1$ satisfies $|B| < d_1$, it must hold that $\#\{j \notin S : \widehat{\beta}_j \neq 0\} < d_1$. □

Proof of Theorem 20. Let $\widetilde{z} = z - \dot{\psi}(\beta^*) = -\dot{\ell}(\beta^*)$ and $\widehat{\beta}(\lambda,t)$ be the artificial estimator in (51) with $\widehat{w}_j = 1$, and $h(\lambda,t) = \widehat{\beta}(\lambda,t) - \beta^*$. Let $\lambda_* \leq \lambda^*$ be penalty levels satisfying

$$[\lambda_*,\lambda^*] \subseteq \cap_{0<t\leq 1}\Big\{\lambda : \phi_0\big(h(\lambda,t)\big) \leq \eta, h(\lambda,t) \in \mathscr{C}(\xi,S), \big|(\Sigma^*)^{-1/2}\widetilde{z}\big|_2 \leq \frac{\alpha\lambda\sqrt{d_1}}{e^\eta\sqrt{c^*}}\Big\}. \tag{54}$$

We pick such an interval $[\lambda_*,\lambda^*]$ containing the penalty level λ of concern in the theorem. This is allowed by Lemma 1 and Theorem 4. We first prove the stronger conclusion

$$\max_{\lambda_*\leq\lambda\leq\lambda^*}\max_{0<t\leq 1}\#\{j : \widehat{\beta}_j(\lambda,t) \neq 0, j \notin S\} < d_1 \tag{55}$$

under the additional assumption

$$\min_{\lambda_*\leq\lambda\leq\lambda^*}\min_{0<t\leq 1}\#\{j : \widehat{\beta}_j(\lambda,t) \neq 0, j \notin S\} \leq d_1. \tag{56}$$

Let $g(\lambda,t) = t\widetilde{z} + \dot{\psi}(\beta^*) - \dot{\psi}(\widehat{\beta}(\lambda,t))$ be the negative gradient at $\widehat{\beta}(\lambda,t)$ in (51). By the KKT conditions for (51), $\mathrm{sgn}(\widehat{\beta}_j(\lambda,t\pm)) \neq 0$ implies $|g(\lambda,t)| = t\lambda$. Thus, (56) implies the existence of $\lambda \in [\lambda_*,\lambda^*]$, $t_1 \in (0,1]$, and $A_1 \subset \{1,\ldots,p\}$ satisfying

$$\big\{j : \mathrm{sgn}(\widehat{\beta}_j(\lambda,t_1)) \neq 0\big\} \cup S \subseteq A_1 \subseteq \big\{j : |g(\lambda,t_1)| = t_1\lambda\big\} \cup S, \ \ |A_1| \leq d_1 + |S|. \tag{57}$$

Moreover, if $\max_{\lambda_*\leq\lambda\leq\lambda^*}\max_{0<t\leq 1}\#\{j : \widehat{\beta}_j(\lambda,t) \neq 0, j \notin S\} \geq d_1$, then by the continuity of $\widehat{\beta}(\lambda,t)$, it would be possible to restrict (57) to $|A_1| = d_1 + |S|$ with some different $\lambda \in [\lambda_*,\lambda^*]$ and $t_1 \in (0,1]$. Therefore, it suffices to deny this possibility by proving $|A_1| < d_1 + |S|$ based on (57) and (54). Let $A_0 = A_1 \setminus S$. We prove $|A_0| < d_1$, which is equivalent to $|A_1| < d_1 + |S|$.

Let $v_{(A)} = (v_j I\{j \in A\}, j \in A_1)' \in \mathbb{R}^{A_1}$ and $v_A = (v_j, j \in A)' \in \mathbb{R}^A$ for all vectors $v = (v_1,\ldots,v_p)'$. Let $h = h(\lambda,t_1)$, $\widehat{\Sigma} = \int_0^1 \ddot{\psi}(\beta^* + xh)dx$, and $g = g(\lambda,t_1) = t_1\widetilde{z} + \dot{\psi}(\beta^*) - \dot{\psi}(\beta^* + h) = t_1\widetilde{z} - \widehat{\Sigma}h$. Since $h_{A_1^c} = 0$, $\widehat{\Sigma}_{A_1}^{-1}g_{(A_1)} = t_1\widehat{\Sigma}_{A_1}^{-1}\widetilde{z}_{A_1} - \widehat{\Sigma}_{A_1}^{-1}(\widehat{\Sigma}h)_{A_1} = t_1\widehat{\Sigma}_{A_1}^{-1}\widetilde{z}_{A_1} - h_{A_1}$. Thus, since $g_j = t_1\lambda\,\mathrm{sgn}(h_j)$ for $j \in A_0$ by the KKT conditions,

$$\big\langle g_{(A_0)}, \widehat{\Sigma}_{A_1}^{-1}g_{(A_1)}\big\rangle = t_1\big\langle g_{(A_0)}, \widehat{\Sigma}_{A_1}^{-1}\widetilde{z}_{A_1}\big\rangle - \big\langle g_{(A_0)}, h_{A_1}\big\rangle \leq t_1\big\langle g_{(A_0)}, \widehat{\Sigma}_{A_1}^{-1}\widetilde{z}_{A_1}\big\rangle.$$

Since $|\widehat{\Sigma}_{A_1}^{-1/2}g_{(A_0)}|_2^2 + |\widehat{\Sigma}_{A_1}^{-1/2}g_{(A_1)}|_2^2 = |\widehat{\Sigma}_{A_1}^{-1/2}g_{(S)}|_2^2 + 2\langle g_{(A_0)}, \widehat{\Sigma}_{A_1}^{-1}g_{(A_1)}\rangle$, we have

$$|\widehat{\Sigma}_{A_1}^{-1/2}g_{(A_0)}|_2^2 + |\widehat{\Sigma}_{A_1}^{-1/2}g_{(A_1)}|_2^2 \leq |\widehat{\Sigma}_{A_1}^{-1/2}g_{(S)}|_2^2 + 2t_1|\widehat{\Sigma}_{A_1}^{-1/2}g_{(A_0)}|_2|\widehat{\Sigma}_{A_1}^{-1/2}\widetilde{z}_{A_1}|_2.$$

By (54) and (46), $|\widehat{\Sigma}_{A_1}^{-1/2}\widetilde{z}_{A_1}|_2 \leq e^{\eta/2}|(\Sigma^*_{A_1})^{-1/2}\widetilde{z}_{A_1}|_2 \leq \alpha\lambda\sqrt{|A_0|/(c^*e^\eta)}$, so that

$$(1-\alpha)|\widehat{\Sigma}_{A_1}^{-1/2}g_{(A_0)}|_2^2 + |\widehat{\Sigma}_{A_1}^{-1/2}g_{(A_1)}|_2^2 \leq |\widehat{\Sigma}_{A_1}^{-1/2}g_{(S)}|_2^2 + \alpha t_1^2\lambda^2|A_0|/(c^*e^\eta).$$

Moreover, since $|A_1| \leq d_1 + |S| \leq d^*$, it follows from (54), (46), and (47) that the eigenvalues of $\widehat{\Sigma}_{A_1}$ all lie in the interval $c_*e^{-\eta}$ and c^*e^η. Thus, since $g_{A_0} = t_1\lambda\,\mathrm{sgn}(\widehat{\beta}_{A_0})$,

$$\frac{(1-\alpha)t_1^2\lambda^2|A_0|}{c^*e^\eta} + \frac{t_1^2\lambda^2|A_0| + |g_S|_2^2}{c^*e^\eta} \leq \frac{|g_S|_2^2}{c_*e^{-\eta}} + \frac{\alpha t_1^2\lambda^2|A_0|}{c^*e^\eta}.$$

Since $|g|_\infty \le t_1\lambda$, the above inequality gives by algebra the dimension bound

$$|A_0| \le \Big(\frac{e^{2\eta}c^*/c_* - 1}{2-2\alpha}\Big)\frac{|g_S|_2^2}{t_1^2\lambda^2} \le \Big(\frac{e^{2\eta}c^*/c_* - 1}{2-2\alpha}\Big)|S| < d_1.$$

This proves (55) under the additional assumption (56).

Now we prove (56). In the special case of $\phi_0(b) = 0$, the condition on λ in (54) is monotone so that we are allowed to pick $\lambda^* = \infty$. Since $\widehat{\beta}(\lambda, 1) = 0$ for very large λ, (56) holds automatically for $\phi_0(b) = 0$. By (46), this special case is equivalent to linear regression since the Hessian does not depend on β. The difference of the general model (1) from linear regression is that the condition $\lambda + z_0^* \le \eta e^{-\eta} F(\xi, S; \phi_0, \phi_0)$, which excludes large λ, is needed to prove $\phi_0(h(\lambda,t)) \le \eta$ by Theorem 4. To overcome this difficulty, we consider very small $t > 0$. Let $b = (\beta - \beta^*)/t$. By (51),

$$\begin{aligned} t^{-1}\big\{\widehat{\beta}(\lambda,t) - \beta^*\big\} &= \arg\min_b \Big\{\psi(\beta^* + tb) - \langle tb, \dot\psi(\beta^*) + t\widetilde{z}\rangle + t\lambda|\beta^* + tb|_1\Big\} \\ &= \arg\min_b \Big\{\int_0^1 (1-x)\big\langle tb, \ddot\psi(\beta^* + xtb)tb\big\rangle dx - t^2\langle b, \widetilde{z}\rangle + t\lambda|\beta^* + tb|_1\Big\} \\ &= \arg\min_b \Big\{\int_0^1 (1-x)\big\langle b, \ddot\psi(\beta^* + xtb)b\big\rangle dx - \langle b, \widetilde{z}\rangle + \lambda|\beta^*/t + b|_1\Big\}. \end{aligned}$$

Let $S_0 = \{j : \beta_j^* \ne 0\}$. Since $\lambda|\beta^*/t + b|_1 - \lambda|\beta^*|_1/t \to \lambda\langle \mathrm{sgn}(\beta^*), b\rangle + \lambda|b_{S_0^c}|_1$ as $t \to 0+$, $t^{-1}\big\{\widehat{\beta}(\lambda,t) - \beta^*\big\}$ converges (along a subsequence if necessary) to

$$\widehat{b}(\lambda) = \arg\min_b \Big\{2^{-1}\big\langle b, \ddot\psi(\beta^*)b\big\rangle - \langle b, \widetilde{z}\rangle + \lambda\langle \mathrm{sgn}(\beta^*), b\rangle + \lambda|b_{S_0^c}|_1\Big\}.$$

Moreover, since $\widetilde{z} - \ddot\psi(\beta^*)\widehat{b}(\lambda)$ is the negative gradient at $\widehat{b}(\lambda)$, we have

$$\{j : |g_j(\lambda,t)| = t\lambda, j \notin S\} \to \{j \notin S : \big(\widetilde{z} - \ddot\psi(\beta^*)\widehat{b}(\lambda)\big)_j = \lambda\,\mathrm{sgn}(\widehat{b}_j(\lambda))\}. \tag{58}$$

Since this limit does not depend on $\phi_0(\cdot)$, the dimension bound (55) in the special case of linear regression implies that the right-hand side of (58) contains a smaller number of elements than d_1. This gives (56) in the general case by (58) and completes the proof. □

References

P. J. Bickel, Y. Ritov, and A. Tsybakov. Simultaneous analysis of Lasso and Dantzig selector. *Annals of Statistics*, 37(4):1705–1732, 2009.

L. M. Bregman. The relaxation method of finding the common point of convex sets and its application to the solution of problems in convex programming. *USSR Computational Mathematics and Mathematical Physics*, 7:200–217, 1967.

F. Bunea, A. Tsybakov, and M. H. Wegkamp. Sparsity oracle inequalities for the Lasso. *Electronic Journal of Statistics*, 1:169–194, 2007.

E. J. Candes and T. Tao. Decoding by linear programming. *IEEE Trans. on Information Theory*, 51:4203–4215, 2005.

E. J. Candes and T. Tao. The dantzig selector: statistical estimation when p is much larger than n (with discussion). *Annals of Statistics*, 35:2313–2404, 2007.

S. Chen, D. L. Donoho, and M. A. Saunders. Atomic decomposition by basis pursuit. *SIAM J. Sci. Comput.*, 20:33–61, 1998.

J. Fan and R. Li. Variable selection via nonconcave penalized likelihood and its oracle properties. *Journal of the American Statistical Association*, 96:1348–1360, 2001.

E. Greenshtein and Y. Ritov. Persistence in high–dimensional linear predictor selection and the virtue of overparametrization. *Bernoulli*, 10:971–988, 2004.

J. Huang, S. Ma, and C.-H. Zhang. Adaptive lasso for sparse high-dimensional regression models. *Statistica Sinica*, 18:1603–1618, 2008.

D. R. Hunter and R. Li. Variable selection using mm algorithms. *Annals of Statistics*, 33:1617–1642, 2005.

V. Koltchinskii. The dantzig selector and sparsity oracle inequalities. *Bernoulli*, 15:799–828, 2009.

P. McCullagh and J. A. Nelder. *Generalized Linear Models*. Chapmann & Hall, 1989.

L. Meier and P. Bühlmann. Smoothing ℓ_1-penalized estimators for high-dimensional time-course data. *Electronic Journal of Statistics*, 1:597–615, 2007.

N. Meinshausen and P. Bühlmann. High-dimensional graphs and variable selection with the lasso. *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*, 37:246–270, 2009.

S. Negahban, P. Ravikumar, M. J. Wainwright, and B. Yu. A unified framework for high-dimensional analysis of m-estimators with decomposable regularizer. Technical Report arXiv:1010.2731, arXiv, 2010.

F. Nielsen and R. Nock. On the centroids of symmetrized bregman divergences. *CoRR*, abs/0711.3242, 2007.

P. Ravikumar, M. J. Wainwright, G. Raskutti, and B. Yu. Model selection in gaussian graphical models: High-dimensional consistency of ℓ_1-regularized mle. In *Advances in Neural Information Processing Systems (NIPS)*, volume 21, 2008.

A. J. Rothman, P. J. Bickel, E. Levina, and J. Zhu. Sparse permutation invariant covariance estimation. *Electronic Journal of Statistics*, 2:494–515, 2008.

N. Städler, P. Bühlmann, and S. van de Geer. ℓ_1-penalization for mixture regression models (with discussion). *Test*, 19(2):209–285, 2010.

T. Sun and C.-H. Zhang. Scaled sparse linear regression. Technical Report arXiv:1104.4595, arXiv, 2011.

R. Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 58:267–288, 1996.

R. Tibshirani and J. Taylor. The solution path of the generalized lasso. *The Annals of Statistics*, 39: 1335–1371, 2011.

J. A. Tropp. Just relax: convex programming methods for identifying sparse signals in noise. *IEEE Transactions on Information Theory*, 52:1030–1051, 2006.

S. van de Geer. The deterministic lasso. Technical Report 140, ETH Zurich, Switzerland, 2007.

S. van de Geer. High–dimensional generalized linear models and the lasso. *Annals of Statistics*, 36: 614–645, 2008.

S. van de Geer and P. Bühlmann. On the conditions used to prove oracle results for the lasso. *Electronic Journal of Statistics*, 3:1360–1392, 2009.

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: 2183–2202, 2009.

F. Ye and C.-H. Zhang. Rate minimaxity of the lasso and dantzig selector for the ℓ_q loss in ℓ_r balls. *Journal of Machine Learning Research*, 11:3481–3502, 2010.

C.-H. Zhang. Least squares estimation and variable selection under minimax concave penalty. In *Mathematisches Forschungsintitut Oberwolfach: Sparse Recovery Problems in High Dimensions*, 3 2009.

C.-H. Zhang. Nearly unbiased variable selection under minimax concave penalty. *The Annals of Statistics*, 38:894–942, 2010a.

C.-H. Zhang and J. Huang. The sparsity and bias of the Lasso selection in high-dimensional linear regression. *Annals of Statistics*, 36(4):1567–1594, 2008.

T. Zhang. Analysis of multi-stage convex relaxation for sparse regularization. *Journal of Machine Learning Research*, 11:1087–1107, 2010b.

T. Zhang. Adaptive forward-backward greedy algorithm for learning sparse representations. *IEEE Transactions on Information Theory*, 57:4689–4708, 2011a.

T. Zhang. Multi-stage convex relaxation for feature selection. Technical Report arXiv:1106.0565, arXiv, 2011b.

P. Zhao and B. Yu. On model selection consistency of Lasso. *Journal of Machine Learning Research*, 7:2541–2567, 2006.

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. *Annals of Statistics*, 36(4):1509–1533, 2008.

Journal of Machine Learning Research 13 (2012) 1865-1890 Submitted 12/10; Revised 10/11; Published 6/12

Regularization Techniques for Learning with Matrices

Sham M. Kakade SKAKADE@MICROSOFT.COM
Microsoft Research New England
One Memorial Drive
Cambridge, MA 02142, USA

Shai Shalev-Shwartz SHAIS@CS.HUJI.AC.IL
School of Computer Science and Engineering
The Hebrew University of Jerusalem
Givat Ram, Jerusalem 91904, Israel

Ambuj Tewari AMBUJ@CS.UTEXAS.EDU
Department of Computer Science
The University of Texas at Austin
Austin, TX 78701, USA

Editor: Manfred K. Warmuth

Abstract

There is growing body of learning problems for which it is natural to organize the parameters into a matrix. As a result, it becomes easy to impose sophisticated prior knowledge by appropriately regularizing the parameters under some matrix norm. This work describes and analyzes a systematic method for constructing such matrix-based regularization techniques. In particular, we focus on how the underlying statistical properties of a given problem can help us decide which regularization function is appropriate.

Our methodology is based on a known duality phenomenon: a function is strongly convex with respect to some norm if and only if its conjugate function is strongly smooth with respect to the dual norm. This result has already been found to be a key component in deriving and analyzing several learning algorithms. We demonstrate the potential of this framework by deriving novel generalization and regret bounds for multi-task learning, multi-class learning, and multiple kernel learning.

Keywords: regularization, strong convexity, regret bounds, generalization bounds, multi-task learning, multi-class learning, multiple kernel learning

1. Introduction

As we tackle more challenging learning problems, there is an increasing need for algorithms that efficiently impose more sophisticated forms of prior knowledge. Examples include: the group Lasso problem (for "shared" feature selection across problems), multiple kernel learning, multi-class prediction, and multi-task learning. A central question here is to understand the performance of such algorithms in terms of the attendant complexity restrictions imposed by the algorithm. Such analyses often illuminate the nature in which our prior knowledge is being imposed.

The predominant modern machine learning method for imposing complexity restrictions is through *regularizing* a vector of parameters. Much work has gone into understanding the relationship between the nature of the regularization and the implicit prior knowledge imposed, particular

for the case of regularization with ℓ_2 and ℓ_1 norms (where the former is more tailored to rotational invariance and margins, while the latter is more suited to sparsity—see Ng, 2004). When dealing with more complex problems, we need systematic tools for designing more complicated regularization schemes. This work examines regularization based on group norms and spectral norms of *matrices*. We analyze the performance of such regularization methods and provide general hints for choosing a regularization function based on the underlying statistical properties of a given problem.

In particular, we use a recently developed methodology, based on the notion of *strong convexity*, for designing and analyzing the regret or generalization ability of a wide range of learning algorithms (see, e.g., Shalev-Shwartz, 2007 and Kakade et al., 2008). In fact, most of our efficient algorithms (both in the batch and online settings) impose some complexity control via the use of some *strictly convex* penalty function either explicitly via a regularizer or implicitly in the design of an online update rule. Central to understanding these algorithms is the manner in which these penalty functions are strictly convex, that is, the behavior of the "gap" by which these convex functions lie above their tangent planes, which is strictly positive for strictly convex functions. Here, the notion of strong convexity provides one means to characterize this gap in terms of some general norm rather than just the standard Euclidean norm.

The importance of strong convexity can be understood using the duality between strong convexity and strong smoothness. Strong smoothness measures how well a function is approximated at some point by its linearization obtained from a first-order Taylor expansion. Linear functions are easy to manipulate (e.g., because of the linearity of expectation). Hence, if a function is sufficiently smooth we can more easily control its behavior. We further distill the analysis given by Shalev-Shwartz (2007) and Kakade et al. (2008): based on the strong-convexity/smoothness duality, we derive a key inequality which seamlessly enables us to design and analyze a family of learning algorithms.

Our focus in this work is on learning with matrices. We characterize a number of matrix based regularization functions, of recent interest, as being strongly convex functions. This immediately allows us to derive learning algorithms by relying on the family of learning algorithms mentioned previously. Specifying the general performance bounds for the specific matrix based regularization method, we are able to systematically decide which regularization function is more appropriate based on underlying statistical properties of a given problem.

1.1 Our Contributions

This paper has a two-fold aim. First, we want to provide a unified framework for obtaining regret and generalization error bounds for algorithms based on strongly convex regularizers. We hope that it lets us view a large amount of previous work in this area from a unified perspective. Whether we have succeeded in our attempt or not is for the reader to judge. Second, we want to demonstrate that the proposed framework is also capable to generating some novel results.

Contributions towards the first aim include the following.

- We show how the framework based on strong convexity/strong smoothness duality (see Shalev-Shwartz, 2007; Kakade et al., 2008) provides a methodology for analyzing matrix based learning methods, which are of much recent interest. These results reinforce the usefulness of this framework in providing both learning algorithms, and their associated complexity analysis. For this reason, we further distill the analysis given by Shalev-Shwartz (2007) and Kakade

et al. (2008). We emphasize a key inequality which immediately enables us to design and analyze a family of learning algorithms.

- We provide template algorithms (both in the online and batch settings) for a number of machine learning problems of recent interest, which use matrix parameters. In particular, we provide a simple derivation of generalization/mistake bounds for: (i) online and batch multi-task learning using group or spectral norms, (ii) online multi-class categorization using group or spectral norms, and (iii) multiple kernel learning.

For the expert reader, who may want to jump directly to novel results, we summarize the new results below.

- We generalize recent results of Juditsky and Nemirovski (2008) to obtain a general result on the strong convexity/strong smoothness of certain group norms (Theorem 13).

- For the case of multi-class learning, we describe and analyze a new "group perceptron" algorithm (Algorithm 3) and show that with a shared structure between classes, this algorithm can significantly outperform previously proposed multi-class perceptron algorithms (Corollary 24).

- Our unified analysis simplifies previous analyses of recently proposed algorithms. For example, the generality of this framework allows us to simplify the proofs of previously proposed regret bounds for online multi-task learning (e.g., Cavallanti et al., 2008; Agarwal et al., 2008). Moreover, our multi-task guarantees (Corollaries 20 and 22) are valid for "heterogenous" tasks also: one task might be a regression task using squared loss, another might be a classification task using logistic loss and yet another task might be using hinge loss for classification.

- Bounds that follow immediately from our analysis are sometimes much sharper than previous results. For instance, Theorem 25 improves upon the bounds for multiple kernel learning given by Lanckriet et al. (2004) and Srebro and Ben-David (2006).

1.2 Related Work

We first discuss related work on learning with matrix parameters and then discuss the use of strong convexity in learning.

1.2.1 MATRIX LEARNING

This is growing body of work studying learning problems in which the parameters can be organized as matrices. Several examples are multi-class categorization (Crammer and Singer, 2000), multi-task and multi-view learning (Cavallanti et al., 2008; Agarwal et al., 2008), and online PCA (Warmuth and Kuzmin, 2006). It was also studied under the framework of group Lasso (Yuan and Lin, 2006; Obozinski et al., 2010; Bach, 2008).

In the context of learning vectors (rather than matrices), the study of the relative performance of different regularization techniques based on properties of a given task dates back to Littlestone (1988) and Kivinen and Warmuth (1997). In the context of batch learning, it was studied by several authors (e.g., see Ng, 2004).

We also note that much of the work on multi-task learning for regression is on union support recovery—a setting where the generative model specifies a certain set of relevant features (over all the tasks), and the analysis here focuses on the conditions and sample sizes under which the union of the relevant features can be correctly identified (e.g., Obozinski et al., 2010; Lounici et al., 2009). Essentially, this is a generalization of the issue of identifying the relevant feature set in the standard single task regression setting, under ℓ_1-regularized regression. In contrast, our work focuses on the agnostic setting of just understanding the sample size needed to obtain a given error rate (rather than identifying the relevant features themselves). The use of interesting matrix norms like group norms and Schatten norms in the context of multi-task learning appears in the work of Argyriou et al. (2008).

We also discuss related work on kernel learning in Section 6. Our analysis here uses the equivalence between kernel learning and group Lasso (as noted by Bach, 2008).

1.2.2 STRONG CONVEXITY/STRONG SMOOTHNESS

The notion of *strong convexity* takes its roots in optimization. Zalinescu (2002) attributes it to a paper of Polyak in the 1960s. Relatively recently, its use in machine learning has been two fold: in deriving regret bounds for online algorithms and generalization bounds in batch settings.

The duality of strong convexity and strong smoothness was first used by Shalev-Shwartz and Singer (2006) and Shalev-Shwartz (2007) in the context of deriving low regret online algorithms. Here, once we choose a particular strongly convex penalty function, we immediately have a family of algorithms along with a regret bound for these algorithms that is in terms of a certain strong convexity parameter. A variety of algorithms (and regret bounds) can be seen as special cases.

A similar technique, in which the Hessian is directly bounded, is described by Grove et al. (2001) and Shalev-Shwartz and Singer (2007). Another related approach involved bounding a Bregman divergence (Kivinen and Warmuth, 1997, 2001; Gentile, 2003) (see Cesa-Bianchi and Lugosi, 2006 for a detailed survey). The Bregman approach also implicitly relies on exploiting the strong convexity of the underlying Bregman function. Two canonical functions here are: $\frac{1}{2}\|\cdot\|_2^2$, which is strongly convex w.r.t. ℓ_2 norm, and negative entropy, which is strongly convex w.r.t. ℓ_1 norm. These lead to additive and multiplicative updates respectively. Another interesting application of the very same duality is for deriving and analyzing boosting algorithms (Shalev-Shwartz and Singer, 2008).

More recently, Kakade et al. (2008) showed how to use the very same duality for bounding the Rademacher complexity of classes of linear predictors. That the Rademacher complexity is closely related to Fenchel duality was shown in Meir and Zhang (2003), and the work by Kakade et al. (2008) made the further connection to strong convexity. Again, under this characterization, a number of generalization and margin bounds (for methods which use linear prediction) are immediate corollaries, as one only needs to specify the strong convexity parameter from which these bounds easily follow (see Kakade et al., 2008 for details).

The concept of strong smoothness (essentially a second order upper bound on a function) has also been in play in a different literature: the analysis of the concentration of martingales in *smooth* Banach spaces (Pinelis, 1994; Pisier, 1975). This body of work seeks to understand the concentration properties of a random variable $\|X_t\|$, where X_t is a (vector valued) martingale and $\|\cdot\|$ is a smooth norm, say an L_p-norm.

Recently, Juditsky and Nemirovski (2008) used the fact that a *norm* is strongly convex if and only if its conjugate is strongly smooth. This duality was useful in deriving concentration properties

of a random variable $\|\mathbf{M}\|$, where now $\mathbf{M}$ is a random matrix. The norms considered here were the (Schatten) L_p-matrix norms and certain "block" composed norms (such as the $\|\cdot\|_{2,q}$ norm).

1.3 Organization

The rest of the paper is organized as follows. In Section 2, we describe a general family of learning algorithms. In particular, after presenting the duality of strong-convexity/strong-smoothness, we isolate an important inequality (Corollary 4) and show that this inequality alone seamlessly yields regret bounds in the online learning model and Rademacher bounds (that leads to generalization bounds in the batch learning model). We further highlight the importance of strong convexity to matrix learning applications by drawing attention to families of strongly convex functions over matrices. Next, in Section 3 we show how the obtained bounds can be used for systematically choosing an adequate prior knowledge (i.e., regularization) based on properties of the given task. We then turn to describe the applicability of our approach to more complex prediction problems. In particular, we study multi-task learning (Section 4), multi-class categorization (Section 5), and kernel learning (Section 6). Naturally, many of the algorithms we derive have been proposed before. Nevertheless, our unified analysis enables us to simplify previous analyzes, understand the merits and pitfalls of different schemes, and even derive new algorithms/analyses.

2. Preliminaries and Techniques

In this section we describe the necessary background. Most of the results below are not new and are based on results by Shalev-Shwartz (2007), Kakade et al. (2008), and Juditsky and Nemirovski (2008). Nevertheless, we believe that the presentation given here is simpler and slightly more general.

Our results are based on basic notions from convex analysis and matrix computation. The reader not familiar with some of the objects described below may find short explanations in Appendix A.

2.1 Notation

We consider convex functions $f : \mathcal{X} \to \mathbb{R} \cup \{\infty\}$, where $\mathcal{X}$ is a Euclidean vector space equipped with an inner product $\langle \cdot, \cdot \rangle$. We denote $\mathbb{R}^* = \mathbb{R} \cup \{\infty\}$. The subdifferential of f at $x \in \mathcal{X}$ is denoted by $\partial f(x)$. The Fenchel conjugate of f is denoted by $f^\star$. Given a norm $\|\cdot\|$, its dual norm is denoted by $\|\cdot\|_\star$. We say that a convex function f is V-Lipschitz w.r.t. a norm $\|\cdot\|$ if for all $x, x' \in \mathcal{X}$ we have $|f(x) - f(x')| \leq V\|x - x'\|$. Of particular interest are p-norms, $\|x\|_p = (\sum_i |x_i|^p)^{1/p}$.

When dealing with matrices, we consider the vector space $\mathcal{X} = \mathbb{R}^{m \times n}$ of real matrices of size $m \times n$ and the vector space $\mathcal{X} = \mathbb{S}^n$ of symmetric matrices of size $n \times n$, both equipped with the inner product, $\langle \mathbf{X}, \mathbf{Y} \rangle := \mathrm{Tr}(\mathbf{X}^\top \mathbf{Y})$. Given a matrix $\mathbf{X}$, the vector $\sigma(\mathbf{X})$ is the vector that contains the singular values of $\mathbf{X}$ in a non-increasing order. For $\mathbf{X} \in \mathbb{S}^n$, the vector $\lambda(\mathbf{X})$ is the vector that contains the eigenvalues of $\mathbf{X}$ arranged in non-increasing order.

2.2 Strong Convexity–Strong Smoothness Duality

Recall that the domain of $f : \mathcal{X} \to \mathbb{R}^*$ is $\{x : f(x) < \infty\}$ (allowing f to take infinite values is the effective way to restrict its domain to a proper subset of $\mathcal{X}$). We first define strong convexity.

Definition 1 *A function $f : \mathcal{X} \to \mathbb{R}^*$ is β-strongly convex w.r.t. a norm $\|\cdot\|$ if for all x, y in the relative interior of the domain of f and $\alpha \in (0,1)$ we have*

$$f(\alpha x + (1-\alpha)y) \leq \alpha f(x) + (1-\alpha) f(y) - \tfrac{1}{2}\beta\alpha(1-\alpha)\|x-y\|^2.$$

We now define strong smoothness. Note that a strongly smooth function f is always finite.

Definition 2 *A function $f : \mathcal{X} \to \mathbb{R}$ is β-strongly smooth w.r.t. a norm $\|\cdot\|$ if f is everywhere differentiable and if for all x, y we have*

$$f(x+y) \leq f(x) + \langle \nabla f(x), y \rangle + \tfrac{1}{2}\beta\|y\|^2.$$

The following theorem states that strong convexity and strong smoothness are dual properties. Recall that the biconjugate $f^{\star\star}$ equals f if and only if f is closed and convex.

Theorem 3 *(Strong Convexity/Strong Smoothness Duality) Assume that f is a closed and convex function. Then f is β-strongly convex w.r.t. a norm $\|\cdot\|$ if and only if $f^\star$ is $\frac{1}{\beta}$-strongly smooth w.r.t. the dual norm $\|\cdot\|_\star$.*

Subtly, note that while the domain of a strongly convex function f may be a proper subset of $\mathcal{X}$ (important for a number of settings), its conjugate $f^\star$ always has a domain which is $\mathcal{X}$ (since if $f^\star$ is strongly smooth then it is finite and everywhere differentiable). The above theorem can be found, for instance, in Zalinescu (2002) (see Corollary 3.5.11 on p. 217 and Remark 3.5.3 on p. 218). In the machine learning literature, a proof of one direction (strong convexity $\Rightarrow$ strong smoothness) can be found in Shalev-Shwartz (2007). We could not find a proof of the reverse implication in a place easily accessible to machine learning researchers. So, a self-contained proof is provided in the appendix.

The following direct corollary of Theorem 3 is central in proving both regret and generalization bounds.

Corollary 4 *If f is β strongly convex w.r.t. $\|\cdot\|$ and $f^\star(\mathbf{0}) = 0$, then, denoting the partial sum $\sum_{j \leq i} v_j$ by $v_{1:i}$, we have, for any sequence $v_1, \ldots, v_n$ and for any u,*

$$\sum_{i=1}^n \langle v_i, u \rangle - f(u) \leq f^\star(v_{1:n}) \leq \sum_{i=1}^n \langle \nabla f^\star(v_{1:i-1}), v_i \rangle + \frac{1}{2\beta}\sum_{i=1}^n \|v_i\|_\star^2 \,.$$

Proof The first inequality is Fenchel-Young and the second follows from an easy induction and the definition of smoothness. ■

2.3 Implications of Strong-convexity / Strong-smoothness Duality

We consider two well-known learning models.

- **Online convex optimization:** Let $\mathcal{W}$ be a convex set. Online convex optimization is a two player repeated game. On round t of the game, the learner (first player) should choose $w_t \in \mathcal{W}$ and the environment (second player) responds with a convex function over $\mathcal{W}$, that is, $l_t : \mathcal{W} \to \mathbb{R}$. The goal of the learner is to minimize its regret defined as:

$$\frac{1}{n}\sum_{t=1}^n l_t(w_t) - \min_{w \in \mathcal{W}} \frac{1}{n}\sum_{t=1}^n l_t(w) \,.$$

- **Batch learning of linear predictors:** Let $\mathcal{D}$ be a distribution over $\mathcal{X} \times \mathcal{Y}$ which is unknown. The goal is to learn a prediction rule from $\mathcal{X}$ to $\mathcal{Y}$. The prediction rule we use is based on a linear mapping $x \mapsto \langle w, x \rangle$, and the quality of the prediction is assessed by a loss function $l(\langle w, x \rangle, y)$. Our primary goal is to find w that has low risk (a.k.a. generalization error), defined as $L(w) = \mathbb{E}[l(\langle w, x \rangle, y)]$, where expectation is with respect to $\mathcal{D}$. To do so, we can sample n i.i.d. examples from $\mathcal{D}$ and observe the empirical risk, $\hat{L}(w) = \frac{1}{n} \sum_{i=1}^{n} l(\langle w, x_i \rangle, y_i)$. The goal of the learner is to find $\hat{w}$ with a low excess risk defined as:

$$L(\hat{w}) - \min_{w \in \mathcal{W}} L(w) \ ,$$

  where $\mathcal{W}$ is a set of vectors that forms the comparison class.

We now seamlessly provide learning guarantees for both models based on Corollary 4. We start with the online convex optimization model.

2.3.1 REGRET BOUND FOR ONLINE CONVEX OPTIMIZATION

Algorithm 1 provides one common algorithm (online mirror descent) which achieves the following regret bound. It is one of a family of algorithms that enjoy the same regret bound (see Shalev-Shwartz, 2007). Note that successfully implementing the algorithm requires the ability to efficiently compute the subgradient ∂l_t of the loss function and the gradient mapping $\nabla f^\star$.

Theorem 5 *(Regret) Suppose Algorithm 1 is used with a function f that is β-strongly convex w.r.t. a norm $\|\cdot\|$ on $\mathcal{W}$ and that $f^\star(\mathbf{0}) = 0$. Suppose the loss functions l_t are convex and V-Lipschitz w.r.t. the dual norm $\|\cdot\|_\star$. Then, the algorithm run with any positive η enjoys the regret bound,*

$$\sum_{t=1}^{T} l_t(w_t) - \min_{u \in \mathcal{W}} \sum_{t=1}^{T} l_t(u) \leq \frac{\max_{u \in \mathcal{W}} f(u)}{\eta} + \frac{\eta V^2 T}{2\beta} \ .$$

Proof Apply Corollary 4 to the sequence $-\eta v_1, \ldots, -\eta v_T$ to get, for all u,

$$-\eta \sum_{t=1}^{T} \langle v_t, u \rangle - f(u) \leq -\eta \sum_{t=1}^{T} \langle v_t, w_t \rangle + \frac{1}{2\beta} \sum_{t=1}^{T} \|\eta v_t\|_\star^2 \ .$$

Using the fact that l_t is V-Lipschitz, we get $\|v_t\|_\star \leq V$. Plugging this into the inequality above and rearranging gives, $\sum_{t=1}^{T} \langle v_t, w_t - u \rangle \leq \frac{f(u)}{\eta} + \frac{\eta V^2 T}{2\beta}$. By convexity of l_t, $l_t(w_t) - l_t(u) \leq \langle v_t, w_t - u \rangle$. Therefore, $\sum_{t=1}^{T} l_t(w_t) - \sum_{t=1}^{T} l_t(u) \leq \frac{f(u)}{\eta} + \frac{\eta V^2 T}{2\beta}$. Since the above holds for all $u \in \mathcal{W}$ the result follows. ■

Remark 6 *The result above assumes strong convexity and derives an $O(\sqrt{T})$ rate (once we optimize over η). It seems that strong convexity is required to get such rates in general. In particular, when the "weight vector" comes from an infinite dimensional normed space with norm $\|\cdot\|$ and the functions l_t are 1-Lipschitz w.r.t. the dual norm $\|\cdot\|_\star$, a necessary and sufficient condition for a $O(\sqrt{T})$ regret rate is that the* Martingale type *of the dual space is 2. Moreover, this last condition is equivalent to the existence of a strongly convex function in the original normed space. For details, we refer the interested reader to the work of Sridharan and Tewari (2010).*

Algorithm 1 Online Mirror Descent

$w_1 \leftarrow \nabla f^\star(\mathbf{0})$
for $t = 1$ to T **do**
  Play $w_t \in \mathcal{W}$
  Receive l_t and pick $v_t \in \partial l_t(w_t)$
  $w_{t+1} \leftarrow \nabla f^\star\left(-\eta \sum_{s=1}^t v_t\right)$
end for

2.3.2 GENERALIZATION BOUND FOR THE BATCH MODEL VIA RADEMACHER COMPLEXITY

Let $\mathcal{T} = (x_1, \ldots, x_n) \in \mathcal{X}^n$ be the examples in a training set obtained by sampling i.i.d. examples from $\mathcal{D}$. For a class of real valued functions $\mathcal{F} \subseteq \mathbb{R}^{\mathcal{X}}$, define its Rademacher complexity on $\mathcal{T}$ to be

$$\mathcal{R}_{\mathcal{T}}(\mathcal{F}) := \mathbb{E}\left[\sup_{f\in\mathcal{F}} \frac{1}{n}\sum_{i=1}^n \varepsilon_i f(x_i)\right] .$$

Here, the expectation is over ε_i's, which are i.i.d. Rademacher random variables, that is, $\mathbb{P}(\varepsilon_i = -1) = \mathbb{P}(\varepsilon_1 = +1) = \frac{1}{2}$. It is well known that bounds on Rademacher complexity of a class immediately yield generalization bounds for classifiers picked from that class (assuming the loss function is Lipschitz). Recently, Kakade et al. (2008) proved Rademacher complexity bounds for classes consisting of linear predictors using strong convexity arguments. We now give a quick proof of their main result using Corollary 4. This proof is essentially the same as their original proof but highlights the importance of Corollary 4.

Theorem 7 *(Generalization) Let f be a β-strongly convex function w.r.t. a norm $\|\cdot\|$ and assume that $f^\star(\mathbf{0}) = 0$. Let $\mathcal{X} = \{x : \|x\|_\star \le X\}$ and $\mathcal{W} = \{w : f(w) \le f_{\max}\}$. Consider the class of linear functions, $\mathcal{F} = \{x \mapsto \langle w, x\rangle : w \in \mathcal{W}\}$. Then, for any data set $\mathcal{T} \in \mathcal{X}^n$, we have*

$$\mathcal{R}_{\mathcal{T}}(\mathcal{F}) \le X\sqrt{\frac{2f_{\max}}{\beta n}} .$$

Proof Let $\lambda > 0$. Apply Corollary 4 with $u = w$ and $v_i = \lambda\varepsilon_i x_i$ to get,

$$\begin{aligned}\sup_{w\in\mathcal{W}} \sum_{i=1}^n \langle w, \lambda\varepsilon_i x_i\rangle &\le \frac{\lambda^2}{2\beta}\sum_{i=1}^n \|\varepsilon_i x_i\|_\star^2 + \sup_{w\in\mathcal{W}} f(w) + \sum_{i=1}^n \langle \nabla f^\star(v_{1:i-1}), \varepsilon_i x_i\rangle \\ &\le \frac{\lambda^2 X^2 n}{2\beta} + f_{\max} + \sum_{i=1}^n \langle \nabla f^\star(v_{1:i-1}), \varepsilon_i x_i\rangle .\end{aligned}$$

Now take expectation on both sides. The left hand side is $n\lambda\mathcal{R}_{\mathcal{T}}(\mathcal{F})$ and the last term on the right hand side becomes zero. Dividing throughout by $n\lambda$, we get, $\mathcal{R}_{\mathcal{T}}(\mathcal{F}) \le \frac{\lambda X^2}{2\beta} + \frac{f_{\max}}{n\lambda}$. Optimizing over λ gives us the result. ■

Combining the above with the contraction lemma and standard Rademacher based generalization bounds (see, e.g., Bartlett and Mendelson, 2002; Kakade et al., 2008) we obtain the following result.

Corollary 8 *Let f be a β-strongly convex function w.r.t. a norm $\|\cdot\|$ and assume that $f^\star(\mathbf{0}) = 0$. Let $\mathcal{X} = \{x : \|x\|_\star \le X\}$ and $\mathcal{W} = \{w : f(w) \le f_{\max}\}$. Let l be an ρ-Lipschitz scalar loss function and let $\mathcal{D}$ be an arbitrary distribution over $\mathcal{X} \times \mathcal{Y}$. Then, the algorithm that receives n i.i.d. examples and returns $\hat{w}$ that minimizes the empirical risk, $\hat{L}(w)$, satisfies*

$$\mathbb{E}\left[L(\hat{w}) - \min_{w \in \mathcal{W}} L(w)\right] \;\le\; 2\rho X \sqrt{\frac{2 f_{\max}}{\beta n}}\,,$$

where expectation is with respect to the choice of the n i.i.d. examples.

We note that it is also easy to obtain a generalization bound that holds with high probability, but for simplicity of the presentation we stick to expectations.

2.4 Strongly Convex Matrix Functions

Before we consider strongly convex matrix functions, let us recall the following result about strong convexity of vector ℓ_q norm. Its proof is standard and can be found, for example, in the work of Shalev-Shwartz (2007).

Lemma 9 *Let $q \in [1,2]$. The function $f : \mathbb{R}^d \to \mathbb{R}$ defined as $f(w) = \frac{1}{2}\|w\|_q^2$ is $(q-1)$-strongly convex with respect to $\|\cdot\|_q$ over $\mathbb{R}^d$.*

We mainly use the above lemma to obtain results with respect to the norms $\|\cdot\|_2$ and $\|\cdot\|_1$. The case $q = 2$ is straightforward. Obtaining results with respect to $\|\cdot\|_1$ is slightly more tricky since for $q = 1$ the strong convexity parameter is 0 (meaning that the function is not strongly convex). To overcome this problem, we shall set q to be slightly more than 1, for example, $q = \ln(d)/(\ln(d)-1)$. For this choice of q, the strong convexity parameter becomes $q - 1 = 1/(\ln(d)-1) \ge 1/\ln(d)$ and the value of p corresponding to the dual norm is $p = (1-1/q)^{-1} = \ln(d)$. Note that for any $x \in \mathbb{R}^d$ we have

$$\|x\|_\infty \le \|x\|_p \le (d\|x\|_\infty^p)^{1/p} = d^{1/p}\|x\|_\infty = e\,\|x\|_\infty \le 3\,\|x\|_\infty\,.$$

Hence the dual norms are also equivalent up to a factor of 3: $\|w\|_1 \ge \|w\|_q \ge \|w\|_1/3$. The above lemma therefore implies the following corollary.

Corollary 10 *The function $f : \mathbb{R}^d \to \mathbb{R}$ defined as $f(w) = \frac{1}{2}\|w\|_q^2$ for $q = \frac{\ln(d)}{\ln(d)-1}$ is $1/(3\ln(d))$-strongly convex with respect to $\|\cdot\|_1$ over $\mathbb{R}^d$.*

We now consider two families of strongly convex matrix functions.

2.4.1 SCHATTEN q-NORMS

The first result we need is the counterpart of Lemma 9 for the q-Schatten norm defined as $\|\mathbf{X}\|_{S(q)} := \|\sigma(\mathbf{X})\|_q$ (see Argyriou et al., 2010 for some recent work involving general Schatten norms). This extremely useful result can be found in Ball et al. (1994). The proof there uses complex analysis. We are not aware of an elementary proof.

Theorem 11 *(Schatten matrix functions) Let $q \in [1,2]$. The function $F : \mathbb{R}^{m\times n} \to \mathbb{R}$ defined as $F(\mathbf{X}) = \frac{1}{2}\|\sigma(\mathbf{X})\|_q^2$ is $(q-1)$-strongly convex w.r.t. the q-Schatten norm $\|\mathbf{X}\|_{S(q)} := \|\sigma(\mathbf{X})\|_q$ over $\mathbb{R}^{m\times n}$.*

As above, choosing q to be $\frac{\ln m'}{\ln(m')-1}$ for $m' = \min\{m,n\}$ gives the following corollary.

Corollary 12 *The function $F : \mathbb{R}^{m\times n} \to \mathbb{R}$ defined as $F(\mathbf{W}) = \frac{1}{2}\|\mathbf{W}\|^2_{S(q)}$ for $q = \frac{\ln(m')}{\ln(m')-1}$ is $1/(3\ln(m'))$-strongly convex with respect to $\|\cdot\|_{S(1)}$ over $\mathbb{R}^{m\times n}$.*

2.4.2 GROUP NORMS

Let $\mathbf{X} = (\mathbf{X}^1\mathbf{X}^2\ldots\mathbf{X}^n)$ be a $m\times n$ real matrix with *columns* $\mathbf{X}^i \in \mathbb{R}^m$. Given two (vector) norms $\|\cdot\|_r$ and $\|\cdot\|_p$, we can define a new norm $\|\mathbf{X}\|_{r,p}$ as

$$\|\mathbf{X}\|_{r,p} := \|\left(\|\mathbf{X}^1\|_r, \ldots, \|\mathbf{X}^n\|_r\right)\|_p \ .$$

That is, we apply $\|\cdot\|_r$ to each column of $\mathbf{X}$ to get a vector in $\mathbb{R}^n$ to which we apply the norm $\|\cdot\|_p$ to get the value of $\|\mathbf{X}\|_{r,p}$. It is easy to check that this is indeed a norm. The dual of $\|\cdot\|_{r,p}$ is $\|\cdot\|_{s,t}$ where $1/r+1/s = 1$ and $1/p+1/t = 1$. The following theorem, which appears in a slightly weaker form in the work of Juditsky and Nemirovski (2008), provides us with an easy way to construct strongly convex group norms. We provide a proof in the appendix which is much simpler than that of Juditsky and Nemirovski (2008) and is completely "calculus free".

Theorem 13 *(Group Norms) Let Ψ,Φ be absolutely symmetric norms on $\mathbb{R}^m,\mathbb{R}^n$. Let $\Phi^2 \circ \sqrt{} : \mathbb{R}^n \to \mathbb{R}^*$ denote the following function,*

$$(\Phi^2 \circ \sqrt{})(x) := \Phi^2(\sqrt{x_1}, \ldots, \sqrt{x_n}) \ . \tag{1}$$

Suppose, $(\Phi^2 \circ \sqrt{})$ is a norm on $\mathbb{R}^n$. Further, let the functions Ψ^2 and Φ^2 be σ_1- and σ_2-smooth w.r.t. Ψ and Φ respectively. Then, $\|\cdot\|^2_{\Psi,\Phi}$ is $(\sigma_1+\sigma_2)$-smooth w.r.t. $\|\cdot\|_{\Psi,\Phi}$.

The condition that the function defined in (1) be a norm may appear strange but in fact it already occurs in the matrix analysis literature. Norms satisfying it are called *quadratic symmetric gauge* functions (or Q-norms) (Bhatia, 1997, p. 89). It is easy to see that $\|\cdot\|_p$ for $p \geq 2$ is a Q-norm. Now using strong convexity/strong smoothness duality and the discussion preceding Corollary 10, we get the following corollary.

Corollary 14 *The function $F : \mathbb{R}^{m\times n} \to \mathbb{R}$ defined as $F(\mathbf{W}) = \frac{1}{2}\|\mathbf{W}\|^2_{2,q}$ for $q = \frac{\ln(n)}{\ln(n)-1}$ is $1/(3\ln(n))$-strongly convex with respect to $\|\cdot\|_{2,1}$ over $\mathbb{R}^{m\times n}$.*

2.5 Putting it All Together

Combining Lemma 9 and Corollary 10 with the bounds given in Theorem 5 and Corollary 8 we obtain the following two corollaries.

Corollary 15 *Let $\mathcal{W} = \{w \in \mathbb{R}^d : \|w\|_1 \leq W\}$ and let $l_1,\ldots,l_n$ be a sequence of convex functions that are X-Lipschitz w.r.t. $\|\cdot\|_\infty$. Then, online mirror descent run with $f(w) = \frac{1}{2}\|w\|^2_q$ for $q = \ln(d)/(\ln(d)-1)$ achieves the following regret bound:*

$$\frac{1}{n}\sum_{t=1}^n l_t(w_t) - \min_{w\in\mathcal{W}} \frac{1}{n}\sum_{t=1}^n l_t(w) \ \leq \ XW\sqrt{\frac{3\ln(d)}{n}} \ .$$

Corollary 16 *Let $\mathcal{W} = \{w : \|w\|_1 \leq W\}$ and let $\mathcal{X} = \{x \in \mathbb{R}^d : \|x\|_\infty \leq X\}$. Let l be an ρ-Lipschitz scalar loss function and let $\mathcal{D}$ be an arbitrary distribution over $\mathcal{X} \times \mathcal{Y}$. Then, any minimizer $\hat{w}$ of the empirical risk*

$$\hat{L}(w) = \frac{1}{n}\sum_{i=1}^{n} l(\langle w, x_i \rangle, y_i)$$

satisfies

$$\mathbb{E}\left[L(\hat{w}) - \min_{w \in \mathcal{W}} L(w)\right] \;\leq\; 2\rho X W \sqrt{\frac{3\ln(d)}{n}}\,.$$

Results of the same flavor can be obtained for learning matrices. For simplicity, we present the following two corollaries only for the online model, but it is easy to derive their batch counterparts.

Corollary 17 *Let $\mathcal{W} = \{\mathbf{W} \in \mathbb{R}^{k\times d} : \|\mathbf{W}\|_{2,1} \leq W\}$ and let $l_1, \ldots, l_n$ be a sequence of functions that are X-Lipschitz w.r.t. $\|\cdot\|_{2,\infty}$. Then, online mirror descent run with $f(\mathbf{W}) = \frac{1}{2}\|\mathbf{W}\|^2_{2,q}$ for $q = \ln(d)/(\ln(d)-1)$ achieves the following regret bound:*

$$\frac{1}{n}\sum_{t=1}^{n} l_t(\mathbf{W}_t) - \min_{\mathbf{W}\in\mathcal{W}} \frac{1}{n}\sum_{t=1}^{n} l_t(\mathbf{W}) \;\leq\; X W \sqrt{\frac{3\ln(d)}{n}}\,.$$

Corollary 18 *Let $\mathcal{W} = \{\mathbf{W} \in \mathbb{R}^{k\times d} : \|\mathbf{W}\|_{S(1)} \leq W\}$ and let $l_1, \ldots, l_n$ be a sequence of functions that are X-Lipschitz w.r.t. $\|\cdot\|_{S(\infty)}$. Then, online mirror descent run with $f(\mathbf{W}) = \frac{1}{2}\|\mathbf{W}\|^2_{S(q)}$ for $q = \ln(k')/(\ln(k')-1)$ achieves the following regret bound:*

$$\frac{1}{n}\sum_{t=1}^{n} l_t(\mathbf{W}_t) - \min_{\mathbf{W}\in\mathcal{W}} \frac{1}{n}\sum_{t=1}^{n} l_t(\mathbf{W}) \;\leq\; X W \sqrt{\frac{3\ln(k')}{n}}\,,$$

where $k' = \min\{k, d\}$.

Remark 19 *The gradient mapping $\mathbf{W} \mapsto \nabla f^\star(\mathbf{W})$ that is needed to run the online mirror descent algorithm can be easily computed both for the group norm ($f(\mathbf{W}) = \frac{1}{2}\|\mathbf{W}\|^2_{r,p}$) and Schatten norm ($f(\mathbf{W}) = \frac{1}{2}\|\mathbf{W}\|^2_{S(q)}$) cases. In the latter case, a singular value decomposition of $\mathbf{W}$ will be needed (see Theorem 30 in the Appendix).*

3. Matrix Regularization

We are now ready to demonstrate the power of the general techniques we derived in the previous section. Consider a learning problem (either online or batch) in which $\mathcal{X}$ is a subset of a matrix space (of dimension $k \times d$) and we would like to learn a linear predictor of the form $\mathbf{X} \mapsto \langle \mathbf{W}, \mathbf{X} \rangle$ where $\mathbf{W}$ is also a matrix of the same dimension. The loss function takes the form $l(\langle \mathbf{W}, \mathbf{X} \rangle, y)$ and we assume for simplicity that l is 1-Lipschitz with respect to its first argument. For example, l can be the absolute loss, $l(a,y) = |a-y|$, or the hinge-loss, $l(a,y) = \max\{0, 1-ya\}$.

For the sake of concreteness, let us focus on the batch learning setting, but we note that the discussion below is relevant to the online learning model as well. Our prior knowledge on the learning problem is encoded by the definition of the comparison class $\mathcal{W}$ that we use. In particular,

all the comparison classes we use take the form $\mathcal{W} = \{\mathbf{W} : \|\mathbf{W}\| \leq W\}$, where the only difference is what norm do we use. We shall compare the following four classes:

$$\mathcal{W}_{1,1} = \{\mathbf{W} : \|\mathbf{W}\|_{1,1} \leq W_{1,1}\}\ , \qquad \mathcal{W}_{2,2} = \{\mathbf{W} : \|\mathbf{W}\|_{2,2} \leq W_{2,2}\}\ ,$$
$$\mathcal{W}_{2,1} = \{\mathbf{W} : \|\mathbf{W}\|_{2,1} \leq W_{2,1}\}\ , \qquad \mathcal{W}_{S(1)} = \{\mathbf{W} : \|\mathbf{W}\|_{S(1)} \leq W_{S(1)}\}\ .$$

Let us denote $X_{\infty,\infty} = \sup_{x\in\mathcal{X}} \|\mathbf{X}\|_{\infty,\infty}$. We define $X_{2,2}, X_{2,\infty}, X_{S(\infty)}$ analogously. Applying the results of the previous section to these classes we obtain the bounds given in Table 1 where for simplicity we ignore constants.

class	$\mathcal{W}_{1,1}$	$\mathcal{W}_{2,2}$	$\mathcal{W}_{2,1}$	$\mathcal{W}_{S(1)}$
bound	$W_{1,1} X_{\infty,\infty} \sqrt{\frac{\ln(kd)}{n}}$	$W_{2,2} X_{2,2} \sqrt{\frac{1}{n}}$	$W_{2,1} X_{2,\infty} \sqrt{\frac{\ln(d)}{n}}$	$W_{S(1)} X_{S(\infty)} \sqrt{\frac{\ln(\min\{d,k\})}{n}}$

Table 1: List of bounds for learning with matrices. For simplicity we ignore constants.

Let us now discuss which class should be used based on prior knowledge on properties of the learning problem. We start with the well known difference between $\mathcal{W}_{1,1}$ and $\mathcal{W}_{2,2}$. Note that both of these classes ignore the fact that $\mathbf{W}$ is organized as a $k \times d$ matrix and simply refer to $\mathbf{W}$ as a single vector of dimension kd. The difference between $\mathcal{W}_{1,1}$ and $\mathcal{W}_{2,2}$ is therefore the usual difference between ℓ_1 and ℓ_2 regularization. To understand this difference, suppose that $\mathbf{W}$ is some matrix that performs well on the distribution we have. Then, we should take the radius of each class to be the minimal possible while still containing $\mathbf{W}$, namely, either $\|\mathbf{W}\|_{1,1}$ or $\|\mathbf{W}\|_{2,2}$. Clearly, $\|\mathbf{W}\|_{2,2} \leq \|\mathbf{W}\|_{1,1}$ and therefore in terms of this term there is a clear advantage to use the class $\mathcal{W}_{2,2}$. On the other hand, $X_{2,2} \geq X_{\infty,\infty}$. We therefore need to understand which of these inequalities is more important. Of course, in general, the answer to this question is data dependent. However, we can isolate properties of the distribution that can help us choose the better class.

One useful property is sparsity of either $\mathbf{X}$ or $\mathbf{W}$. If $\mathbf{X}$ is assumed to be s sparse (i.e., it has at most s non-zero elements), then we have $X_{2,2} \leq \sqrt{s} X_{\infty,\infty}$. That is, for a small s, the difference between $X_{2,2}$ and $X_{\infty,\infty}$ is small. In contrast, if $\mathbf{X}$ is very dense and each of its entries is bounded away from zero, for example, $\mathcal{X} \in \{\pm 1\}^{k\times d}$, then $\|\mathbf{X}\|_{2,2} = \sqrt{kd}\|\mathbf{X}\|_{\infty,\infty}$. The same arguments are true for $\mathbf{W}$. Hence, with prior knowledge about the sparsity of $\mathbf{X}$ and $\mathbf{W}$, we can guess which of the bounds will be smaller.

Next, we tackle the more interesting cases of $\mathcal{W}_{2,1}$ and $\mathcal{W}_{S(1)}$. For the former, recall that we first apply ℓ_2 norm on each column of $\mathbf{W}$ and then apply ℓ_1 norm on the obtained vector of norm values. Similarly, to calculate $\|\mathbf{X}\|_{2,\infty}$ we first apply ℓ_2 norm on columns of $\mathbf{X}$ and then apply ℓ_∞ norm on the obtained vector of norm values. Let us now compare $\mathcal{W}_{2,1}$ to $\mathcal{W}_{1,1}$. Suppose that the columns of $\mathbf{X}$ are very sparse. Therefore, the ℓ_2 norm of each column of $\mathbf{X}$ is very close to its ℓ_∞ norm. On the other hand, if some of the columns of $\mathbf{W}$ are dense, then $\|\mathbf{W}\|_{2,1}$ can be order of $\sqrt{k}$ smaller than $\|\mathbf{W}\|_{1,1}$. In that case, the class $\mathcal{W}_{2,1}$ is preferable over the class $\mathcal{W}_{1,1}$. As we show later, this is the case in multi-class problems, and we shall indeed present an improved multi-class algorithm that uses the class $\mathcal{W}_{2,1}$. Of course, in some problems, columns of $\mathbf{X}$ might be very dense while columns of $\mathbf{W}$ can be sparse. In such cases, using $\mathcal{W}_{1,1}$ is better than using $\mathcal{W}_{2,1}$.

Now lets compare $\mathcal{W}_{2,1}$ to $\mathcal{W}_{2,2}$. Similarly to the previous discussion, choosing $\mathcal{W}_{2,1}$ over $\mathcal{W}_{2,2}$ makes sense if we assume that the vector of ℓ_2 norms of columns, $(\|\mathbf{W}^1\|_2,\ldots,\|\mathbf{W}^d\|_2)$, is sparse. This implies that we assume a "group"-sparsity pattern of $\mathbf{W}$, that is, each column of $\mathbf{W}$ is either the all zeros column or is dense. This type of grouped-sparsity has been studied in the context of group Lasso and multi-task learning (Argyriou et al., 2008). Indeed, we present bounds for multi-task learning that relies on this assumption. Without the group-sparsity assumption, it might be better to use $\mathcal{W}_{2,2}$ over $\mathcal{W}_{2,1}$.

Finally, we discuss when it makes sense to use $\mathcal{W}_{S(1)}$ (see Srebro et al., 2005 for one of the early applications of this norm in machine learning). Recall that $\|\mathbf{W}\|_{S(1)} = \|\sigma(\mathbf{W})\|_1$, where $\sigma(\mathbf{W})$ is the vector of singular values of $\mathbf{W}$, and $\|\mathbf{X}\|_{S(\infty)} = \|\sigma(\mathbf{X})\|_\infty$. Therefore, the class $\mathcal{W}_{S(1)}$ should be used when we assume that the *spectrum* of $\mathbf{W}$ is sparse while the spectrum of $\mathbf{X}$ is dense. This means that the prior knowledge we employ is that $\mathbf{W}$ is of low rank while $\mathcal{X}$ is of high rank. Note that $\mathcal{W}_{2,2}$ can be defined equivalently as $\mathcal{W}_{S(2)}$. Therefore, the difference between $\mathcal{W}_{S(1)}$ and $\mathcal{W}_{2,2}$ is similar to the difference between $\mathcal{W}_{1,1}$ and $\mathcal{W}_{2,2}$ just that instead of considering sparsity properties of the elements of $\mathbf{W}$ and $\mathbf{X}$ we consider sparsity properties of the spectrum of $\mathbf{W}$ and $\mathbf{X}$.

In the next sections we demonstrate how to apply the general methodology described above in order to derive a few generalization and regret bounds for problems of recent interest.

4. Multi-task Learning

Suppose we are simultaneously solving k-multivariate prediction problems, where each learning example is of the form $(\mathbf{X},\mathbf{y})$ where $\mathbf{X}\in\mathcal{X}\subseteq\mathbb{R}^{k\times d}$ is a matrix of example vectors with examples from different tasks sitting in rows of $\mathbf{X}$, and $\mathbf{y}\in\mathbb{R}^k$ are the responses for the k problems. To predict the k responses, we learn a matrix $\mathbf{W}\in\mathbb{R}^{k\times d}$ such that $\mathrm{Diag}(\mathbf{W}^\top\mathbf{X})$ is a good predictor of $\mathbf{y}$. In this section, we denote *row* j of $\mathbf{W}$ by $\mathbf{w}^j$. The predictor for the jth task is therefore $\mathbf{w}^j$. The quality of a prediction $\langle\mathbf{w}^j,\mathbf{x}^j\rangle$ for the j'th task is assessed by a loss function $l^j:\mathbb{R}\times\mathcal{Y}^j\to\mathbb{R}$. The total loss of $\mathbf{W}$ on an example $(\mathbf{X},\mathbf{y})$ is defined to be the sum of the individual losses,

$$l(\mathbf{W},\mathbf{X},\mathbf{y}) = \sum_{j=1}^{k} l^j(\langle\mathbf{w}^j,\mathbf{x}^j\rangle,y^j)\ .$$

This formulation allows us to mix regression and classification problems and even use different loss functions for different tasks. Such "heterogeneous" multi-task learning has attracted recent attention (Yang et al., 2009).

If the tasks are related, then it is natural to use regularizers that "couple" the tasks together so that similarities across tasks can be exploited. Considerations of common sparsity patterns (same features relevant across different tasks) lead to the use of group norm regularizers (i.e., using the comparison class $\mathcal{W}_{2,1}$ defined in the previous section) while rank considerations (the $\mathbf{w}^j$'s lie in a low dimensional linear space) lead to the use of unitarily invariant norms as regularizers (i.e., the comparison class is $\mathcal{W}_{S(1)}$).

We now describe online and batch multi-task learning using various matrix norms.

Algorithm 2 Online Multi-task Mirror Descent

$\mathbf{W}_1 \leftarrow \nabla f^\star(\mathbf{0})$
for $t = 1$ to T **do**
  Predict using $\mathbf{W}_t \in \mathcal{W} \subseteq \mathbb{R}^{k\times d}$
  Receive $\mathbf{X}_t \in \mathbb{R}^{k\times d}$, $\mathbf{y}_t \in \mathbb{R}^k$
  Pick $\mathbf{V}_t$ whose row j is $\mathbf{v}_t^j = (l^j)'\left(\left\langle \mathbf{w}_t^j, \mathbf{x}_t^j \right\rangle, y_t^j\right)\mathbf{x}_t^j$
  $\mathbf{W}_{t+1} \leftarrow \nabla f^\star\left(-\eta \sum_{s=1}^t \mathbf{V}_t\right)$
end for

4.1 Online Multi-task Learning

In the online model, on round t the learner first uses $\mathbf{W}_t$ to predict the vector of responses and then it pays the cost

$$l_t(\mathbf{W}_t) = l(\mathbf{W}_t, \mathbf{X}_t, \mathbf{y}_t) = \sum_{j=1}^k l^j\left(\left\langle \mathbf{w}_t^j, \mathbf{x}_t^j \right\rangle, y_t^j\right).$$

Let $\mathbf{V}_t \in \mathbb{R}^{k\times d}$ be a sub-gradient of l_t at $\mathbf{W}_t$. It is easy to verify that the j'th row of $\mathbf{V}_t$, denoted $\mathbf{v}_t^j$, is a sub-gradient of $l^j\left(\left\langle \mathbf{w}_t^j, \mathbf{x}_t^j \right\rangle, y_t^j\right)$ at $\mathbf{w}_t^j$. Assuming that l^j is ρ-Lipschitz with respect to its first argument, we obtain that $\mathbf{v}_t^j = \tau_t^j \mathbf{x}_t^j$ for some $\tau_t^j \in [-\rho, \rho]$. In other words, $\mathbf{V}_t = \mathrm{Diag}(\tau_t)\mathbf{X}_t$. It is easy to verify that $\|\mathbf{V}_t\|_{r,p} \le \rho \|\mathbf{X}\|_{r,p}$ for any $r, p \ge 1$. In addition, since any Schatten norm is sub-multiplicative we also have that $\|\mathbf{V}_t\|_{S(\infty)} \le \|\mathrm{Diag}(\tau_t)\|_{S(\infty)} \|\mathbf{X}_t\|_{S(\infty)} \le \rho \|\mathbf{X}_t\|_{S(\infty)}$.

Algorithm 2 is simply the instantiation of Algorithm 1 for the multi-task setting. Using $f(\mathbf{W})$ to be one of the functions $\frac{1}{2}\|\mathbf{W}\|^2_{q_1,q_1}$, $\frac{1}{2}\|\mathbf{W}\|^2_{2,2}$, $\frac{1}{2}\|\mathbf{W}\|^2_{2,q_2}$, $\frac{1}{2}\|\mathbf{W}\|^2_{S(q_3)}$, for appropriate choices[1] of q_1, q_2, q_3, we obtain the following result.

Corollary 20 *Let $\mathcal{W}_{1,1}, \mathcal{W}_{2,2}, \mathcal{W}_{2,1}, \mathcal{W}_{S(1)}$ be the classes defined in Section 3 and let $X_{\infty,\infty}$, $X_{2,2}$,$X_{2,\infty}$, $X_{S(\infty)}$ be the radii of $\mathcal{X}$ w.r.t. the corresponding dual norms. Then, Algorithm 2 achieves with regret bounds given in Table 1 (ignoring constants).*

Let us now discuss few implications of these bounds,[2] and for simplicity assume that $k < d$. Recall that each column of $\mathbf{X}$ represents the value of a single feature for all the tasks. As discussed in the previous section, if the matrix $\mathbf{X}$ is dense and if we assume that $\mathbf{W}$ is sparse, then using the class $\mathcal{W}_{1,1}$ is better than using $\mathcal{W}_{2,2}$. Such a scenario often happens when we have many irrelevant features and only are few features that can predict the target reasonably well. Concretely, suppose that $\mathbf{X} \in \{0,1\}^{k\times d}$ and that it typically has s_x non-zero values. Suppose also that there exists a matrix $\mathbf{W}$ that predicts the targets of the different tasks reasonably well and has s_w non-zero values. Then, the bound for $\mathcal{W}_{1,1}$ is order of $s_w\sqrt{\ln(dk)/n}$ while the bound for $\mathcal{W}_{2,2}$ is order of $\sqrt{s_w s_x/n}$. Thus, $\mathcal{W}_{1,1}$ will be better if $s_w < s_x/\ln(dk)$.

Now, consider the class $\mathcal{W}_{2,1}$. Let us further assume the following. The non-zero elements of $\mathbf{W}$ are grouped into s_g columns and are roughly distributed evenly over those columns; The non-zeros of $\mathbf{X}$ are roughly distributed evenly over the columns. Then, the bound for $\mathcal{W}_{2,1}$ is

1. The choices are $q_1 = \ln(kd)/(\ln(kd)-1)$, $q_2 = \ln(d)/(\ln(d)-1)$, $q_3 = \ln(k')/(\ln(k')-1)$ where $k' = \min\{k,d\}$.
2. We should alert the reader that the discussion here is based on *upper bounds* on the regret. While we have made the best effort to derive the tightest bounds we could, our understanding would have a firmer foundation if we also had matching *lower bounds*.

$s_g\sqrt{(s_w/s_g)\,(s_x/d)\,\ln(d)/n} = \sqrt{s_g\,s_w\,(s_x/d)\,\ln(d)/n}$. This bound will be better than the bound of $\mathcal{W}_{2,2}$ if $s_g \ln(d) < d$ and will be better than the bound of $\mathcal{W}_{1,1}$ if $s_g s_x/d < s_w$. We see that there are scenarios in which the group norm is better than the non-grouped norms and that the most adequate class depends on properties of the problem and our prior beliefs on a good predictor $\mathbf{W}$.

As to the bound for $\mathcal{W}_{S(1)}$, it is easy to verify that if the rows of $\mathbf{W}$ sits in a low dimensional subspace then the spectrum of $\mathbf{W}$ will be sparse. Similarly, the value of $\|\mathbf{X}\|_{S(\infty)}$ depends on the maximal singular value of $\mathbf{X}$, which is likely to be small if we assume that all the "energy" of $\mathbf{X}$ is spread over its entire spectrum. In such cases, $\mathcal{W}_{S(1)}$ can be the best choice. This is an example of a different type of prior knowledge on the problem.

4.2 Batch Multi-task Learning

In the batch setting, we see a data set $\mathcal{T} = ((\mathbf{X}_1, \mathbf{y}_1), \ldots, (\mathbf{X}_n, \mathbf{y}_n))$ consisting of i.i.d. samples drawn from a distribution $\mathcal{D}$ over $\mathcal{X} \times \mathcal{Y}$. In the k-task setting, $\mathcal{X} \subseteq \mathbb{R}^{k\times d}$. Analogous to the single task case, we define the risk and empirical risk of a multi-task predictor $\mathbf{W} \in \mathbb{R}^{k\times d}$ as:

$$\widehat{L}(\mathbf{W}) := \frac{1}{n}\sum_{i=1}^{n}\sum_{j=1}^{k} l^j\left(\left\langle \mathbf{w}^j, \mathbf{X}_i^j\right\rangle, y_i^j\right),$$

$$L(\mathbf{W}) := \mathbb{E}_{(\mathbf{X},\mathbf{y})\sim\mathcal{D}}\left[\sum_{j=1}^{k} l^j\left(\left\langle \mathbf{w}^j, \mathbf{X}^j\right\rangle, y^j\right)\right].$$

Let $\mathcal{W}$ be some class of matrices, and define the empirical risk minimizer,

$$\widehat{\mathbf{W}} := \operatorname{argmin}_{\mathbf{W}\in\mathcal{W}} \widehat{L}(\widehat{\mathbf{W}}).$$

To obtain excess risk bounds for $\widehat{\mathbf{W}}$, we need to consider the k-task Rademacher complexity

$$\mathcal{R}_{\mathcal{T}}^k(\mathcal{W}) := \mathbb{E}\left[\sup_{\mathbf{W}\in\mathcal{W}} \frac{1}{n}\sum_{i=1}^{n}\sum_{j=1}^{k} \varepsilon_i^j \left\langle \mathbf{w}^j, \mathbf{X}_i^j\right\rangle\right],$$

because, assuming each l^j is ρ-Lipschitz, we have the bound

$$\mathbb{E}\left[L(\widehat{\mathbf{W}}) - \min_{\mathbf{W}\in\mathcal{W}} L(\mathbf{W})\right] \le \rho\mathbb{E}\left[\mathcal{R}_{\mathcal{T}}^k(\mathcal{W})\right].$$

This bound follows easily from standard results, such as the Lipschitz contraction inequality (Bartlett and Mendelson, 2002), and Theorem 8 of Maurer (2006). We can use matrix strong convexity to give the following k-task Rademacher bound.

Theorem 21 *(Multi-task Generalization) Suppose $F(\mathbf{W}) \le f_{\max}$ for all $\mathbf{W} \in \mathcal{W}$ for a function F that is β-strongly convex w.r.t. some (matrix) norm $\|\cdot\|$. If the norm $\|\cdot\|_\star$ is invariant under sign changes of the rows of its argument matrix then, for any data set $\mathcal{T}$, we have, $\mathcal{R}_{\mathcal{T}}^k(\mathcal{W}) \le X\sqrt{\frac{2f_{\max}}{\beta n}}$, where X is an upper bound on $\|\mathbf{X}_i\|_\star$.*

Proof We can rewrite $\mathcal{R}^k_{\mathcal{I}}(\mathcal{W})$ as

$$\mathbb{E}\left[\sup_{\mathbf{W}\in\mathcal{W}}\frac{1}{n}\sum_{i=1}^{n}\sum_{j=1}^{k}\varepsilon_i^j\left\langle\mathbf{w}^j,\mathbf{X}_i^j\right\rangle\right]=\mathbb{E}\left[\sup_{\mathbf{W}\in\mathcal{W}}\frac{1}{n}\sum_{j=1}^{k}\left\langle\mathbf{w}^j,\sum_{i=1}^{n}\varepsilon_i^j\mathbf{X}_i^j\right\rangle\right]$$
$$=\mathbb{E}\left[\sup_{\mathbf{W}\in\mathcal{W}}\frac{1}{n}\left\langle\mathbf{W},\sum_{i=1}^{n}\tilde{\mathbf{X}}_i\right\rangle\right],$$

where $\tilde{\mathbf{X}}_i\in\mathbb{R}^{k\times d}$ is defined by $\tilde{\mathbf{X}}_i^j=\varepsilon_i^j\mathbf{X}_i^j$ and we have switched to a matrix inner product in the last line. By the assumption on the dual norm $\|\cdot\|_\star$, $\|\tilde{\mathbf{X}}_i\|_\star=\|\mathbf{X}_i\|_\star\le X$. Now using Corollary 4 and proceeding as in the proof of Theorem 7, we get, for any $\lambda>0$, $\mathcal{R}^k_{\mathcal{I}}(\mathcal{W})\le\left(\frac{f_{\max}}{\lambda n}+\frac{\lambda X^2}{2\beta}\right)$. Optimizing over λ proves the theorem. ■

Note that both group (r,p)-norms and Schatten-p norms satisfy the invariance under row flips mentioned in the theorem above. Thus, we get the following corollary.

Corollary 22 *Let $\mathcal{W}_{1,1},\mathcal{W}_{2,2},\mathcal{W}_{2,1},\mathcal{W}_{S(1)}$ be the classes defined in Section 3 and let $X_{\infty,\infty}$, $X_{2,2}$, $X_{2,\infty}$, $X_{S(\infty)}$ be the radii of $\mathcal{X}$ w.r.t. the corresponding dual norms. Then, the (expected) excess multi-task risk of the empirical multi-task risk minimizer $\widehat{\mathbf{W}}$ satisfies the same bounds as given in Table 1.*

5. Multi-class Learning

In this section we consider multi-class categorization problems. We focus on the online learning model. On round t, the online algorithm receives an instance $x_t\in\mathbb{R}^d$ and is required to predict its label as a number in $\{1,\ldots,k\}$. Following the construction of Crammer and Singer (2000), the prediction is based on a matrix $\mathbf{W}_t\in\mathbb{R}^{k\times d}$ and is defined as the index of the maximal element of the vector $\mathbf{W}_t x_t$. We use the hinge-loss function adapted to the multi-class setting. That is,

$$l_t(\mathbf{W}_t)\;=\;\max_r(1_{[r\neq y_t]}-(\langle\mathbf{w}_t^{y_t},x_t\rangle-\langle\mathbf{w}_t^r,x_t\rangle))=\max_r(1_{[r\neq y_t]}-(\langle\mathbf{W},\mathbf{X}_t^{r,y_t}\rangle)),$$

where $\mathbf{X}_t^{r,y_t}$ is a matrix with x_t on the y'th row, $-x_t$ on the r'th row, and zeros in all other elements. It is easy to verify that $l_t(\mathbf{W}_t)$ upper bounds the zero-one loss, that is, if the prediction of $\mathbf{W}_t$ is r then $l_t(\mathbf{W}_t)\ge 1_{[r\neq y_t]}$.

A sub-gradient of $l_t(\mathbf{W}_t)$ is either a matrix of the form $-\mathbf{X}_t^{r,y_t}$ or the all zeros matrix. Note that each column of $\mathbf{X}_t^{r,y_t}$ is very sparse (contains only two elements). Therefore,

$$\|\mathbf{X}_t^{r,y_t}\|_{\infty,\infty}=\|x_t\|_\infty,\qquad\|\mathbf{X}_t^{r,y_t}\|_{2,2}=\sqrt{2}\,\|x_t\|_2,$$
$$\|\mathbf{X}_t^{r,y_t}\|_{2,\infty}=\sqrt{2}\,\|x_t\|_\infty,\qquad\|\mathbf{X}_t^{r,y_t}\|_{S(\infty)}=\sqrt{2}\,\|x_t\|_2.$$

Based on this fact, we can easily obtain the following result.

Corollary 23 *Let $\mathcal{W}_{1,1},\mathcal{W}_{2,2},\mathcal{W}_{2,1},\mathcal{W}_{S(1)}$ be the classes defined in Section 3 and let $X_2=\max_t\|x_t\|_2$ and $X_\infty=\max_t\|x_t\|_\infty$. Then, there exist online multi-class learning algorithms with regret bounds given by the following table.*

class	$\mathcal{W}_{1,1}$	$\mathcal{W}_{2,2}$	$\mathcal{W}_{2,1}$	$\mathcal{W}_{S(1)}$
bound	$W_{1,1} X_\infty \sqrt{\frac{\ln(kd)}{n}}$	$W_{2,2} X_2 \sqrt{\frac{1}{n}}$	$W_{2,1} X_\infty \sqrt{\frac{\ln(d)}{n}}$	$W_{S(1)} X_2 \sqrt{\frac{\ln(\min\{d,k\})}{n}}$

Let us now discuss the implications of this bound. First, if $X_2 \approx X_\infty$, which will happen if instance vectors are sparse, then $\mathcal{W}_{1,1}$ and $\mathcal{W}_{2,1}$ will be inferior to $\mathcal{W}_{2,2}$. In such a case, using $\mathcal{W}_{S(1)}$ can be competitive if $\mathbf{W}$ sits in a low dimensional space so that $\|\mathbf{W}\|_{S(1)} \approx \|\mathbf{W}\|_{2,2}$. Using $\mathcal{W}_{S(1)}$ in such a case was previously suggested by Amit et al. (2007), who observed that empirically, the class $\mathcal{W}_{S(1)}$ performs better than $\mathcal{W}_{2,2}$ when there is a shared structure between classes. They only gave empirical results and our analysis given in Corollary 23 provides a first rigorous regret bound for their algorithm. Unfortunately, our bound for $\mathcal{W}_{S(1)}$ is not better than the bound for $\mathcal{W}_{2,2}$ even in the low rank case. Explaining the empirical success of the trace-norm based multi-class algorithm thus remains a challenging open question.

Second, if X_2 is much larger than X_∞, and if columns of $\mathbf{W}$ share common sparsity pattern, then $\mathcal{W}_{2,1}$ can be factor of $\sqrt{k}$ better than $\mathcal{W}_{1,1}$ and factor of $\sqrt{d}$ better than $\mathcal{W}_{2,2}$. To demonstrate this, let us assume that each vector x_t is in $\{\pm 1\}^d$ and it represents experts advice of d experts. Therefore, $X_2 = \sqrt{d} X_\infty$. Next, assume that a combination of the advice of $s \ll d$ experts predicts very well the correct label (e.g., the label is represented by the binary number obtained from the advice of $s = \log(k)$ experts). In that case, W will be a matrix such that all of its columns will be 0 except s columns which will take values in $\{\pm 1\}$. The bounds for $\mathcal{W}_{1,1}, \mathcal{W}_{2,2}$, and $\mathcal{W}_{2,1}$ in that case become (proportional to) $ks\sqrt{\ln(kd)}$, $\sqrt{ksd}$, and $s\sqrt{k\ln(d)}$ respectively. That is, $\mathcal{W}_{2,1}$ is a factor of $\sqrt{k}$ better than $\mathcal{W}_{1,1}$ and a factor of $\sqrt{d/s}$ better than $\mathcal{W}_{2,2}$ (ignoring logarithmic terms). The class $\mathcal{W}_{S(1)}$ will also have a dependent on $\sqrt{d}$ in such a case and thus it will be much worse than $\mathcal{W}_{2,2}$ when d is large.

For concreteness, we now use our result for deriving a group multi-class perceptron algorithm. To the best of our knowledge, this algorithm is new, and based on the discussion above, it should outperform both the multi-class perceptron of Crammer and Singer (2000) as well as the vanilla application of the p-norm perceptron framework of Gentile (2003) and Grove et al. (2001) for multi-class categorization.

The algorithm is a specification of the general online mirror descent procedure (Algorithm 1) with $f(\mathbf{W}) = \frac{1}{2}\|\mathbf{W}\|^2_{2,r}$, $r = \ln(d)/(\ln(d)-1)$, and with a conservative update (i.e., we ignore rounds on which no prediction mistake has been made). Recall that the Fenchel dual function is $f^\star(\mathbf{V}) = \frac{1}{2}\|\mathbf{V}\|^2_{2,p}$ where $p = (1-1/r)^{-1} = \ln(d)$. The (i,j) element of the gradient of $f^\star$ is

$$(\nabla f^\star(\mathbf{V}))_{i,j} \;=\; \frac{\|\mathbf{V}^j\|_2^{p-2}}{\|\mathbf{V}\|_{2,p}^{p-2}} V_{i,j} \;. \tag{2}$$

To analyze the performance of Algorithm 3, let $I \subseteq [n]$ be the set of rounds on which the algorithm made a prediction mistake. Note that the above algorithm is equivalent (in terms of the number of mistakes) to an algorithm that performs the update $\mathbf{V}_{t+1} = \mathbf{V}_t + \eta \mathbf{U}_t$ for any η (see Gentile, 2003). Therefore, we can apply our general online regret bound (Corollary 17) on the sequence of examples in I we obtain that for any $\mathbf{W}$,

$$\sum_{t\in I} l_t(\mathbf{W}_t) - \sum_{t\in I} l_t(\mathbf{W}) \;\le\; X_\infty \|\mathbf{W}\|_{2,1} \sqrt{6\ln(d)\,|I|} \;.$$

Algorithm 3 Group Multi-class Perceptron

$p = \log d$
$\mathbf{V}_1 = \mathbf{0} \in \mathbb{R}^{k \times d}$
for $t = 1, \ldots, T$ **do**
  Set $\mathbf{W}_t = \nabla f^\star(\mathbf{V}_t)$ (as defined in (2))
  Receive $\mathbf{x}_t \in \mathbb{R}^d$
  $\hat{y}_t = \arg\max_{r \in [k]} \ (\mathbf{W}_t \mathbf{x}_t)_r$
  Predict $\hat{y}_t$ and receive true label y_t
  $\mathbf{U}_t \in \mathbb{R}^{k \times d}$ is the matrix with $\mathbf{x}_t$ in the $\hat{y}_t$ row and $-\mathbf{x}_t$ in the y_t row
  Update: $\mathbf{V}_{t+1} = \mathbf{V}_t - \mathbf{U}_t$
end for

Recall that $l_t(\mathbf{W}_t)$ upper bounds the zero-one error and therefore the above implies that

$$|I| - \sum_{t \in I} l_t(\mathbf{W}) \ \leq \ X_\infty \|\mathbf{W}\|_{2,1} \sqrt{6 \ln(d) \, |I|} \ .$$

Solving for $|I|$ gives us the following result.

Corollary 24 *The number of mistakes Algorithm 3 will make on any sequence of examples for which $\|x_t\|_\infty \leq X_\infty$ is upper bounded by*

$$\min_{\mathbf{W}} \ \sum_t l_t(\mathbf{W}) + X_\infty \|\mathbf{W}\|_{2,1} \sqrt{3 \ln(d) \sum_t l_t(\mathbf{W})} + 3 X_\infty^2 \|\mathbf{W}\|_{2,1}^2 \ln(d) \ .$$

6. Kernel Learning

We briefly review the kernel learning setting first explored in Lanckriet et al. (2004). Let $\mathcal{X}$ be an input space and let $\mathcal{T} = (\mathbf{x}_1, \ldots, \mathbf{x}_n) \in \mathcal{X}^n$ be the training data set. Kernel algorithms work with the space of linear functions,

$$\left\{ \mathbf{x} \mapsto \sum_{i=1}^n \alpha_i K(\mathbf{x}_i, \mathbf{x}) \ : \ \alpha_i \in \mathbb{R} \right\} \ .$$

In kernel learning, we consider a kernel *family* $\mathcal{K}$ and consider the class,

$$\left\{ \mathbf{x} \mapsto \sum_{i=1}^n \alpha_i K(\mathbf{x}_i, \mathbf{x}) \ : \ K \in \mathcal{K}, \ \alpha_i \in \mathbb{R} \right\} \ .$$

In particular, we can choose a finite set $\{K_1, \ldots, K_k\}$ of base kernels and consider the convex combinations, $\mathcal{K}_c^+ = \left\{ \sum_{j=1}^k \mu_j K_j \ : \ \mu_j \geq 0, \ \sum_{j=1}^k \mu_j = 1 \right\}$. This is the unconstrained function class. In applications, one constrains the function class in some way. The class considered by Lanckriet et al. (2004) is

$$\mathcal{F}_{\mathcal{K}_c^+} = \left\{ \mathbf{x} \mapsto \sum_{i=1}^n \alpha_i K(\mathbf{x}_i, \cdot) \ : \ K = \sum_{j=1}^k \mu_j K_j, \ \mu_j \geq 0, \ \sum_{j=1}^k \mu_j = 1, \ \alpha^\top K(\mathcal{T}) \alpha \leq 1/\gamma^2 \right\} \qquad (3)$$

where $\gamma > 0$ is a margin parameter and $K(\mathcal{T})_{i,j} = K(\mathbf{x}_i, \mathbf{x}_j)$ is the Gram matrix of K on the data set $\mathcal{T}$.

Theorem 25 *(Kernel learning) Consider the class $\mathcal{F}_{\mathcal{K}^+}$ defined in* (3). *Let $K_j(\mathbf{x},\mathbf{x}) \leq B$ for $1 \leq j \leq k$ and $\mathbf{x} \in \mathcal{X}$. Then, $\mathcal{R}_{\mathcal{T}}(\mathcal{F}_{\mathcal{K}^+}) \leq e\sqrt{\frac{B\log k}{\gamma^2 n}}$.*

The proof follows directly from the equivalence between kernel learning and group Lasso (Bach, 2008), and then applying our bound on the class $\mathcal{W}_{2,1}$. For completeness, we give a rigorous proof in the appendix.

Note that the dependence on the number of base kernels, k, is rather mild (only logarithmic)—implying that we can learn a kernel as a (convex) combination of a rather large number of base kernels. Also, let us discuss how the above improves upon the prior bounds provided by Lanckriet et al. (2004) and Srebro and Ben-David (2006) (neither of which had logarithmic k dependence). The former proves a bound of $O\left(\sqrt{\frac{Bk}{\gamma^2 n}}\right)$ which is quite inferior to our bound. We cannot compare our bound directly to the bound of Srebro and Ben-David (2006) as they do not work with Rademacher complexities. However, if one compares the resulting generalization error bounds, then their bound is $O\left(\sqrt{\frac{k\log\frac{n^3 B}{\gamma^2 k}+\frac{B}{\gamma^2}\log\frac{\gamma n}{\sqrt{B}}\log\frac{nB}{\gamma^2}}{n}}\right)$ and ours is $O\left(\sqrt{\frac{B\log k}{\gamma^2 n}}\right)$. If $k \geq n$, their bound is vacuous (while ours is still meaningful). If $k \leq n$, our bound is better.

Finally, we note that recently Ying and Campbell (2009) devoted a dedicated effort to derive a result similar to Theorem 25 using a Rademacher chaos process of order two over candidate kernels. In contrast to their proof, our result seamlessly follows from the general framework of deriving bounds using the strong-convexity/strong-smoothness duality.

Acknowledgments

We thank Andreas Argyriou, Shmuel Friedland and Karthik Sridharan for helpful discussions. We are also thankful to the three anonymous reviewers for their constructive criticism and helpful suggestions.

Appendix A. Convex Analysis and Matrix Computation

We briefly recall some key definitions from convex analysis that are useful throughout the paper (for details, see any of the several excellent references on the subject, for example, Borwein and Lewis, 2006; Rockafellar, 1970).

A.1 Convex Analysis

We consider convex functions $f : \mathcal{X} \to \mathbb{R} \cup \{\infty\}$, where $\mathcal{X}$ is a Euclidean vector space equipped with an inner product $\langle \cdot, \cdot \rangle$. We denote $\mathbb{R}^* = \mathbb{R} \cup \{\infty\}$. Recall that the subdifferential of f at $x \in \mathcal{X}$, denoted by $\partial f(x)$, is defined as $\partial f(x) := \{y \in \mathcal{X} \ : \ \forall z,\ f(x+z) \geq f(x) + \langle y, z\rangle\}$. The Fenchel conjugate $f^\star : \mathcal{X} \to \mathbb{R}^*$ is defined as $f^\star(y) := \sup_{x\in\mathcal{X}} \langle x, y\rangle - f(x)$.

We also deal with a variety of norms in this paper. Recall that given a norm $\|\cdot\|$ on $\mathcal{X}$, its dual norm is defined as $\|y\|_\star := \sup\{\langle x, y\rangle \ : \ \|x\| \leq 1\}$. An important property of the dual norm is that the Fenchel conjugate of the function $\frac{1}{2}\|x\|^2$ is $\frac{1}{2}\|y\|_\star^2$.

The definition of Fenchel conjugate implies that for any x, y, $f(x) + f^\star(y) \geq \langle x, y \rangle$, which is known as the Fenchel-Young inequality. An equivalent and useful definition of the subdifferential can be given in terms of the Fenchel conjugate: $\partial f(x) = \{y \in \mathcal{X} \,:\, f(x) + f^*(y) = \langle x, y \rangle\}$.

A.2 Convex Analysis of Matrix Functions

We consider the vector space $\mathcal{X} = \mathbb{R}^{m \times n}$ of real matrices of size $m \times n$ and the vector space $\mathcal{X} = \mathbb{S}^n$ of symmetric matrices of size $n \times n$, both equipped with the inner product, $\langle \mathbf{X}, \mathbf{Y} \rangle := \mathrm{Tr}(\mathbf{X}^\top \mathbf{Y})$. Recall that any matrix $\mathbf{X} \in \mathbb{R}^{m \times n}$ can be decomposed as $\mathbf{X} = \mathbf{U}\mathrm{Diag}(\sigma(\mathbf{X}))\mathbf{V}$ where $\sigma(\mathbf{X})$ denotes the vector $(\sigma_1, \sigma_2, \ldots \sigma_l)$ ($l = \min\{m, n\}$), where $\sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_l \geq 0$ are the singular values of $\mathbf{X}$ arranged in non-increasing order, and $\mathbf{U} \in \mathbb{R}^{m \times m}, \mathbf{V} \in \mathbb{R}^{n \times n}$ are orthogonal matrices. Also, any matrix $\mathbf{X} \in \mathbb{S}^n$ can be decomposed as, $X = \mathbf{U}\mathrm{Diag}(\lambda(\mathbf{X}))\mathbf{U}^\top$ where $\lambda(\mathbf{X}) = (\lambda_1, \lambda_2, \ldots \lambda_n)$, where $\lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_n$ are the eigenvalues of $\mathbf{X}$ arranged in non-increasing order, and $\mathbf{U}$ is an orthogonal matrix. Two important results relate matrix inner products to inner products between singular (and eigen-) values

Theorem 26 (von Neumann) *Any two matrices $X, Y \in \mathbb{R}^{m \times n}$ satisfy the inequality*

$$\langle \mathbf{X}, \mathbf{Y} \rangle \leq \langle \sigma(\mathbf{X}), \sigma(\mathbf{Y}) \rangle \ .$$

Equality holds above, if and only if, there exist orthogonal $\mathbf{U}, \mathbf{V}$ such that

$$\mathbf{X} = \mathbf{U}\mathrm{Diag}(\sigma(\mathbf{X}))\mathbf{V}, \qquad \mathbf{Y} = \mathbf{U}\mathrm{Diag}(\sigma(\mathbf{Y}))\mathbf{V} \ .$$

Theorem 27 (Fan) *Any two matrices $\mathbf{X}, \mathbf{Y} \in \mathbb{S}^n$ satisfy the inequality*

$$\langle \mathbf{X}, \mathbf{Y} \rangle \leq \langle \lambda(\mathbf{X}), \lambda(\mathbf{Y}) \rangle \ .$$

Equality holds above, if and only if, there exists orthogonal $\mathbf{U}$ such that

$$\mathbf{X} = \mathbf{U}\mathrm{Diag}(\lambda(\mathbf{X}))\mathbf{U}^\top, \qquad \mathbf{Y} = \mathbf{U}\mathrm{Diag}(\lambda(\mathbf{Y}))\mathbf{U}^\top \ .$$

We say that a function $g : \mathbb{R}^n \to \mathbb{R}^*$ is symmetric if $g(x)$ is invariant under arbitrary permutations of the components of x. We say g is absolutely symmetric if $g(x)$ is invariant under arbitrary permutations and sign changes of the components of x.

Given a function $f : \mathbb{R}^l \to \mathbb{R}^*$, we can define a function $f \circ \sigma : \mathbb{R}^{m \times n} \to \mathbb{R}^*$ as,

$$(f \circ \sigma)(\mathbf{X}) := f(\sigma(\mathbf{X})) \ .$$

Similarly, given a function $g : \mathbb{R}^n \to \mathbb{R}^*$, we can define a function $g \circ \lambda : \mathbb{S}^n \to \mathbb{R}^*$ as,

$$(g \circ \lambda)(\mathbf{X}) := g(\lambda(\mathbf{X})) \ .$$

This allows us to define functions over matrices starting from functions over vectors. Note that when we use $f \circ \sigma$ we are assuming that $\mathcal{X} = \mathbb{R}^{m \times n}$ and for $g \circ \lambda$ we have $\mathcal{X} = \mathbb{S}^n$. The following result allows us to immediately compute the conjugate of $f \circ \sigma$ and $g \circ \lambda$ in terms of the conjugates of f and g respectively.

Theorem 28 *(Lewis, 1995) Let $f : \mathbb{R}^l \to \mathbb{R}^*$ be an absolutely symmetric function. Then,*

$$(f \circ \sigma)^\star = f^\star \circ \sigma \, .$$

Let $g : \mathbb{R}^n \to \mathbb{R}^*$ be a symmetric function. Then,

$$(g \circ \lambda)^\star = g^\star \circ \lambda \, .$$

Proof Lewis (1995) proves this for singular values. For the eigenvalue case, the proof is entirely analogous to that in Lewis (1995), except that Fan's inequality is used instead of von Neumann's inequality. ■

Using this general result, we are able to define certain matrix norms.

Corollary 29 *(Matrix norms) Let $f : \mathbb{R}^l \to \mathbb{R}^*$ be absolutely symmetric. Then if $f = \|\cdot\|$ is a norm on $\mathbb{R}^l$ then $f \circ \sigma = \|\sigma(\cdot)\|$ is a norm on $\mathbb{R}^{m \times n}$. Further, the dual of this norm is $\|\sigma(\cdot)\|_\star$.*

Let $g : \mathbb{R}^n \to \mathbb{R}^*$ be symmetric. Then if $g = \|\cdot\|$ is a norm on $\mathbb{R}^n$ then $g \circ \lambda = \|\lambda(\cdot)\|$ is a norm on $\mathbb{S}^n$. Further, the dual of this norm is $\|\lambda(\cdot)\|_\star$.

Another nice result allows us to compute subdifferentials of $f \circ \sigma$ and $g \circ \lambda$ (note that elements in the subdifferential of $f \circ \sigma$ and $g \circ \lambda$ are matrices) from the subdifferentials of f and g respectively.

Theorem 30 *(Lewis, 1995) Let $f : \mathbb{R}^l \to \mathbb{R}^*$ be absolutely symmetric and $\mathbf{X} \in \mathbb{R}^{m \times n}$. Then,*

$$\partial(f \circ \sigma)(\mathbf{X}) = \{\mathbf{U}\mathrm{Diag}(\mu)\mathbf{V}^\top \, : \, \mu \in \partial f(\sigma(\mathbf{X}))\mathbf{U}, \mathbf{V} \text{ orthogonal}, \, \mathbf{X} = \mathbf{U}\mathrm{Diag}(\sigma(\mathbf{X}))\mathbf{V}^\top\} \, .$$

Let $g : \mathbb{R}^n \to \mathbb{R}^*$ be symmetric and $\mathbf{X} \in \mathbb{S}^n$. Then,

$$\partial(g \circ \lambda)(\mathbf{X}) = \{\mathbf{U}\mathrm{Diag}(\mu)\mathbf{U}^\top \, : \, \mu \in \partial g(\lambda(\mathbf{X}))\mathbf{U} \text{ orthogonal}, \, \mathbf{X} = \mathbf{U}\mathrm{Diag}(\lambda(X))\mathbf{U}^\top\} \, .$$

Proof Again, Lewis (1995) proves the case for singular values. For the eigenvalue case, again, the proof is identical to that in Lewis (1995), except that Fan's inequality is used instead of von Neumann's inequality. ■

Appendix B. Technical Proofs

We now provide the proofs omitted from the main body of the paper.

B.1 Proof of Theorem 3

First, (Shalev-Shwartz, 2007, Lemma 15) yields one half of the claim (f strongly convex $\Rightarrow f^\star$ strongly smooth). It is left to prove that f is strongly convex assuming that $f^\star$ is strongly smooth. For simplicity assume that $\beta = 1$. Denote $g(y) = f^\star(x+y) - (f^\star(x) + \langle \nabla f^\star(x), y \rangle)$. By the smoothness assumption, $g(y) \leq \frac{1}{2}\|y\|_\star^2$. This implies that $g^\star(a) \geq \frac{1}{2}\|a\|^2$ because of (Shalev-Shwartz and

Singer, 2008, Lemma 19) and that the conjugate of half squared norm is half squared of the dual norm. Using the definition of g we have

$$\begin{aligned} g^\star(a) &= \sup_y \langle y, a\rangle - g(y) \\ &= \sup_y \langle y, a\rangle - (f^\star(x+y) - (f^\star(x) + \langle \nabla f^\star(x), y\rangle)) \\ &= \sup_y \langle y, a+\nabla f^\star(x)\rangle - f^\star(x+y) + f^\star(x) \\ &= \sup_z \langle z-x, a+\nabla f^\star(x)\rangle - f^\star(z) + f^\star(x) \\ &= f(a+\nabla f^\star(x)) + f^\star(x) - \langle x, a+\nabla f^\star(x)\rangle \end{aligned}$$

where we have used that $f^{\star\star} = f$, in the last step. Denote $u = \nabla f^\star(x)$. From the equality in Fenchel-Young (e.g., Shalev-Shwartz and Singer, 2008, Lemma 17) we obtain that $\langle x, u\rangle = f^\star(x) + f(u)$ and thus

$$g^\star(a) = f(a+u) - f(u) - \langle x, a\rangle \ .$$

Combining with $g^\star(a) \geq \frac{1}{2}\|a\|^2$, we have

$$f(a+u) - f(u) - \langle x, a\rangle \ \geq \ \frac{1}{2}\|a\|^2 \ , \tag{4}$$

which holds for all a, x, with $u = \nabla f^\star(x)$.

Now let us prove that for any point u' in the relative interior of the domain of f that if $x \in \partial f(u')$ then $u' = \nabla f^\star(x)$. Let $u := \nabla f^\star(x)$ and we must show that $u' = u$. By Fenchel-Young, we have that $\langle x, u'\rangle = f^\star(x) + f(u')$, and, again by Fenchel-Young (and $f^{\star\star} = f$), we have $\langle x, u\rangle = f^\star(x) + f(u)$. We can now apply (4) to obtain:

$$\begin{aligned} 0 &= \langle x, u\rangle - f(u) - \left(\langle x, u'\rangle - f(u')\right), \\ &= f(u') - f(u) - \langle x, u'-u\rangle \ \geq \ \frac{1}{2}\|u'-u\|^2 \ , \end{aligned}$$

which implies that $u' = \nabla f^\star(x)$.

Next, let u_1, u_2 be two points in the relative interior of the domain of f, let $\alpha \in (0,1)$, and let $u = \alpha u_1 + (1-\alpha)u_2$. Let $x \in \partial f(u)$ (which is non-empty[3]). We have that $u = \nabla f^\star(x)$, by the previous argument. Now we are able to apply (4) twice, once with $a = u_1 - u$ and once with $a = u_2 - u$ (and both with x) to obtain

$$\begin{aligned} f(u_1) - f(u) - \langle x, u_1 - u\rangle &\geq \frac{1}{2}\|u_1 - u\|^2 \\ f(u_2) - f(u) - \langle x, u_2 - u\rangle &\geq \frac{1}{2}\|u_2 - u\|^2 \ . \end{aligned}$$

Finally, summing up the above two equations with coefficients α and $1-\alpha$ we obtain that f is strongly convex.

3. The set $\partial f(u)$ is not empty for all u in the relative interior of the domain of f. See the relative max formula in (Borwein and Lewis, 2006, page 42) or (Rockafellar, 1970, page 253). If u is not in the interior of f, then $\partial f(u)$ is empty. But, a function is defined to be essentially strictly convex if it is strictly convex on any subset of $\{u : \partial f(u) \neq \emptyset\}$. The last set is called the domain of ∂f and it contains the relative interior of the domain of f, so we are fine here.

B.2 Proof of Theorem 13

Note that an equivalent definition of σ-smoothness of a function f w.r.t. a norm $\|\cdot\|$ is that, for all x, y and $\alpha \in [0,1]$, we have

$$f(\alpha x + (1-\alpha)y) \geq \alpha f(x) + (1-\alpha) f(y) - \frac{1}{2}\sigma\alpha(1-\alpha)\|x-y\|^2 .$$

Let $\mathbf{X}, \mathbf{Y} \in \mathbb{R}^{m\times n}$ be arbitrary matrices with columns $\mathbf{X}^i$ and $\mathbf{Y}^i$ respectively. We need to prove

$$\|(1-\alpha)\mathbf{X} + \alpha\mathbf{Y}\|^2_{\Psi,\Phi} \geq \alpha\|\mathbf{X}\|^2_{\Psi,\Phi} + (1-\alpha)\|\mathbf{Y}\|^2_{\Psi,\Phi} - \frac{1}{2}(\sigma_1+\sigma_2)\alpha(1-\alpha)\|\mathbf{X}-\mathbf{Y}\|^2_{\Psi,\Phi} . \tag{5}$$

Using smoothness of Ψ and that Φ is a Q-norm, we have,

$$\begin{aligned}
\|(1-\alpha)\mathbf{X} + \alpha\mathbf{Y}\|^2_{\Psi,\Phi} &= (\Phi^2 \circ \sqrt{})(\ldots, \Psi^2(\alpha\mathbf{X}^i + (1-\alpha)\mathbf{Y}^i), \ldots) \\
&\geq (\Phi^2 \circ \sqrt{})(\ldots, \alpha\Psi^2(\mathbf{X}^i) + (1-\alpha)\Psi^2(\mathbf{Y}^i) \\
&\quad - \frac{1}{2}\sigma_1\alpha(1-\alpha)\Psi^2(\mathbf{X}^i - \mathbf{Y}^i), \ldots) \\
&\geq (\Phi^2 \circ \sqrt{})(\ldots, \alpha\Psi^2(\mathbf{X}^i) + (1-\alpha)\Psi^2(\mathbf{Y}^i), \ldots) \\
&\quad - \frac{1}{2}\sigma_1\alpha(1-\alpha)(\Phi^2 \circ \sqrt{})(\ldots, \Psi^2(\mathbf{X}^i - \mathbf{Y}^i), \ldots) \\
&= \Phi^2(\ldots, \sqrt{\alpha\Psi^2(\mathbf{X}^i) + (1-\alpha)\Psi^2(\mathbf{Y}^i)}, \ldots) \\
&\quad - \frac{1}{2}\sigma_1\alpha(1-\alpha)\|\mathbf{X}-\mathbf{Y}\|^2_{\Psi,\Phi} .
\end{aligned} \tag{6}$$

Now, we use that, for any $x, y \geq 0$ and $\alpha \in [0,1]$, we have $\sqrt{\alpha x^2 + (1-\alpha)y^2} \geq \alpha x + (1-\alpha)y$. Thus, we have

$$\begin{aligned}
&\Phi^2(\ldots, \sqrt{\alpha\Psi^2(\mathbf{X}^i) + (1-\alpha)\Psi^2(\mathbf{Y}^i)}, \ldots) \\
&\geq \Phi^2(\ldots, \alpha\Psi(\mathbf{X}^i) + (1-\alpha)\Psi(\mathbf{Y}^i), \ldots) \\
&\geq \alpha\Phi^2(\ldots, \Psi(\mathbf{X}^i), \ldots) + (1-\alpha)\Phi^2(\ldots, \Psi(\mathbf{Y}^i), \ldots) \\
&\quad - \frac{1}{2}\sigma_2\alpha(1-\alpha)\Phi^2(\ldots, \Psi(\mathbf{X}^i) - \Psi(\mathbf{Y}^i), \ldots) \\
&\geq \alpha\|\mathbf{X}\|^2_{\Psi,\Phi} + (1-\alpha)\|\mathbf{Y}\|^2_{\Psi,\Phi} - \frac{1}{2}\sigma_2\alpha(1-\alpha)\Phi^2(\ldots, \Psi(\mathbf{X}^i - \mathbf{Y}^i), \ldots) \\
&= \alpha\|\mathbf{X}\|^2_{\Psi,\Phi} + (1-\alpha)\|\mathbf{Y}\|^2_{\Psi,\Phi} - \frac{1}{2}\sigma_2\alpha(1-\alpha)\|\mathbf{X}-\mathbf{Y}\|^2_{\Psi,\Phi} .
\end{aligned}$$

Plugging this into (6) proves (5).

B.3 Proof of Theorem 25

Let $\mathcal{H}_j$ be the RKHS of K_j, $\mathcal{H}_j = \left\{\sum_{i=1}^l \alpha_i K_j(\tilde{\mathbf{x}}_i, \cdot) \,:\, l > 0,\ \tilde{\mathbf{x}}_i \in \mathcal{X},\ \alpha \in \mathbb{R}^l\right\}$ equipped with the inner product

$$\left\langle \sum_{i=1}^l \alpha_i K_j(\tilde{\mathbf{x}}_i, \cdot), \sum_{j=1}^m \alpha'_i K_j(\tilde{\mathbf{x}}'_j, \cdot) \right\rangle_{\mathcal{H}_j} = \sum_{i,j} \alpha_i \alpha'_j K_j(\tilde{\mathbf{x}}_i, \tilde{\mathbf{x}}'_j).$$

Consider the space $\mathcal{H} = \mathcal{H}_1 \times \ldots \times \mathcal{H}_k$ equipped with the inner product

$$\langle \vec{u}, \vec{v} \rangle := \sum_{i=1}^{k} \langle u_i, v_i \rangle_{\mathcal{H}_i} \ .$$

For $\vec{w} \in \mathcal{H}$, let $\|\cdot\|_{2,1}$ be the norm defined by

$$\|\vec{w}\|_{2,1} = \sum_{i=1}^{k} \|w_i\|_{\mathcal{H}_i} \ .$$

It is easy to verify that $\mathcal{F}_{\mathcal{K}_c^+} \subseteq \mathcal{F}_r$ where

$$\mathcal{F}_r := \{\mathbf{x} \mapsto \left\langle \vec{w}, \vec{\phi}(\mathbf{x}) \right\rangle \ : \ \vec{w} \in \mathcal{H}, \ \|\vec{w}\|_{2,1} \leq 1/\gamma\} \ ,$$

and

$$\vec{\phi}(\mathbf{x}) = (K_1(\mathbf{x}, \cdot), \ldots, K_k(\mathbf{x}, \cdot)) \in \mathcal{H} \ .$$

Since $\|K_j(\mathbf{x}, \cdot)\|_{\mathcal{H}_j} \leq \sqrt{B}$, we also have $\|\vec{\phi}(\mathbf{x})\|_{2,s} \leq k^{1/s}\sqrt{B}$ for any $\mathbf{x} \in \mathcal{X}$. The claim now follows directly from the results we derived in Section 2.

References

A. Agarwal, A. Rakhlin, and P. Bartlett. Matrix regularization techniques for online multitask learning. Technical Report EECS-2008-138, EECS Department, University of California, Berkeley, 2008.

Y. Amit, M. Fink, N. Srebro, and S. Ullman. Uncovering shared structures in multiclass classification. In *Proceedings of the 24th International Conference on Machine Learning*, pages 17–24, 2007.

A. Argyriou, T. Evgeniou, and M. Pontil. Convex multi-task feature learning. *Machine Learning*, 73(3):243–272, 2008.

A. Argyriou, C. A. Micchelli, and M. Pontil. On spectral learning. *Journal of Machine Learning Research*, 11:935–953, 2010.

F. Bach. Consistency of the group Lasso and multiple kernel learning. *Journal of Machine Leaning Research*, 9:1179–1225, 2008.

K. Ball, E. A. Carlen, and E. H. Lieb. Sharp uniform convexity and smoothness inequalities for trace norms. *Inventiones Mathematicae*, 115:463–482, 1994.

P. L. Bartlett and S. Mendelson. Rademacher and Gaussian complexities: Risk bounds and structural results. *Journal of Machine Learning Research*, 3:463–482, 2002.

R. Bhatia. *Matrix Analysis*. Springer, 1997.

J. Borwein and A. Lewis. *Convex Analysis and Nonlinear Optimization*. Springer, 2006.

G. Cavallanti, N. Cesa-Bianchi, and C. Gentile. Linear algorithms for online multitask classification. In *Proceedings of the Twenty-first Annual Conference on Computational Learning Theory*, pages 251–262, 2008.

N. Cesa-Bianchi and G. Lugosi. *Prediction, Learning, and Games*. Cambridge University Press, 2006.

K. Crammer and Y. Singer. On the learnability and design of output codes for multiclass problems. In *Proceedings of the Thirteenth Annual Conference on Computational Learning Theory*, pages 35–46, 2000.

C. Gentile. The robustness of the p-norm algorithms. *Machine Learning*, 53(3):265–299, 2003.

A. J. Grove, N. Littlestone, and D. Schuurmans. General convergence results for linear discriminant updates. *Machine Learning*, 43(3):173–210, 2001.

A. Juditsky and A. Nemirovski. Large deviations of vector-valued martingales in 2-smooth normed spaces. *submitted to Annals of Probability*, 2008. preprint available at `arxiv.org/abs/0809.0813`.

S. M. Kakade, K. Sridharan, and A. Tewari. On the complexity of linear prediction: Risk bounds, margin bounds, and regularization. In *Advances in Neural Information Processing Systems 22*, pages 793–800, 2008.

J. Kivinen and M. Warmuth. Exponentiated gradient versus gradient descent for linear predictors. *Information and Computation*, 132(1):1–64, January 1997.

J. Kivinen and M. Warmuth. Relative loss bounds for multidimensional regression problems. *Machine Learning*, 45(3):301–329, July 2001.

G.R.G. Lanckriet, N. Cristianini, P.L. Bartlett, L. El Ghaoui, and M.I. Jordan. Learning the kernel matrix with semidefinite programming. *Journal of Machine Learning Research*, 5:27–72, 2004.

A. S. Lewis. The convex analysis of unitarily invariant matrix functions. *Journal of Convex Analysis*, 2(2):173–183, 1995.

N. Littlestone. Learning quickly when irrelevant attributes abound: A new linear-threshold algorithm. *Machine Learning*, 2:285–318, 1988.

K. Lounici, M. Pontil, A. B Tsybakov, and S. van de Geer. Taking advantage of sparsity in multi-task learning. In *Proceedings of the Twenty-second Annual Conference on Computational Learning Theory*, 2009.

A. Maurer. Bounds for linear multi-task learning. *Journal of Machine Learning Research*, 7:117–139, 2006.

R. Meir and T. Zhang. Generalization error bounds for Bayesian mixture algorithms. *Journal of Machine Learning Research*, 4:839–860, 2003.

A.Y. Ng. Feature selection, L_1 vs. L_2 regularization, and rotational invariance. In *Proceedings of the Twenty-First International Conference on Machine Learning*, pages 78–85, 2004.

G. Obozinski, B. Taskar, and M Jordan. Joint covariate selection and joint subspace selection for multiple classification problems. *Statistics and Computing*, 20:231–252, 2010.

I. Pinelis. Optimum bounds for the distributions of martingales in Banach spaces. *Annals of Probability*, 22(4):1679–1706, 1994.

G. Pisier. Martingales with values in uniformly convex spaces. *Israel Journal of Mathematics*, 20 (3–4):326–350, 1975.

R.T. Rockafellar. *Convex Analysis*. Princeton University Press, 1970.

S. Shalev-Shwartz. *Online Learning: Theory, Algorithms, and Applications*. PhD thesis, Hebrew University of Jerusalem, 2007.

S. Shalev-Shwartz and Y. Singer. Convex repeated games and Fenchel duality. In *Advances in Neural Information Processing Systems 20*, pages 1265–1272, 2006.

S. Shalev-Shwartz and Y. Singer. A primal-dual perspective of online learning algorithms. *Machine Learning Journal*, 69(2-3):115–142, 2007.

S. Shalev-Shwartz and Y. Singer. On the equivalence of weak learnability and linear separability: New relaxations and efficient boosting algorithms. In *Proceedings of the Twenty-first Annual Conference on Computational Learning Theory*, pages 311–322, 2008.

N. Srebro and S. Ben-David. Learning bounds for support vector machines with learned kernels. In *Proceedings of the Nineteenth Annual Conference on Computational Learning Theory*, pages 169–183, 2006.

N. Srebro, J. Rennie, and T. Jaakkola. Maximum margin matrix factorization. In *Advances in Neural Information Prcoessing Systems 17*, pages 1329–1336, 2005.

K. Sridharan and A. Tewari. Convex games in Banach spaces. In *Proceedings of the 23rd Annual Conference on Learning Theory*, pages 1–13. Omnipress, 2010.

M. Warmuth and D. Kuzmin. Online variance minimization. In *Proceedings of the Nineteenth Annual Conference on Computational Learning Theory*, pages 514–528, 2006.

X. Yang, S. Kim, and E. P. Xing. Heterogeneous multitask learning with joint sparsity constraints. In *Advances in Neural Information Processing Systems 23*, pages 2151–2159, 2009.

Y. Ying and C. Campbell. Generalization bounds for learning the kernel. In *Proceedings of the Twenty-second Annual Conference on Computational Learning Theory*, 2009.

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.

C. Zalinescu. *Convex Analysis in General Vector Spaces*. World Scientific Publishing Co. Inc., River Edge, NJ, 2002.

Journal of Machine Learning Research 13 (2012) 1891-1926 Submitted 12/10; Revised 12/11; Published 6/12

Confidence-Weighted Linear Classification for Text Categorization

Koby Crammer KOBY@EE.TECHNION.AC.IL
Department of Electrical Engineering
The Technion
Haifa 32000, Israel

Mark Dredze MDREDZE@CS.JHU.EDU
Human Language Technology Center of Excellence
Johns Hopkins University
Baltimore, MD 21211, USA

Fernando Pereira PEREIRA@GOOGLE.COM
Google, Inc.
1600 Amphiteatre Pkwy
Mountain View, CA 94043, USA

Editor: Michael Collins

Abstract

Confidence-weighted online learning is a generalization of margin-based learning of linear classifiers in which the margin constraint is replaced by a probabilistic constraint based on a distribution over classifier weights that is updated online as examples are observed. The distribution captures a notion of confidence on classifier weights, and in some cases it can also be interpreted as replacing a single learning rate by adaptive per-weight rates. Confidence-weighted learning was motivated by the statistical properties of natural-language classification tasks, where most of the informative features are relatively rare. We investigate several versions of confidence-weighted learning that use a Gaussian distribution over weight vectors, updated at each observed example to achieve high probability of correct classification for the example. Empirical evaluation on a range of text-categorization tasks show that our algorithms improve over other state-of-the-art online and batch methods, learn faster in the online setting, and lead to better classifier combination for a type of distributed training commonly used in cloud computing.

Keywords: online learning, confidence prediction, text categorization

1. Introduction

While online learning is among the oldest approaches to machine learning, starting with the perceptron algorithm (Rosenblatt, 1958), it is still one of the most popular and and successful for many practical tasks. In online learning, algorithms operate in rounds, whereby the algorithm is shown a single example for which it must first make a prediction and then update its hypothesis once it has seen the correct label. While predictions traditionally take the form of either positive or negative labels (binary classification), algorithms have been extended to a variety of multi-class, regression, ranking and structured prediction problems. By operating one example at a time, online methods are fast, simple, make few assumptions about the data, and perform fairly well across many domains and tasks. For those reasons, online methods are often favored for large data problems, and they are also a natural fit for systems that learn from interaction with a user or another system. In addition to

their nice empirical properties, online algorithms have been analyzed in the mistake bound model (Littlestone, 1989), which supports both theoretical and empirical comparisons of performance. Cesa-Bianchi and Lugosi (2006) provide an in-depth analysis of online learning algorithms.

Much of the machine learning in natural-language processing (NLP) is based on linear classifiers over very high dimension sparse representations of the input trained on large training sets. These properties make online learning a natural choice. Extensions of online learning to structured problems (Collins, 2002; McDonald et al., 2004) achieved some of the best results in structured tasks such as part-of-speech tagging (Collins, 2002; Shen et al., 2007), text segmentation (McDonald et al., 2005a), noun-phrase chunking (Collins, 2002), parsing (McDonald et al., 2005b; Carreras et al., 2008), and machine translation (Chiang et al., 2008). Popular online methods for those tasks include the perceptron (Rosenblatt, 1958), passive-aggressive (Crammer et al., 2006a) and exponentiated gradient (Globerson et al., 2007).

Online learning algorithms are typically used as blackboxes in NLP, without consideration of the peculiarities of natural language. Feature representations of text for tasks from spam filtering to parsing need to capture the variety of words, word combinations, and word attributes in the text, yielding very high-dimensional feature vectors, even though most of the features are absent in most texts. Nevertheless, those many rare features are very informative about the examples that contain them; indeed, features that occur frequently are typically less informative, hence the common use of stop-lists of frequent words such as function words, and of tf-idf term weighting.[1] In Figure 1, we show the most predictive features for a simple NLP classification task and their frequency in data. Notice that while some predictive features are very common, most are relatively rare, indicating that modeling even infrequent features may be useful for learning. Therefore, it is worth investigating whether learning algorithms for linear classifiers could be improved to take advantage of these particularities of natural language data.

The foregoing motivation led us to propose *confidence-weighted* (CW) learning, a class of online learning methods that maintain a probabilistic measure of confidence in each weight. Less confident weights are updated more aggressively than more confident ones. Weight confidence is formalized with a Gaussian distribution over weight vectors, which is updated for each new training example so that the probability of correct classification for that example under the updated distribution meets a specified confidence. The result is an algorithm with superior classification accuracy over state-of-the-art online and batch baselines, faster learning, and new classifier combination methods for parallel training.

While our motivation for CW learning is from observations about NLP problems, the approach makes no assumptions about the input space and can be applied to other machine learning problems (Ma et al., 2009).

This paper brings together two types of confidence-weighted algorithms originally introduced by Dredze et al. (2008) and Crammer et al. (2008). In addition to a unified presentation, we include alternative formulations of the diagonal covariance algorithms along with empirical results. We also include further empirical evidence of the strength of these methods and an analysis of algorithmic behavior on NLP problems.

1. We note that data sparsity is different from model sparsity. Sparsifying regularizers, such as those that constrain the L_1 norm of weight vectors (Andrew and Gao, 2007; Gao et al., 2007). are often proposed to remove redundant features in very high-dimensional data, but they are complementary to the methods we present here to learn better in the presence of many rare but relevant features.

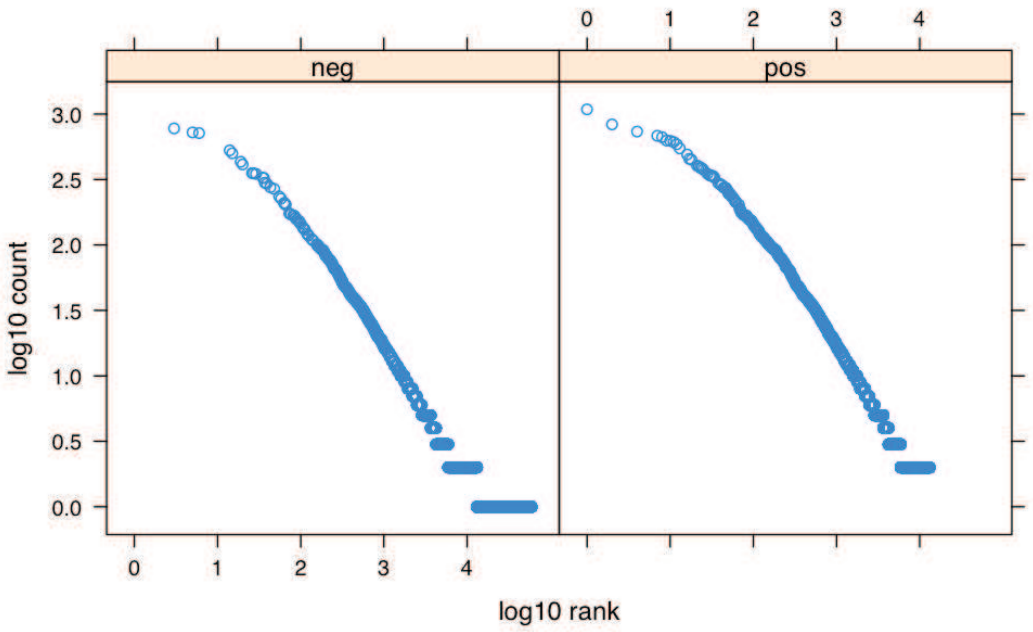


Figure 1: The top quartile of negative (left) and positive (right) features as ranked by mutual information with the label for sentiment data (described in Section 7). The x-axis is their (log) rank by mutual information and the y-axis is their total (log) count in the data. While some very frequent features are useful for predicting the label (high on the curve) there are a large number of low frequency features (low on the curve) that are still useful for learning. A sparse model would likely remove these low frequency features despite their predictive value.

We begin with a discussion of the motivating particularities of natural language data. We then introduce the confidence-weighted framework. From this framework we derive two types of algorithm following different formulations of the main constraint, each with a full covariance and several diagonalized versions. A series of experiments shows CW learning's empirical benefits and an analysis reveals how algorithmic properties manifest themselves empirically. We conclude with a discussion of related work.

2. Characteristics of NLP Data

Extensive experience with building classifiers for a wide range of language processing tasks shows that correct classification requires many specific features, including the presence at specified positions of particular words, affixes, or word combinations (such as bigrams) in the example to be classified. An individual example has a very small fraction of those features, but collectively, examples to be classified may involve a very large number of features ($10^6 - 10^9$), most of which only occur in a few examples. The vector representation of the typical example is a very sparse high

dimensional vector where only a small fraction of elements is nonzero, and feature frequencies have a heavy-tailed distribution (Figure 1).

Online algorithms do well with large numbers of features and examples, but they are not designed specifically for very sparse examples with a heavy-tailed feature frequency distribution. This can have a detrimental effect on learning. Typical linear classifier training algorithms update the weights of binary features only when they occur. The result is many updates for frequent features and few updates for rare features. Similarly, features that occur early in the data stream take more responsibility for correct prediction than those observed later. The result is a model that could have good weight estimates for common features but inaccurate weights for the great majority of features, which occur relatively rarely.

An illustrative case arises in sentiment classification. In this task, a product review is represented as *n*-grams and the goal is to label the review as being positive or negative about the product. Consider a positive review that simply read *"I liked this author."* An online update would increase the weight of both "liked" and "author." Since both are common words, over several examples the algorithm would converge to the correct values, a positive weight for "liked" and zero weight for "author." Now consider a slightly modified negative example: *"I liked this author, but found the book dull."* Since "dull" is a rare feature, the algorithm has a poor estimate of its weight. An update would decrease the weight of both "liked" and "dull." The algorithm does not know that "dull" is rare and the changed behavior is likely caused by the poorly estimated rare feature ("dull") instead of the well estimated common feature ("liked.") An algorithm that maintains no information about the relative frequency or of second order information about features would attribute equal negative weight to both "liked" and "dull", which slows convergence.

This example demonstrates how a lack of memory for previous examples—a property that allows online learning—can hurt learning. A simple solution is to augment an online algorithm with additional information, a memory of past examples. Specifically, the algorithm can maintain a *confidence value for each feature weight*. For example, assuming binary features, the algorithm could keep a count of the number of times each feature has been observed or how many times each weight has been updated. The larger the count, the more confidence we have in the weight of that feature. These estimates are then used to influence weight updates. Instead of equally updating every feature weight for the on-features of an example, the update favors changing low-confidence weights more aggressively than high-confidence ones. At each update, the confidence in the weights of observed features is increased, which will focus the update on the low confidence weights. In the example above, the update would decrease the weight of "dull" but make only a small change to "liked" since the algorithm already has a good estimate of this weight.

In the next section, we use this motivation from language data to present a new family of learning algorithms that associate a confidence value with each weight. For now, we wish to dispel two potential misinterpretations of the preceding very informal argument. First, while our approach is motivated by learning with sparse binary features with a heavy-tailed frequency distribution, the algorithms do not depend on those assumptions. Second, our notion of weight confidence is based on a probabilistic interpretation of passive-aggressive online learning, which differs from the more familiar Bayesian learning for linear classifiers. Nevertheless, analogously to Bayesian learning, it can be used to provide a useful notion of prediction confidence through a margin distribution (Dredze and Crammer, 2008a,b; Dredze et al., 2010).

A summary of the notation used throughout this paper appears in Table 1.

x_i	Example on round i
$\hat{y}_i$	Prediction on round i
y_i	Label on round i
w_i	Weight vector on round i
μ_i	The mean of the distribution on round i
Σ_i	The covariance matrix of the distribution on round i
m_i	Margin on round i
v_i	Margin variance on round i
η	Confidence level
ϕ	The free parameter for CW, defined as $\phi = \Phi^{-1}(\eta)$

Table 1: A reference table for notation used throughout the paper.

3. Online Learning of Linear Classifiers

Online algorithms operate in rounds, where each round corresponds to a single example. On round i the algorithm receives an example $x_i \in \mathbb{R}^d$ to which it applies its current prediction rule to produce a prediction $\hat{y}_i \in \{-1,+1\}$ (for binary classification). It then receives the true label $y_i \in \{-1,+1\}$ and suffers a loss $\ell(y_i,\hat{y}_i)$, which in this work will be the zero-one loss: $\ell(y_i,\hat{y}_i) = 1$ if $y_i \neq \hat{y}_i$ and $\ell(y_i,\hat{y}_i) = 0$ otherwise. The algorithm then updates its prediction rule and proceeds to the next round. For online evaluations, error is reported as the total loss ℓ on the training data and in batch evaluations, error is reported on held out data.

As is common in linear classification, our prediction rules are linear threshold functions

$$f_w(x) : f_w(x) = \text{sign}(x \cdot w) \ .$$

Two functions f_w and f_{cw} are the same for non-negative c. Thus, we can identify f_w with w, which we will do in what follows.

The signed *margin* of an example (x,y) with respect to a specific classifier w is defined to be $y(w \cdot x)$. The sign of the margin is positive iff the classifier w correctly predicts the true label y. The absolute value of the margin $|y(w \cdot x)| = |w \cdot x|$ can be thought of as the *confidence* [2] in the prediction, with larger positive values corresponding to more confident correct predictions. We denote the margin at round i by $m_i = y_i(w_i \cdot x_i)$.

A variety of linear classifier training algorithms, including the perceptron and linear support vector machines, restrict w to be a linear combination of the input examples. Online algorithms of that kind typically have updates of the form

$$w_{i+1} = w_i + \alpha_i y_i x_i \ , \tag{1}$$

for some non-negative coefficients α_i.

In this paper we focus on passive-aggressive (PA) updates (Crammer et al., 2006a) for linear classifiers. After predicting with w_i on the ith round and receiving the true label y_i, the algorithm

2. Note that we use the term "confidence" here as is commonly used in the literature to refer to the size of the margin. This should not be confused with the idea of weight confidence used in this work. In fact, while margin size is often taken as prediction confidence, such as in active learning (Tong and Koller, 2001), this interpretation is open to debate.

updates the prediction function such that the example (x_i, y_i) will be classified correctly with a fixed margin (which can always be scaled to 1):

$$w_{i+1} = \min_{w} \quad \frac{1}{2}\|w_i - w\|^2 \\ \text{s.t.} \quad y_i(w \cdot x_i) \geq 1 \ . \tag{2}$$

The general form of this problem is to enforce some learning constraints, in this case a prediction margin on the example, while minimizing the divergence to the current weights, which are assumed to be good since they encapsulate all previously observed examples. Solving this problem leads to an update of the form given by (1) with coefficient α_i defined on each round as:

$$\alpha_i = \frac{\max\{1 - y_i(w_i \cdot x_i), 0\}}{\|x_i\|^2} \ , \tag{3}$$

Like the perceptron, this is a mistake driven update, whereby $\alpha_i > 0$ iff the learning condition was not met, ie. the example was not classified with a margin of at least 1. Note that the numerator of (3) is the hinge loss, which is zero only if the example is classified with a margin of 1. In practice, slack variables are introduced for non-separable data, restricting (3) as $\max\{\alpha_i, C\}$, for some free parameter C.

Crammer et al. (2006a) provide a theoretical analysis of algorithms of this form, which have been shown to work well in a variety of applications (McDonald et al., 2004, 2005a; Chiang et al., 2008).

4. Distributions over Classifiers

Following the motivation of Section 2, we need a notion of confidence for the weight vector w maintained by an online learner for linear classifiers. Before any examples are seen, all of the weights in w are equally uncertain. As examples are observed, the confidence in the weights of features that are often active should increase faster than the confidence in the weights of rarely seen features.

Our concrete implementation of this idea is to represent the state of the learner with a probability density over w, specifically a Gaussian distribution $\mathcal{N}(\mu, \Sigma)$ with mean $\mu \in \mathbb{R}^d$ and covariance matrix $\Sigma \in \mathbb{R}^{d \times d}$. The values μ_p and $\Sigma_{p,p}$ represent knowledge of and confidence in the weight of feature p. The smaller $\Sigma_{p,p}$, the more confidence we have in the mean weight value μ_p. Each covariance term $\Sigma_{p,p'}$ captures our knowledge of the interaction between features p and p'. The Gaussian distribution naturally matches our intuition for confidence, as the covariance of the distribution is inversely proportional to our confidence: the smaller the determinant of the covariance, the less we expect the true weight value to deviate from the current estimate. This Gaussian representation is illustrated in Figure 2, which shows a Gaussian distribution over two-dimensional weight vectors. The black line represents an example $x = (0.5, 1)$, $y = +1$, which divides the space between classifiers that correctly classify this point (blue crosses below) and those that classify it incorrectly (green dots above).

In the CW model, the traditional signed margin $y(w \cdot x)$ becomes a univariate Gaussian random variable M, where the mean of the distribution is the signed margin,

$$M \sim \mathcal{N}\left(y(\mu \cdot x), x^\top \Sigma x\right) \ . \tag{4}$$

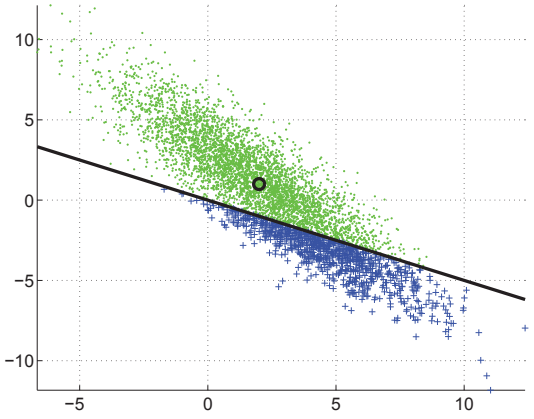


Figure 2: Gaussian distribution over two-dimensional weight vectors. Points above the black line (green dots) incorrectly classify the example $((0.5,1),+1)$ and points below the line (blue crosses) classify it correctly. The density around a point is proportional to its relative weight. The black circle marks the mean of the Gaussian.

There are several ways to make predictions in this framework. A Gibbs predictor samples from the distribution a single weight vector w, which is equivalent to drawing a margin value using (4), and takes its sign as the prediction. Other alternatives use averaging rather than sampling. For example, we can use the average weight vector $\mathrm{E}[w] = \mu$, as is done in Bayes point machines (Herbrich et al., 2001), which use a single weight vector to approximate a distribution. Alternatively, we can use the average margin $\mathrm{E}[M]$. These two approaches are equivalent by linearity of expectation, $\mathrm{E}[w \cdot x] = \mu \cdot x$. Another approach estimates $\mathrm{E}[\mathrm{sign}(M)]$ from many draws of w for fixed μ,Σ, and x. Since the sign function attains only two values (-1 or $+1$) this is equivalent to computing the probability of a *correct* prediction (not a large margin prediction), given by

$$\Pr[M \geq 0] = \Pr_{w \sim \mathcal{N}(\mu,\Sigma)}[y(w \cdot x) \geq 0].$$

When possible we omit the explicit dependence on the distribution parameters and simply write $\Pr[y(w \cdot x) \geq 0]$. If the probability is larger than half, then the (weighted) majority votes for $y = +1$, otherwise, for $y = -1$. Note that from the discussion below this prediction rule is equivalent to the previous two. Conceptually, it is useful to think of prediction as drawing a weight vector w from the distribution, ie. $w \sim \mathcal{N}(\mu,\Sigma)$, and predicting the label according to the sign of $w \cdot x$. However, as we said above, the average of many such draws is equivalent to the simple prediction rule $\mathrm{sign}(\mu \cdot x)$, which we will use in what follows.

5. Learning Confidence-Weighted Classifiers

In the previous section we formalized our confidence-weighted learning framework in terms of Gaussian distributions over weight vectors. In this section we discuss how to learn such distributions.

CW is an online learning algorithm, so on round i the algorithm receives example x_i for which it issues a prediction $\hat{y}_i$.[3] The algorithm predicts $\hat{y}_i$ as $\text{sign}(\mu_i \cdot x_i)$, which is equivalent to averaging the predictions of many sampled weight vectors from the distribution. On being presented with the label y_i, the algorithm adjusts the distribution to enforce a learning condition. Following the intuition underlying the PA algorithms of Crammer et al. (2006a), we require that an update achieves both a large margin on the example and minimizes the change in weights. In this case, a large prediction margin is formalized as ensuring that the probability of a correct prediction for training example i is no smaller than the confidence level $\eta \in [0,1]$:

$$\Pr\left[y_i\left(w \cdot x_i\right) \geq 0\right] \geq \eta \ .$$

Minimization of weight changed is enforced by finding a new distribution closest in the KL divergence[4] sense to the current distribution $\mathcal{N}(\mu_i, \Sigma_i)$. Thus, on round i, the algorithm updates the distribution by solving the following optimization problem:

$$(\mu_{i+1}, \Sigma_{i+1}) = \min \mathrm{D}_{\mathrm{KL}}\left(\mathcal{N}(\mu, \Sigma) \,\|\, \mathcal{N}(\mu_i, \Sigma_i)\right) \tag{5}$$

$$\text{s.t. } \Pr\left[y_i\left(w \cdot x_i\right) \geq 0\right] \geq \eta \ . \tag{6}$$

This update can be understood as a probabilistic counterpart of the PA objective (2).

We now develop both the objective and the constraint of this optimization problem following Boyd and Vandenberghe (2004, page 158). We start with the objective (5) and write the KL divergence between two Gaussians as

$$\mathrm{D}_{\mathrm{KL}}\left(\mathcal{N}(\mu_0, \Sigma_0) \,\|\, \mathcal{N}(\mu_1, \Sigma_1)\right) = \frac{1}{2}\left(\log\left(\frac{\det \Sigma_1}{\det \Sigma_0}\right) + \mathrm{Tr}\left(\Sigma_1^{-1}\Sigma_0\right) + \left(\mu_1 - \mu_0\right)^{\top} \Sigma_1^{-1}\left(\mu_1 - \mu_0\right) - d\right) \ .$$

We now proceed with the constraint in (6). As noted above, under the distribution $\mathcal{N}(\mu, \Sigma)$, the margin for (x_i, y_i) has a Gaussian distribution with mean

$$m_i = y_i\left(\mu_i \cdot x_i\right) \ , \tag{7}$$

and variance

$$\sigma_i^2 = v_i = x_i^{\top} \Sigma_i x_i \ . \tag{8}$$

Thus the probability of a *wrong* classification is

$$\Pr\left[M \leq 0\right] = \Pr\left[\frac{M - m}{\sigma} \leq \frac{-m}{\sigma}\right] \ .$$

Since $(M - m)/\sigma$ is a normally distributed random variable, the above probability equals $\Phi(-m/\sigma)$, where

$$\Phi(u) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{u} e^{-v^2} dv \ ,$$

3. For a related batch formulation of CW learning, see recent work of Crammer et al. (2009b).

4. $\mathrm{D}_{\mathrm{KL}}\left(p(x) \,\|\, q(x)\right) = \int p(x) \log\left(\frac{p(x)}{q(x)}\right) dx$.

is the cumulative Gaussian distribution. Thus we can rewrite (6) as

$$\frac{-m}{\sigma} \leq \Phi^{-1}(1-\eta) = -\Phi^{-1}(\eta) \ .$$

Substituting m and σ by their definitions and rearranging terms we obtain

$$y_i(\mu \cdot x_i) \geq \phi \sqrt{x_i^\top \Sigma x_i} \ ,$$

where $\phi = \Phi^{-1}(\eta)$. To conclude the update rule solves the following optimization problem:

$$(\mu_{i+1}, \Sigma_{i+1}) = \arg\min_{\mu,\Sigma} \frac{1}{2}\log\left(\frac{\det\Sigma_i}{\det\Sigma}\right) + \frac{1}{2}\mathrm{Tr}\left(\Sigma_i^{-1}\Sigma\right) + \frac{1}{2}(\mu_i - \mu)^\top \Sigma_i^{-1}(\mu_i - \mu)$$
$$\text{s.t. } y_i(\mu \cdot x_i) \geq \phi \sqrt{x_i^\top \Sigma x_i} \ . \tag{9}$$

Conceptually, this is a large-margin constraint, where the value of the margin requirement depends on the example x_i via a quadratic form.

Unfortunately, this constraint is not convex in Σ since the term $\sqrt{x_i^\top \Sigma x_i}$ is concave in Σ. We propose two alternatives to obtain a convex constraint: linearization (Section 5.1) and change of variables (Section 5.2). Additionally, we propose few alternatives to solve the learning optimization problem restricted to *diagonal* matrices in Section 6.

5.1 Linearization of the Constraint

In out first approach to obtain a convex problem we simply linearize the constraint of (9) by omitting the square root to obtain the revised optimization problem.

$$(\mu_{i+1}, \Sigma_{i+1}) = \arg\min \frac{1}{2}\log\left(\frac{\det\Sigma_i}{\det\Sigma}\right) + \frac{1}{2}\mathrm{Tr}\left(\Sigma_i^{-1}\Sigma\right) + \frac{1}{2}(\mu_i - \mu)^\top \Sigma_i^{-1}(\mu_i - \mu)$$
$$\text{s.t. } y_i(\mu \cdot x_i) \geq \phi\left(x_i^\top \Sigma x_i\right) \ . \tag{10}$$

We call this formulation `var`, since we have replaced the standard deviation in the constraint with the variance. This formulation was introduced by Dredze et al. (2008). The following lemma summarizes the solution of this formulation,

Lemma 1 *The optimal solution of this form is,*

$$\mu_{i+1} = \mu_i + \alpha y_i \Sigma_i x_i$$
$$\Sigma_{i+1}^{-1} = \Sigma_i^{-1} + 2\alpha\phi x_i x_i^\top \ ,$$

where the value of the parameter α (a Lagrange multiplier) is given by

$$\alpha_i = \max\left\{0, \frac{-(1+2\phi m_i) + \sqrt{(1+2\phi m_i)^2 - 8\phi(m_i - \phi v_i)}}{4\phi v_i}\right\} \ .$$

where $m_i = y_i(\mu_i \cdot x_i)$ (see (7)) and $v_i = x_i^\top \Sigma_i x_i$ (see (8)).

The derivation appears in Section 5.1.1 below. The resulting algorithm is shown in Figure 1, where the update uses (11) and (13) to update the distribution with coefficients β_i ((15)) and α_i ($\max\{(18), 0\}$.)

Algorithm 1 Binary CW Online Algorithm. The two versions of the Confidence-Weighted algorithm: (1) linearization and (2) change of variables. The numbers in parentheses refer to equations in the text, where more detail can be found.

Input: $\eta \in [0.5, 1]$

Initialize:

$$\mu_1 = 0 \ , \ \Sigma_1 = I,$$
$$\phi = \Phi^{-1}(\eta) \ , \ \phi' = 1 + \phi^2/2 \ , \ \phi'' = 1 + \phi^2 \ .$$

for $i = 1, 2 \ldots$ **do**

Receive a training example $x_i \in \mathbb{R}^d$

Compute Gaussian margin distribution $m_i \sim \mathcal{N}\left(\mu_i \cdot x_i, x_i^\top \Sigma_i x_i\right)$

Receive true label y_i

Suffer loss $\ell_i = 1$ iff $y_i \mathrm{E}\left[\mathrm{sign}\left(m_i\right)\right] \leq 0$

Compute Update:

- Define: $m_i = y_i \left(\mu_i \cdot x_i\right)$ (7) $v_i = x_i^\top \Sigma_i x_i$ (8)
- `Linearization:`

$$\alpha_i = \max\left\{0, \frac{-\left(1 + 2\phi m_i\right) + \sqrt{\left(1 + 2\phi m_i\right)^2 - 8\phi\left(m_i - \phi v_i\right)}}{4\phi v_i}\right\} \tag{18}$$

$$\beta_i = \frac{2\alpha_i \phi}{1 + 2\alpha\phi v_i} \tag{15}$$

- `Change of Variables:`

$$v_i^+ = \left(\frac{-\alpha v_i \phi + \sqrt{\alpha^2 v_i^2 \phi^2 + 4 v_i}}{2}\right)^2 \tag{28}$$

$$\alpha_i = \max\left\{0, \frac{-m_i \phi' + \sqrt{m_i^2 \frac{\phi^4}{4} + v_i \phi^2 \phi''}}{v_i \phi''}\right\} \tag{31}$$

$$\beta_i = \frac{\alpha_i \phi}{\sqrt{v_i^+} + v_i \alpha_i \phi} \tag{27}$$

Update

$$\mu_{i+1} = \mu_i + \alpha_i y_i \Sigma_i x_i \tag{11,20}$$
$$\Sigma_{i+1} = \Sigma_i - \beta_i \Sigma_i x_i x_i^\top \Sigma_i \tag{14,25}$$

end for

Output: Final μ and Σ

5.1.1 DERIVATION OF LEMMA 1

The optimization objective is convex in μ and Σ simultaneously and the constraint became linear, so any convex optimization solver van be used to solve this problem. The Lagrangian for this optimization is

$$\begin{aligned}\mathcal{L} &= \frac{1}{2}\log\left(\frac{\det\Sigma_i}{\det\Sigma}\right)+\frac{1}{2}\mathrm{Tr}\left(\Sigma_i^{-1}\Sigma\right)+\frac{1}{2}\left(\mu_i-\mu\right)^\top\Sigma_i^{-1}\left(\mu_i-\mu\right)\\ &\quad+\alpha\left(-y_i\left(\mu\cdot x_i\right)+\phi\left(x_i^\top\Sigma x_i\right)\right)\ .\end{aligned}$$

Taking partial derivatives, we know that at the optimum, we must have

$$\frac{\partial}{\partial\mu}\mathcal{L}=\Sigma_i^{-1}\left(\mu-\mu_i\right)-\alpha y_i x_i=0\ .$$

Assuming Σ_i is non-singular and rearranging terms we get

$$\mu_{i+1}=\mu_i+\alpha y_i\Sigma_i x_i\ . \tag{11}$$

At the optimum, we must also have

$$\frac{\partial}{\partial\Sigma}\mathcal{L}=-\frac{1}{2}\Sigma^{-1}+\frac{1}{2}\Sigma_i^{-1}+\phi\alpha x_i x_i^\top=0\ , \tag{12}$$

and solving for Σ^{-1} we obtain

$$\Sigma_{i+1}^{-1}=\Sigma_i^{-1}+2\alpha\phi x_i x_i^\top\ . \tag{13}$$

Before proceeding, we observe that (13) computes Σ_{i+1}^{-1} as the sum of a rank-one positive semi-definite (PSD) matrix and Σ_i^{-1} . Thus, if Σ_i^{-1} is PSD, so are Σ_{i+1}^{-1} and Σ_{i+1} thus Σ_i is indeed non-singular, as assumed above. The update guarantees that the eigenvalues of the inverse-covariance matrix always increase.

Finally, we compute the inverse of (13) using the Woodbury identity (Petersen and Pedersen, 2008, Equation 135) and get

$$\begin{aligned}\Sigma_{i+1}&=\left(\Sigma_i^{-1}+2\alpha\phi x_i x_i^\top\right)^{-1}\\ &=\Sigma_i-\Sigma_i x_i\left(\frac{1}{2\alpha\phi}+x_i^\top\Sigma_i x_i\right)^{-1}x_i^\top\Sigma_i\\ &=\Sigma_i-\Sigma_i x_i\frac{2\alpha\phi}{1+2\alpha\phi v_i}x_i^\top\Sigma_i\\ &=\Sigma_i-\beta_i\Sigma_i x_i x_i^\top\Sigma_i\ ,\end{aligned} \tag{14}$$

where

$$\begin{aligned}v_i &= x_i^\top\Sigma_i x_i\\ \beta_i &= \frac{2\alpha\phi}{1+2\alpha\phi v_i}\ .\end{aligned} \tag{15}$$

The KKT conditions for the optimization imply that the either $\alpha = 0$, and no update is needed, or the constraint in (10) is an equality after the update. Substituting (11) and (14) into the equality version of (10), we obtain

$$y_i \left(x_i \cdot (\mu_i + \alpha y_i \Sigma_i x_i) \right) = \phi \left(x_i^\top \left(\Sigma_i - \Sigma_i x_i \beta_i x_i^\top \Sigma_i \right) x_i \right) . \tag{16}$$

Rearranging terms we get

$$y_i \left(x_i \cdot \mu_i \right) + \alpha x_i^\top \Sigma_i x_i = \phi x_i^\top \Sigma_i x_i - \phi v_i^2 \beta_i \ . \tag{17}$$

Substituting (7), (8), and (15) into (17) we obtain

$$m_i + \alpha v_i = \phi v_i - \phi v_i^2 \frac{2\alpha\phi}{1 + 2\alpha\phi v_i} \ .$$

We multiply both sides by $1 + 2\alpha\phi v_i$ and get

$$(m_i + \alpha v_i)(1 + 2\alpha\phi v_i) = \phi v_i (1 + 2\alpha\phi v_i) - 2\alpha\phi^2 v_i^2 \ .$$

Rearranging the terms we obtain,

$$\begin{aligned} 0 &= m_i + \alpha v_i + 2\alpha\phi v_i m_i + 2\alpha^2 \phi v_i^2 - \phi v_i \\ &= \alpha^2 \left(2\phi v_i^2 \right) + \alpha v_i (1 + 2\phi m_i) + (m_i - \phi v_i) \ . \end{aligned}$$

The above equality is a quadratic equation in α. Its smaller root is always negative and thus is not a valid Lagrange multiplier. Let γ_i be its larger root:

$$\gamma_i = \frac{-(1 + 2\phi m_i) + \sqrt{(1 + 2\phi m_i)^2 - 8\phi (m_i - \phi v_i)}}{4\phi v_i} \ . \tag{18}$$

The constraint (10) is satisfied before the update if $m_i - \phi v_i \geq 0$. If $1 + 2\phi m_i \leq 0$, then $m_i \leq \phi v_i$ and from (18) we have that $\gamma_i > 0$. If, instead, $1 + 2\phi m_i \geq 0$, then, again by (18), we have

$$\begin{aligned} &\gamma_i > 0 \\ &\Leftrightarrow \sqrt{(1 + 2\phi m_i)^2 - 8\phi (m_i - \phi v_i)} > (1 + 2\phi m_i) \\ &\Leftrightarrow m_i < \phi v_i \ . \end{aligned}$$

From the KKT conditions, either $\alpha_i = 0$ or (10) is satisfied as an equality. In the later case, (16) holds, and thus $\alpha_i = \gamma_i > 0$, which concludes the derivation of the lemma.

5.2 Change of Variables

While linearization yielded a closed form convex solution to our optimization, it required approximating the constraint. We now proceed with the second alternative of obtaining a convex optimization problem by a change of variables, which allows us to achieve an exact convex update.

Since Σ is positive-semidefinite (PSD) it can be written as the square of another PSD matrix[5] Υ:

$$\Sigma = \Upsilon^2 \ , \ \Upsilon = \sqrt{\Sigma} \, .$$

Substituting in (9) gives the revised optimization problem

$$\begin{aligned} (\mu_{i+1}, \Upsilon_{i+1}) = \arg\min \ & \frac{1}{2} \log \left(\frac{\det \Upsilon_i^2}{\det \Upsilon^2} \right) + \frac{1}{2} \mathrm{Tr} \left(\Upsilon_i^{-2} \Upsilon^2 \right) + \frac{1}{2} (\mu_i - \mu)^\top \Upsilon_i^{-2} (\mu_i - \mu) \\ \text{s.t.} \ & y_i (\mu \cdot x_i) \geq \phi \| \Upsilon x_i \| \\ & \Upsilon \text{ is PSD} \, . \end{aligned} \tag{19}$$

Note that, the objective is convex since $-\log\det\Upsilon^2 = -2\log\det\Upsilon$ which is well defined since Υ is PSD. The constraint is a second-order cone inequality and therefore convex.

We call this formulation `stdev`, since we have maintained the standard deviation in the constraint. This formulation was introduced by Crammer et al. (2008).

Standard optimization techniques can solve the convex program (19), but these methods can be slow. Instead, as before we derive a closed-form solution which we summarize in the following lemma:

Lemma 2 *The optimal solution of this form is,*

$$\begin{aligned} \mu_{i+1} &= \mu_i + \alpha y_i \Sigma_i x_i \\ \Sigma_{i+1} &= \Sigma_i - \beta \Sigma_i x_i x_i^\top \Sigma_i \, , \end{aligned}$$

where

$$\beta = \frac{\alpha\phi}{\sqrt{v_i^+} + v_i \alpha \phi} \quad , \quad v_i^+ = x_i^\top \Sigma_{i+1} x_i \, .$$

and the value of the parameter α (a Lagrange multiplier) is given by

$$\alpha = \max \left\{ 0, \frac{1}{v_i} \frac{-m_i \phi' + \sqrt{m_i^2 \frac{\phi^4}{4} + v_i \phi^2 \phi''}}{\phi''} \right\} \, .$$

where $m_i = y_i (\mu_i \cdot x_i)$ (see (7)), $v_i = x_i^\top \Sigma_i x_i$ (see (8)), and for simplicity we define $\phi' = 1 + \phi^2/2$, $\phi'' = 1 + \phi^2$.

The resulting algorithm is shown in Figure 1.

5.2.1 DERIVATION OF LEMMA 2

The Lagrangian for (19) is

$$\mathcal{L} = \frac{1}{2} \log \left(\frac{\det \Upsilon_i^2}{\det \Upsilon^2} \right) + \frac{1}{2} \mathrm{Tr} \left(\Upsilon_i^{-2} \Upsilon^2 \right) + \frac{1}{2} (\mu_i - \mu)^\top \Upsilon_i^{-2} (\mu_i - \mu) + \alpha \left(-y_i (\mu \cdot x_i) + \phi \| \Upsilon x_i \| \right) \, .$$

5. We use a decomposition in terms of PSD matrices because it yields a convex optimization problem. In general, a PSD matrix Σ can be written as $\Sigma = AA^\top$, which is not convex because it is rotation-invariant. Alternatively, any symmetric S matrix can be used $\Sigma = S^2$, but this is not convex either, since it is invariant to reflections.

At the optimum, it must be that

$$\frac{\partial}{\partial \mu}\mathcal{L} = \Upsilon_i^{-2}(\mu - \mu_i) - \alpha y_i x_i = 0 \,.$$

Therefore, if Υ_i is non-singular, the update for the mean is

$$\mu_{i+1} = \mu_i + \alpha y_i \Upsilon_i^2 x_i \,. \tag{20}$$

At the optimum, we must also have

$$\frac{\partial}{\partial \Upsilon}\mathcal{L} = -\Upsilon^{-1} + \frac{1}{2}\Upsilon_i^{-2}\Upsilon + \frac{1}{2}\Upsilon\Upsilon_i^{-2} + \alpha\phi\frac{x_i x_i^\top \Upsilon}{2\sqrt{x_i^\top \Upsilon^2 x_i}} + \alpha\phi\frac{\Upsilon x_i x_i^\top}{2\sqrt{x_i^\top \Upsilon^2 x_i}} = 0 \,. \tag{21}$$

Defining the matrix

$$C = \Upsilon_i^{-2} + \alpha\phi\frac{x_i x_i^\top}{\sqrt{x_i^\top \Upsilon^2 x_i}} \,, \tag{22}$$

we get

$$\frac{\partial}{\partial \Upsilon}\mathcal{L} = -\Upsilon^{-1} + \frac{1}{2}\Upsilon C + \frac{1}{2}C\Upsilon = 0$$

at the optimum. From this, it follows easily that at the optimum

$$\Upsilon = C^{-\frac{1}{2}} \,.$$

Substituting (22) into this equation, we obtain the update

$$\Upsilon_{i+1}^{-2} = \Upsilon_i^{-2} + \alpha\phi\frac{x_i x_i^\top}{\sqrt{x_i^\top \Upsilon_{i+1}^2 x_i}} \,.$$

Conveniently, the final form of the updates can be expressed in terms of the covariance matrix:[6]

$$\mu_{i+1} = \mu_i + \alpha y_i \Sigma_i x_i \tag{23}$$

$$\Sigma_{i+1}^{-1} = \Sigma_i^{-1} + \alpha\phi\frac{x_i x_i^\top}{\sqrt{x_i^\top \Sigma_{i+1} x_i}} \,. \tag{24}$$

As before we observe that if Σ_i^{-1} is PSD, so are Σ_{i+1}^{-1} and Σ_{i+1} with monotonically decreasing eigenvalues. Thus Σ_i is indeed non-singular, as assumed above.

6. Furthermore, writing the Lagrangian of (10) and solving it would yield the same solution as Equations (23,24). Thus the optimal solution of both (10) and (19) are the same.

It remains to determine the value of the Lagrange multiplier α. As before we compute the inverse of (24) using the Woodbury identity (Petersen and Pedersen, 2008) to get,

$$\begin{aligned}
\Sigma_{i+1} &= \left(\Sigma_i^{-1} + \alpha\phi \frac{x_i x_i^\top}{\sqrt{x_i^\top \Sigma_{i+1} x_i}}\right)^{-1} \\
&= \Sigma_i - \Sigma_i x_i \left(\frac{\sqrt{x_i^\top \Sigma_{i+1} x_i}}{\alpha\phi} + x_i^\top \Sigma_i x_i\right)^{-1} x_i^\top \Sigma_i \\
&= \Sigma_i - \Sigma_i x_i \left(\frac{\alpha\phi}{\sqrt{x_i^\top \Sigma_{i+1} x_i} + x_i^\top \Sigma_i x_i \alpha\phi}\right) x_i^\top \Sigma_i \\
&= \Sigma_i - \beta_i \Sigma_i x_i x_i^\top \Sigma_i \ . \qquad (25)
\end{aligned}$$

where we define

$$v_i^+ = x_i^\top \Sigma_{i+1} x_i \ , \qquad (26)$$

and

$$\beta_i = \frac{\alpha_i \phi}{\sqrt{v_i^+} + v_i \alpha_i \phi} \ . \qquad (27)$$

Multiplying (25) by $x_i^\top$ (left) and x_i (right) we get

$$v_i^+ = v_i - v_i \left(\frac{\alpha\phi}{\sqrt{v_i^+} + v_i \alpha\phi}\right) v_i \ ,$$

which is equivalent to

$$\begin{aligned}
v_i^+ \sqrt{v_i^+} + v_i^+ v_i \alpha\phi &= v_i \sqrt{v_i^+} + v_i^2 \alpha\phi - v_i^2 \alpha\phi \\
&= v_i \sqrt{v_i^+} \ .
\end{aligned}$$

Dividing both sides by $\sqrt{v_i^+}$, we obtain

$$v_i^+ + \sqrt{v_i^+} v_i \alpha\phi - v_i = 0 \ ,$$

which can be solved for v_i^+ to obtain

$$\sqrt{v_i^+} = \frac{-\alpha v_i \phi + \sqrt{\alpha^2 v_i^2 \phi^2 + 4 v_i}}{2} \ . \qquad (28)$$

The KKT conditions for the optimization imply that either $\alpha = 0$ and no update is needed, or the constraint (19) is an equality after the update.

Using the equality version of (19) and Equations (23,25,26,28) we obtain

$$m_i + \alpha v_i = \phi \frac{-\alpha v_i \phi + \sqrt{\alpha^2 v_i^2 \phi^2 + 4 v_i}}{2} \ , \tag{29}$$

which can be rearranged into the following quadratic equation in α:

$$\alpha^2 v_i^2 \left(1 + \phi^2\right) + 2\alpha m_i v_i \left(1 + \frac{\phi^2}{2}\right) + \left(m_i^2 - v_i \phi^2\right) = 0 \ .$$

The smaller root of this equation is always negative and thus not a valid Lagrange multiplier. We use the following abbreviations for writing the larger root γ_i,

$$\phi' = 1 + \phi^2/2 \quad ; \quad \phi'' = 1 + \phi^2 \ .$$

The larger root is then

$$\gamma_i = \frac{-m_i v_i \phi' + \sqrt{m_i^2 v_i^2 \phi'^2 - v_i^2 \phi'' \left(m_i^2 - v_i \phi^2\right)}}{v_i^2 \phi''} \ . \tag{30}$$

The constraint (19) is satisfied before the update if $m_i - \phi\sqrt{v_i} \geq 0$. If $m_i \leq 0$, then $m_i \leq \phi\sqrt{v_i}$ and from (30) we have that $\gamma_i > 0$. If instead $m_i \geq 0$, then, again by (30), we have

$$\begin{aligned} &\gamma_i > 0 \\ &\Leftrightarrow m_i v_i \phi' < \sqrt{m_i^2 v_i^2 \phi'^2 - v_i^2 \phi' \left(m_i^2 - v_i \phi^2\right)} \\ &\Leftrightarrow m_i < \phi v_i \ . \end{aligned}$$

From the KKT conditions, either $\alpha_i = 0$ or (10) is satisfied as an equality, so (29) holds and $\alpha_i = \gamma_i > 0$.

The solution of (30) satisfies the KKT conditions, that is either $\alpha_i \geq 0$ or the constraint of (10) is satisfied before the update with the weights μ_i and Σ_i. We obtain the final form of α_i by simplifying (30) together with last comment and get,

$$\alpha_i = \max \left\{ 0, \frac{1}{v_i} \frac{-m_i \phi' + \sqrt{m_i^2 \frac{\phi^4}{4} + v_i \phi^2 \phi''}}{\phi''} \right\} \ . \tag{31}$$

6. Diagonal Covariance Matrices

So far we have said nothing about the covariance matrix Σ, which grows quadratically in the number of features. Since our intended applications are NLP tasks, computing the full matrix Σ is computationally infeasible. Additionally, even though we initialize the matrix to be diagonal (Figure 1), after applying the updates rule of either (14) (linearization/`var`) or (25) (change of variables/`stdev`), we may obtain a full covariance matrix, as we subtract from Σ_i a rank-one matrix proportional to the outer product of Σx. Therefore, successful applications to NLP problems require a restriction on the size of the matrix Σ.

In this section, we reduce the size of Σ by restriction to a diagonal-covariance matrix.[7] We discuss two main approaches, each of which can be applied to either the linearization or the change of variables formulations. In Section 6.1 we show two ways to use the full-covariance updates discussed above and add a diagonalization step. In Section 6.2 we take an alternative approach and re-develop the update step assuming an explicit diagonal representation of the covariance matrix.

6.1 Approximate Diagonal Update

Both updates above (linearization or change-of-variables) share the same form when updating the covariance matrix ((14) or (25))

$$\Sigma_{i+1} = \Sigma_i - \beta_i \Sigma_i x_i x_i^\top \Sigma_i \ . \tag{32}$$

Our diagonalization step will define the final matrix to be a diagonal matrix with its non-zero elements equals to the diagonal elements of (32). Formally we get,

$$\begin{aligned} \Sigma_{i+1} &= \operatorname{diag}\left(\Sigma_i - \beta_i \Sigma_i x_i x_i^\top \Sigma_i\right) \\ &= \operatorname{diag}(\Sigma_i) - \operatorname{diag}\left(\beta_i \Sigma_i x_i x_i^\top \Sigma_i\right) \\ &= \Sigma_i - \beta_i \operatorname{diag}\left(\Sigma_i x_i x_i^\top \Sigma_i\right) \ , \end{aligned}$$

where the last equality follows since we assume that Σ_i is diagonal and

$$\operatorname{diag}(A) = \begin{cases} A_{p,p'} & p = p' \\ 0 & p \neq p' \end{cases} \ .$$

A naïve implementation of the diagonal operator takes $\Theta(d^2)$ time and space. An efficient implementation first defines $z_i = \Sigma_i x_i$ and then sets,

$$\left(\Sigma_{i+1}\right)_{p,p} = \left(\Sigma_i\right)_{p,p} - \beta_i \left[\left(z_i\right)_p\right]^2 \text{ for } p = 1, \ldots, d \ .$$

We refer to this diagonalization scheme as L_2 since it is equivalent to a projection of the full matrix onto the set of diagonal matrices using the Euclidean norm.

We note in passing that since the diagonalization operator and the inverse operator are *not* commutative, we can first diagonalize the inverse of the covariance matrix and then invert the result. Concretely we start from the update of the inverse-covariance,

$$\Sigma_{i+1}^{-1} = \Sigma_i^{-1} + \eta_i x_i x_i^\top \ ,$$

where

$$\eta_i = 2\alpha_i \phi \ ,$$

for the linearization approach ((13)) and

$$\eta_i = \frac{\alpha_i \phi}{\sqrt{x_i^\top \Sigma_{i+1} x_i}} \ ,$$

7. There are other possible choices for reducing the matrix size, such as enforcing a sparse block diagonal matrix. We select diagonalization since it is the most straightforward reduction and yields a first order model. See a recent paper by Ma et al. (2010) for low rank options.

for the change of variables approach ((24)). We first diagonalize the inverse-covariance and get

$$\begin{aligned}\Sigma_{i+1}^{-1} &= \operatorname{diag}\left(\Sigma_i^{-1} + \eta_i x_i x_i^\top\right) \\ &= \operatorname{diag}\left(\Sigma_i^{-1}\right) + \operatorname{diag}\left(\eta_i x_i x_i^\top\right) \\ &= \Sigma_i^{-1} + \eta_i \operatorname{diag}\left(x_i x_i^\top\right) .\end{aligned}$$

As before we implement the update efficiently by writing

$$\left(\Sigma_{i+1}^{-1}\right)_{p,p} = \left(\Sigma_i^{-1}\right)_{p,p} + \eta_i \left[\left(x_i\right)_p\right]^2 \text{ for } p = 1,\ldots,d ,$$

or in terms of the covariance matrix

$$\left(\Sigma_{i+1}\right)_{p,p} = \frac{1}{\frac{1}{\left(\Sigma_i\right)_{p,p}} + \eta_i \left[\left(x_i\right)_p\right]^2} \text{ for } p = 1,\ldots,d .$$

We refer to this diagonalization scheme as KL since it is equivalent to a projection of the full matrix onto the set of diagonal matrices using the Kullback-Leibler (KL) divergence.

6.2 Exact Diagonal Update

An alternative to the approximate formulation is to explicitly maintain a diagonal and develop a corresponding update. We now assume that the matrix Σ is diagonal. We denote by $\Sigma_{i,(p)}$ the rth diagonal element of the matrix Σ_i, and by $x_{i,(p)}$ the rth element of x_i. We start with the first alternative above where we used linearization. We follow the derivation of Section 5.1 until (11). Proceeding with derivation of (12), but only for the diagonal elements indexed by p we get,

$$\frac{\partial}{\partial \Sigma_{(p)}} \mathcal{L} = -\frac{1}{2\Sigma_{(p)}} + \frac{1}{2\Sigma_{i,(p)}} + \phi\alpha x_{i,(p)}^2 = 0 \text{ for } p = 1,\ldots,d ,$$

Solving for $\Sigma_{(p)}$ we get

$$\Sigma_{i+1,(p)} = \frac{\Sigma_{i,(p)}}{1 + 2\alpha\Sigma_{i,(p)}\phi x_{i,(p)}^2} .$$

Following the logic presented after (15) we get that at the optimum we have

$$y_i \left(x_i \cdot (\mu_i + \alpha y_i \Sigma_i x_i)\right) = \phi \sum_p x_{i,(p)}^2 \frac{\Sigma_{i,(p)}}{1 + 2\alpha\Sigma_{i,(p)}\phi x_{i,(p)}^2} .$$

Substituting (7) and (8) and rearranging the terms we get the constraint

$$f(\alpha) = 0 ,$$

where we defined

$$f(\alpha) = m_i + \alpha v_i - \sum_r \frac{\Sigma_{i,(p)}\phi x_{i,(p)}^2}{1 + 2\alpha\Sigma_{i,(p)}\phi x_{i,(p)}^2} . \tag{33}$$

We will analyze (33) after developing its equivalent for the second alternative above where we perform a change of variables. As above we denote by $\Upsilon_{i,(p)}$ the rth diagonal element of the matrix Υ_i. We follow the derivation of Section 5.2 until (20). Proceeding with derivation of (21), but only for the diagonal elements indexed by r we get

$$\frac{\partial}{\partial \Upsilon_{(p)}} \mathcal{L} = -\Upsilon_{(p)}^{-1} + \Upsilon_{i,(p)}^{-2} \Upsilon_{(p)} + \alpha\phi \frac{x_{i,(p)}^2 \Upsilon_{(p)}}{\sqrt{x_i^\top \Upsilon^2 x_i}} = 0 \,.$$

Rearranging the terms we get

$$\frac{1}{\Upsilon_{(p)}^2} = \frac{1}{\Upsilon_{i,(p)}^2} + \alpha\phi \frac{x_{i,(p)}^2}{\sqrt{x_i^\top \Upsilon^2 x_i}} \,.$$

Thus,

$$\Upsilon_{(p)}^2 = \frac{\Upsilon_{i,(p)}^2 \sqrt{x_i^\top \Upsilon^2 x_i}}{\sqrt{x_i^\top \Upsilon^2 x_i} + \alpha\phi x_{i,(p)}^2 \Upsilon_{i,(p)}^2} \,.$$

Multiplying both sides by $x_{i,(p)}^2$ and summing over r we get

$$x_i^\top \Upsilon^2 x_i = \sum_r x_{i,(p)}^2 \Upsilon_{(p)}^2 = \sqrt{x_i^\top \Upsilon^2 x_i} \sum_r \frac{x_{i,(p)}^2 \Upsilon_{i,(p)}^2}{\sqrt{x_i^\top \Upsilon^2 x_i} + \alpha\phi x_{i,(p)}^2 \Upsilon_{i,(p)}^2} \,.$$

Finally, we obtain

$$\sqrt{x_i^\top \Upsilon^2 x_i} = \sum_r \frac{x_{i,(p)}^2 \Upsilon_{i,(p)}^2}{\sqrt{x_i^\top \Upsilon^2 x_i} + \alpha\phi x_{i,(p)}^2 \Upsilon_{i,(p)}^2} \,.$$

As before we employ the KKT conditions which state that when $\alpha > 0$ we have

$$m_i + \alpha v_i = \phi \sqrt{x_i^\top \Upsilon^2 x_i} \,.$$

Substituting in the last equality we get

$$\sqrt{x_i^\top \Upsilon^2 x_i} = \sum_r \frac{\phi x_{i,(p)}^2 \Upsilon_{i,(p)}^2}{m_i + \alpha v_i + \alpha\phi^2 x_{i,(p)}^2 \Upsilon_{i,(p)}^2} \,.$$

We use again the KKT conditions and get that the optimal value α_{i+1} is the solution of $g(\alpha) = 0$ for

$$g(\alpha) = m_i + \alpha v_i - \sum_r \frac{\phi^2 x_{i,(p)}^2 \Upsilon_{i,(p)}^2}{m_i + \alpha v_i + \alpha\phi^2 x_{i,(p)}^2 \Upsilon_{i,(p)}^2} \,. \tag{34}$$

The function $g(\alpha)$ defined in (34) and the function $f(\alpha)$ defined in (33) are both of the form

$$h(\alpha) = m_i + \alpha v_i - \sum_r \frac{a_r}{b + c_r \alpha} \,,$$

where $v_i, a_r, c_r \geq 0$. The only difference is that $b = 1 > 0$ in (33) and $b = m_i$ in (34). Nevertheless, the optimal value of α satisfies $h(\alpha_i) = 0$. The following lemma summarizes few properties of both functions:

Lemma 3 *Assume that $v_i > 0$ and let $L_i = \max\{0, -m_i/v_i\}$. Both* (33) *and* (34) *have the following properties:*

1. *Their value at L_i is non-positive, that is $f(L_i) \leq 0$, $g(L_i) \leq 0$.*
2. *They are strictly-increasing for $\alpha \geq L_i$*
3. *For each function there exists a value U_i such that their value at U_i is positive, $f(U_i) > 0$, $g(U_i) > 0$*

Proof For the first property we consider two cases $m_i \geq 0$ and $m_i < 0$. We start with the first case and thus $L_i = 0$. Thus, $f(0) = m_i - \sum_r \Sigma_{i,(p)} \phi x_{i,(p)}^2 = m_i - \phi v_i < 0$, where the last ineqluaity follows since we assume that the constraint of (10) does not hold. Also, $g(0) = m_i - \frac{1}{m_i} \sum_r \phi^2 x_{i,(p)}^2 \Upsilon_{i,(p)}^2 = m_i - \phi^2 \frac{v_i}{m_i} < 0$ since we assumed that the constraint of (19) does not hold. When $m_i < 0$ we have $L_i = -m_i/v_i > 0$. In this case (33) becomes,

$$f(L_i) = -\sum_r \frac{\Sigma_{i,(p)} \phi x_{i,(p)}^2}{1 + 2L_i \Sigma_{i,(p)} \phi x_{i,(p)}^2} \leq 0 \,,$$

since $\Sigma_{i,(p)} \phi x_{i,(p)}^2 \geq 0$. Similarly,

$$g(L_i) = -\sum_r \frac{\phi^2 x_{i,(p)}^2 \Upsilon_{i,(p)}^2}{L_i \phi^2 x_{i,(p)}^2 \Upsilon_{i,(p)}^2} = -\frac{d}{L_i} < 0 \,.$$

The second property of strictly-increasing follows immediately since $v_i > 0$ and since for both functions the denominator of each term in the sum over p is an increasing function in α which is non-negative in the range $\alpha \geq L_i$. Finally, the last property follows directly from the second property. ■

The lemma states that for each of f and g there is exactly one α_i (possibly different for each function) such that $f(\alpha_i) = 0$ and $g(\alpha_i) = 0$, but it does not provide an expression for computing α_i explicitly such as in Lemma 1. However, it further tells us that for each function the value of α_i is in the interval $[L_i, U_i]$. A value not far from α_i up to an accuracy of ε can be found using binary search in time proportional to $[(U_i - L_i) \log(1/\varepsilon)]$.

We conclude this section by computing a possible value U_i for each function and start with (33). Note that $a_i = \max\{0, -2m_i/v_i\}$ satisfies $m_i + (a_i/2)v_i \geq 0$. Thus, $b_i = \max_r \left\{ \left(2d\Sigma_{i,(p)} \phi x_{i,(p)}^2 \right) / v_i \right\}$ satisfies $b_i v_i / (2d) - \Sigma_{i,(p)} \phi x_{i,(p)}^2 / \left(1 + 2\alpha \Sigma_{i,(p)} \phi x_{i,(p)}^2 \right) \geq 0$ for $p = 1 \ldots d$. Therefore setting $U_i = \max\{a_i, b_i\}$ satisfies $f(U_i) \geq 0$ as desired. Finally, note that $U_i \geq L_i$ since $a_i \geq L_i$ by construction. For (34) we use the same definition of a_i but define $b_i = \max_r \left\{ \left(2d\Upsilon_{i,(p)}^2 \phi^2 x_{i,(p)}^2 \right) / v_i \right\}$ and $U_i = \max\{a_i, b_i\}$. By a similar argument we have $g(U_i) > 0$ and $L_i \leq U_i$.

To summarize, as opposed to the full covariance case, in the exact diagonal case we do not compute the value of α_i explicitly, but use a binary-search algorithm to efficiently find a good approximation for the optional solution.

7. Evaluation

In this section we evaluate diagonalized versions of the CW algorithm on a range of binary classification problems for NLP tasks. We compare our methods against each other and against competitive online and batch learning algorithms.

We selected a range of 5 tasks and created 17 binary classification problems. We begin with a description of each task.

7.1 20 Newsgroups

The 20 Newsgroups corpus contains approximately 20,000 newsgroup messages, partitioned across 20 different newsgroups.[8] The data set is a popular choice for binary and multi-class text classification as well as unsupervised clustering. Following common practice, we created binary problems from the data set by creating binary decision problems of choosing between two similar groups. Our groups are:

- comp: `comp.sys.ibm.pc.hardware` vs. `comp.sys.mac.hardware`
- sci: `sci.electronics` vs. `sci.med`
- talk `talk.politics.guns` vs. `talk.politics.mideast`

Each message was represented as a binary bag-of-words. For each problem we selected 1800 examples balanced between the two labels.

7.2 Reuters

The Reuters Corpus Volume 1 (RCV1-v2/LYRL2004) contains over 800,000 manually categorized newswire stories (Lewis et al., 2004). Each article contains one or more labels describing its general topic, industry and region. We created the following binary decision tasks from the labeled documents:

- Insurance: Life (I82002) vs. Non-Life (I82003)
- Business Services: Banking (I81000) vs. Financial (I83000)
- Retail Distribution: Specialist Stores (I65400) vs. Mixed Retail (I65600).

These distinctions involve neighboring categories so they are fairly hard to make. Details on document preparation and feature extraction are given by Lewis et al. (2004). For each problem we selected 2000 examples using a bag-of-words representation with binary features. Each problem contains a balanced mixture of examples from each label.

7.3 Sentiment

We used a larger version of the sentiment multi-domain data set of Blitzer et al. (2007) used in Dredze et al. (2010).[9] This data consists of product reviews from 7 Amazon domains (apparel, book, dvd, electronics, kitchen, music, video). The goal in each domain is to classify a product

8. Corpus can be found at `http://people.csail.mit.edu/jrennie/20Newsgroups/`.
9. Data set can be found at `http://www.cs.jhu.edu/~mdredze/datasets/sentiment/`.

review as either positive or negative. Feature extraction creates unigram and bigram features using counts following Blitzer et al. (2007). For the apparel domain we used all 1940 examples and for all other domains we used 2000 examples. Each problem contains a balanced mixture of example labels.

7.4 Spam

We include a spam classification problem as a sample problem from the space of email classification tasks. We chose spam since it is a widely studied problem with several publicly available data sets. We selected the 2006 ECML/PKDD Discovery Challenge spam data set (Bickel, 2006) and use the provided representations (bag-of-words). The goal is to classify an email (bag-of-words) as either spam or ham (not-spam). This corpus contains two data sets: task A, which has three users, and task B, which has 15 users. We use the three users from task A since it has more training examples. For each user we select 2000 examples.

7.5 Pascal

The PASCAL large scale learning challenge workshop provided several large scale binary data sets.[10] We selected the NLP task, which is a Webspam filtering problem. Each example is the text from a web page. The task is to classify a webpage as either spam or ham. We used the default format provided by the workshop and selected 2000 examples.

7.6 USPS

The USPS data set contains examples of all 10 digits as part of a digit recognition task (OCR) (Hull, 1994). We created binary tasks by pairing each digit with another in order: 0/9, 1/2, 3/4, 5/6, 7/8. We used the standard value of each pixel in the image, as well as the product of all the pixel pairs in the image (bi-grams.)

Each data set was randomly divided for 10-fold cross validation experiments. Classifier parameters (ϕ) and the number of training iterations (up to 10) were tuned for each classification task on a single randomized run over the data. Results are reported for each problem as the average accuracy over the 10 folds. Statistical significance is computed using McNemar's test.

7.7 Results

We start by comparing the performance of the diagonalized CW algorithms: `var` (linearization) against `stdev` (change of variables), approximate against exact diagonalization, and for approximate updates, KL against L_2. All six algorithm combinations were run on the data sets described above. The average test error on all data sets is shown in Table 2. For each method, we summarize its overall performance by computing its mean rank among all the other algorithm: if an algorithm has a mean rank of 1 then on average across all data sets it achieved the lowest error on average, whereas a rank of 6 indicates that it ranked 6th in error on average across all tests.

Starting with the KL methods for `var` and `stdev`, the `stdev` method does slightly better, a result shown in Crammer et al. (2008). Comparing the two methods for diagonalization (KL vs. L_2), while L_2 does slightly better for `var` (the best overall), the KL method appears to be more stable overall,

10. Data sets can be found at `http://largescale.first.fraunhofer.de/workshop/`.

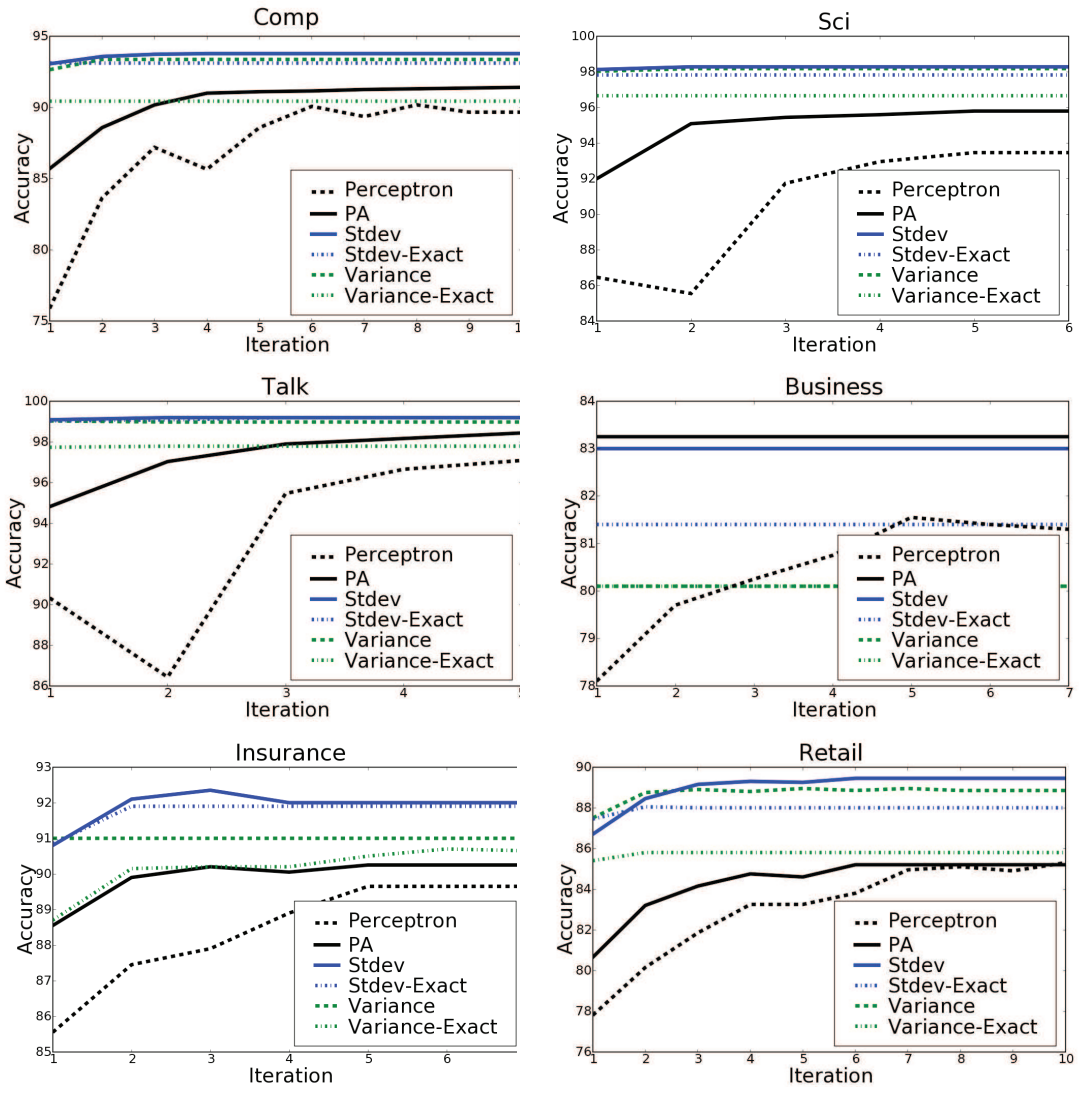


Figure 3: Accuracy on test data after each iteration on six data sets.

achieving the best or closest to best results for the `var` and `stdev` methods. In comparison, the exact methods do worse than the approximations. To understand these results, we examine some learning curves from several of the online experiments in Figure 7.6 and in Figure 7.6, which show accuracy on test data after each iteration. In many of these plots, the exact method does very well after the first iteration, surpassing the performance of the approximate methods. However, after the first iteration the exact method stops improving while the approximate methods continue to improve. By finding a solution that exactly achieves the constraint, exact produces a more aggressive algorithm that learns faster but overfits (Figure 5). In contrast, the approximate solutions do not fully enforce the constraint on each update but this slower learning reduces overfitting and improves generalization error over several iterations.

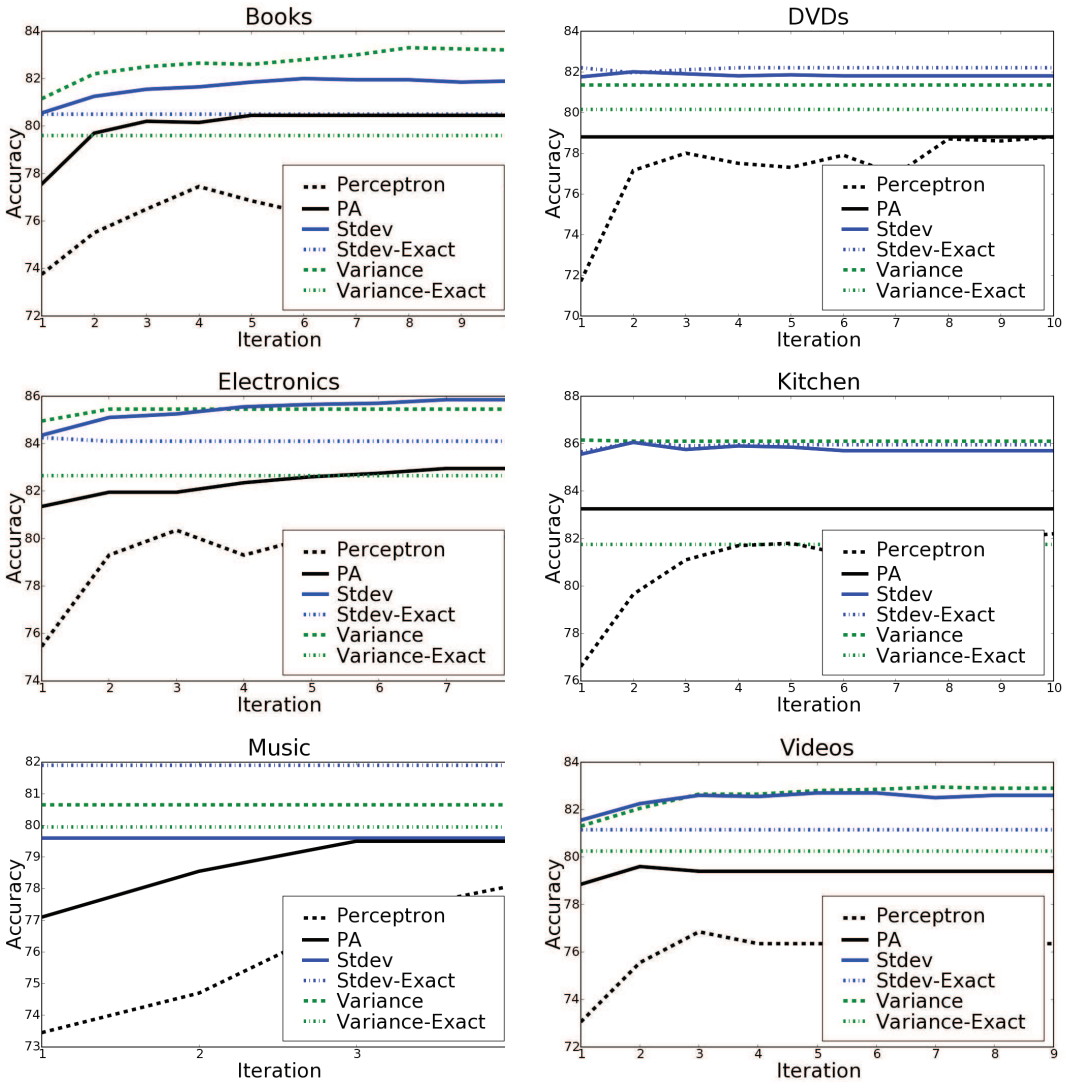


Figure 4: Accuracy on test data after each iteration on the six Amazon data sets.

We next compare the results from our approximation diagonalization CW methods to other popular online learning algorithms (Table 3). We evaluated the perceptron (Rosenblatt, 1958), passive-aggressive (Crammer et al., 2006a), stochastic gradient descent (Zhang, 2004; Blitzer et al., 2007) and a diagonalized second order perceptron (Cesa-Bianchi et al., 2005), all of which perform well for NLP problems. In every experiment, a CW method improved over all of the online learning baselines.

As discussed above, online algorithms are attractive even for batch learning because of their simplicity and ability to operate on extremely large data sets. In the batch setting, these algorithms are run several times over the training data, which yields slower performance than single pass learning (Carvalho and Cohen, 2006). While we have shown that CW improves on accuracy, it also learns faster than other baselines, requiring fewer iterations over the training data. Such behavior can be

		var			stdev		
	Task	*KL*	L_2	*Exact*	*KL*	L_2	*Exact*
Sentiment	Apparel	12.53	**12.47**	14.79	13.66	13.51	14.28
	Books	16.90	17.30	19.60	16.25	22.35	**15.25**
	DVD	17.45	17.05	19.05	17.60	18.30	**16.95**
	Electronics	14.95	15.40	16.65	**14.75**	20.95	15.50
	Kitchen	13.75	**13.65**	15.30	15.40	15.40	14.25
	Music	**17.15**	17.55	19.90	17.75	17.85	19.35
	Video	21.75	**18.55**	25.85	22.50	19.00	23.60
ECML	Spam A	2.65	1.45	3.10	**0.75**	4.15	0.80
	Spam B	1.35	1.20	2.65	**1.00**	1.10	1.05
	Spam C	1.50	1.40	3.40	1.50	3.55	**1.35**
Reuters	Retail	10.55	18.80	18.75	**10.25**	11.05	11.05
	Business	16.35	**15.35**	17.10	16.45	16.80	17.20
	Insurance	**8.20**	9.15	10.20	8.55	9.55	10.10
20 News	Comp	6.69	**5.61**	8.59	6.79	16.64	6.90
	Sci	**2.44**	2.74	3.20	3.04	13.35	3.10
	Talk	0.86	0.43	2.43	**0.27**	8.38	1.14
Pascal	Webspam	3.55	3.10	3.85	**2.95**	3.10	5.35
	Mean Rank	2.53	*2.35*	5.29	*2.47*	4.53	3.59

Table 2: Average Error of all variants of confidence-weighted algorithms presented in this paper over 17 binary text classification tasks. The best score for each data set is set in bold. The mean rank is the average rank of each algorithm across data sets, ranging from 1 (best) to 6.

seen in Figure 7.6 and Figure 7.6, which shows test error after each training iteration for CW and PA. While CW clearly improves over PA, it converges very quickly, reaching near best performance on the first iteration. In contrast, PA benefits from multiple iterations over the data; its performance changes significantly from the first to fifth iteration. The plot also illustrates exact's behavior, which initially beats PA but does not improve. In fact, on eleven of the twelve data sets, `var`-Exact beats PA on the first iteration.

7.8 Batch Learning

While online algorithms are widely used, batch algorithms are still preferred for many tasks. Batch algorithms can make global learning decisions by examining the entire data set, an ability beyond online algorithms. In general, when batch algorithms can be applied they perform better. We compare CW to three standard batch algorithms: naïve Bayes (default configuration in MALLET McCallum, 2002), maximum entropy classification (default configuration in MALLET McCallum, 2002) and support vector machines (LibSVM Chang and Lin, 2001). Classifier parameters (Gaussian prior for maxent and C for SVM) were tuned as for the online methods.

Results for batch learning are shown in table Table 4. As expected, the batch methods tend to do better than the online methods (perceptron, PA, and SGD). However, in 13 out of 17 tasks the CW

		var		stdev					
	Task	*KL*	L_2	*KL*	L_2	*Per*	*PA*	*SOP*	*SGD*
Sentiment	Apparel	12.53	**12.47**	13.66	13.51	17.84	13.35	17.42	13.76
	Books	16.90	17.30	$*$**16.25**	22.35	23.10	18.45	19.75	18.60
	DVD	17.45	$*$**17.05**	$*$17.60	18.30	20.45	20.95	20.55	18.70
	Electronics	14.95	15.40	$\star$**14.75**	$\star$20.95	18.65	17.45	20.20	16.00
	Kitchen	$\star$13.75	$*$**13.65**	15.40	15.40	16.65	15.20	21.20	16.00
	Music	**17.15**	17.55	17.75	17.85	22.35	19.40	21.80	18.20
	Video	21.75	**18.55**	22.50	19.00	21.50	18.90	21.90	19.25
ECML	Spam A	2.65	1.45	$\star$**0.75**	4.15	3.20	1.40	4.20	2.30
	Spam B	1.35	$\star$1.20	$*$**1.00**	1.10	3.00	2.40	1.95	2.80
	Spam C	1.50	$\star$**1.40**	$\star$1.50	3.55	3.65	2.10	2.90	2.15
Reuters	Retail	†10.55	18.80	†**10.25**	†11.05	19.60	17.50	19.45	14.30
	Business	16.35	**15.35**	16.45	16.80	19.00	16.15	21.80	15.45
	Insurance	**8.20**	9.15	8.55	9.55	11.35	12.35	10.15	9.35
20 News	Comp	6.69	†**5.61**	6.79	†16.64	10.30	8.65	10.45	7.88
	Sci	†**2.44**	†2.74	$*$3.04	†13.35	6.70	8.06	4.67	4.06
	Talk	0.86	$*$0.43	†**0.27**	†8.38	3.24	1.57	2.59	1.19
Pascal	Webspam	3.55	3.10	$\star$**2.95**	3.10	7.60	3.90	5.05	3.50
USPS	0 vs 9	0.56	0.56	0.56	0.56	**0.37**	0.93	0.75	0.56
	1 vs 2	1.73	0.87	0.87	4.33	1.73	**0.65**	2.38	42.86
	3 vs 4	1.37	1.09	1.09	1.09	**0.82**	1.09	2.73	45.36
	5 vs 6	**0.91**	0.91	1.52	1.52	4.24	0.91	3.03	48.48
	7 vs 8	2.24	2.24	1.92	2.24	2.56	**1.60**	4.15	53.04

Table 3: Average Error of approximate-diagonal confidence-weighted algorithms and four other online algorithms: The perceptron algorithm (Per), the passive-aggressive (PA) algorithm, the second order perceptron (SOP) and stochastic gradient decent evaluated using 17 binary text classification tasks. The best score for each data set is set in bold. Statistical significance measured by McNemar's test indicates when a CW algorithm is statistically significant ($\star$ $p = 0.05$, $*$ $p = 0.01$, † $p = 0.001$) from each of the four baselines (perceptron, PA, SOP, SGD).

algorithm beats all of the batch methods. The much faster and simpler online algorithm performs better than the slower more complex batch methods.

The speed advantage of online methods in the batch setting can be seen in Table 5, which shows the average training time in seconds for a single experiment (fold) for a representative selection of CW algorithms and some of the baselines. The online times include the multiple iterations selected for each online learning experiment. The differences between the online and batch algorithms are striking. While CW performs better than the batch methods, it is also much faster, while being equivalent in speed to the other online methods. For webspam data, which contains many features, an SVM takes over 1.5 minutes to train while the CW algorithms take between 1-2 seconds.

We also evaluated the effects of commonly used techniques for online and batch learning, including averaging and TFIDF features; they did not improve results so details are omitted. Although

		var		stdev				
	Task	*KL*	L_2	*KL*	L_2	*NB*	*Maxent*	*SVM*
Sentiment	Apparel	12.53	**12.47**	13.66	13.51	12.63	13.56	13.92
	Books	16.90	17.30	⋆**16.25**	†22.35	18.40	18.05	18.25
	DVD	17.45	**17.05**	17.60	18.30	21.00	18.00	19.60
	Electronics	14.95	15.40	⋆**14.75**	†20.95	17.05	15.85	16.25
	Kitchen	⋆13.75	**13.65**	15.40	15.40	15.00	15.25	15.50
	Music	**17.15**	17.55	17.75	17.85	18.65	17.90	18.25
	Video	21.75	18.55	22.50	19.00	22.95	**18.40**	18.80
ECML	Spam A	⋆2.65	1.45	⋆**0.75**	4.15	3.70	1.30	1.75
	Spam B	1.35	1.20	⋆**1.00**	1.10	4.20	1.55	1.90
	Spam C	1.50	1.40	1.50	†3.55	1.40	**1.35**	1.40
Reuters	Retail	∗10.55	⋆18.80	†**10.25**	∗11.05	16.55	12.55	12.90
	Business	16.35	**15.35**	16.45	16.80	20.00	15.85	15.60
	Insurance	**8.20**	9.15	8.55	9.55	11.80	9.10	9.75
20 News	Comp	6.69	5.61	6.79	†16.64	**5.56**	7.82	7.67
	Sci	⋆2.44	⋆2.74	3.04	†13.35	**1.42**	3.40	3.86
	Talk	0.86	⋆0.43	+**0.27**	†8.38	0.97	1.03	1.24
Pascal	Webspam	3.55	⋆3.10	+**2.95**	⋆3.10	19.10	6.05	3.85
USPS	0 vs 9	**0.56**	0.56	0.56	0.56	1.12	33.02	0.56
	1 vs 2	1.73	0.87	0.87	4.33	1.52	42.86	**0.65**
	3 vs 4	1.37	1.09	1.09	1.09	1.91	45.36	**0.55**
	5 vs 6	0.91	0.91	1.52	1.52	3.03	48.48	**0.61**
	7 vs 8	2.24	2.24	1.92	2.24	2.56	53.04	**0.96**

Table 4: Average Error of approximate-diagonal confidence-weighted algorithms and three batch algorithms: Naïve Bayes (NB), Maximum entropy classifier (Maxent) and support vector machine (SVM) evaluated using 17 binary text classification tasks. The best score for each data set is set in bold. Statistical significance measured by McNemar's test indicates when a CW algorithm is statistically significant ($\star$ $p = 0.05$, $*$ $p = 0.01$, † $p = 0.001$) from each of the three baselines (NB, Maxent, SVM).

the above data sets are balanced with respect to labels, we also evaluated the methods on variant data sets with unbalanced label distributions, and still saw similar benefits from the CW methods.

7.9 Large Data Sets

Online algorithms are especially attractive in tasks where training data exceeds available main memory or in streaming settings where training examples cannot be saved. In both of these settings, a single sequential pass over the data is highly preferred to multiple passes common in batch training cases. So far, we have shown that CW algorithms are more aggressive than other online algorithms, an advantage when the algorithm is limited to a single pass. The results is both higher performance and fewer training iterations. The question we now answer is whether this advantage is maintained in large data settings.

Task	*variance KL*	*variance Exact*	*Perceptron*	*PA*	*Maxent*	*SVM*
Apparel	0.2	0.2	0.1	0.04	1	11
Books	0.1	0.8	0.1	0.1	6	25
DVD	0.1	0.9	0.1	0.04	6	22
Electronics	0.1	0.4	0.1	0.03	2	17
Kitchen	0.1	0.6	0.1	0.1	2	14
Music	0.1	0.5	0.1	0.1	4	19
Video	0.3	0.7	0.1	0.2	6	24
Spam A	0.03	0.2	0.1	0.8	3	3
Spam B	0.04	0.2	0.1	0.1	3	3
Spam C	0.1	0.2	0.04	0.04	1	2
Retail	0.1	0.1	0.03	0.03	0.6	4
Business	0.1	0.3	0.1	0.02	0.3	5
Insurance	0.1	0.2	0.1	0.02	0.8	4
Comp	0.04	0.2	0.1	0.2	2	11
Sci	0.1	0.3	0.1	0.04	2	7
Talk	0.1	0.3	0.1	0.1	4	7
Webspam	1	2	3	1	12	103

Table 5: Training times in seconds for a single training run (averaged over 10 trials.)

We selected two large data sets for evaluation. The combined product reviews for all the domains by Blitzer et al. (2007) yield one million sentiment examples. While most reviews were from the book domain, the reviews are taken from a wide range of Amazon product types and are mostly positive. From the Reuters corpus, we created a one vs. all classification task for the *Corporate* topic label, yielding 804,411 examples of which 381,325 are labeled corporate. For the two data sets, we created four random splits each with 10,000 test examples and the remaining examples saved for training. Parameters were optimized by training on 5,000 randomly chosen examples. We evaluated the CW `var`-KL algorithm and the passive-aggressive algorithm using a single pass over this data.

The results are shown as horizontal lines in Figure 6. For the Sentiment data, CW maintains over a 1% lead when compared to PA. On the Reuters data, the results are reversed with PA having the advantage. The difference between these behaviors may be related to the different feature representations used by each data set. The Reuters data contains $288,062$ unique features, for a feature to document ratio of 0.36. In contrast, the sentiment data contains $13,460,254$ unique features, a feature to document ratio of 13.33. This means that Reuters features will occur several times during training while many sentiment features only once. This may give CW an advantage on Sentiment. It is also possible that CW over-fits the Reuters data, something that will be observed in the next set of experiments below.

7.10 Distributed Training

While faster learning over a data stream is important, not all large data sets can be processed by a single processor. Therefore, we looked at the case where many processors are available, each with easy access to a fraction of the training data, but where communication between processors

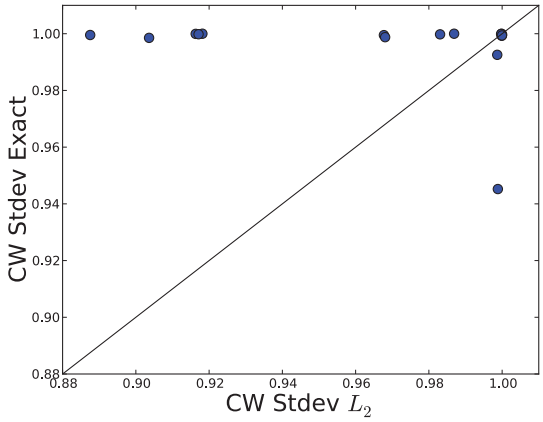


Figure 5: The trained model's accuracy on the training data for fifteen of the data sets for the exact diagonal and L_2 diagonal approximation Stdev methods. Points above the line indicate that the exact algorithm obtained a higher training accuracy than the L_2 diagonal method. Observe that the exact method almost always obtains a higher training accuracy, and is nearly 100% in every case. Coupled with the results on test data, which are worse for the exact methods, these results indicate that the exact method overfits the training data. These results are typical when comparing the exact algorithms against the diagonal approximations.

is limited. In this setting, we would like an algorithm where individual processors train models on their easily accessible data, and then they combine their models. While this often does not perform as well as a single model trained on all of the data, it is a cost-effective way of learning from very large training sets.

One simple approach is to combine many trained models by averaging their weights (McDonald et al., 2010). However, averaging models trained in parallel assumes that each model has an equally accurate estimate of the model weights. This is obviously not the case where different processors saw different portions of the data, made different updates, or saw features that other processors did not. Rather than taking an average over all models, CW provides a confidence value for each weight, allowing for a more intelligent combination of weights from multiple models.

Since each model is a Gaussian distribution over weights, combining multiple trained CW classifiers is equivalent to combining multiple Gaussian distributions. Specifically, we compute the combined model by finding the Gaussian that minimizes the total divergence to the set C of Gaussian distributions (individually trained classifiers) for some divergence operator D:

$$\min_{\mu,\Sigma} \sum_{c \in C} D((\mu,\Sigma)||(\mu_c,\Sigma_c)),$$

If D is the Euclidean distance, then this is just the average of the individual models. However, we can instead rely on the variance estimates of each Gaussian by choosing the KL divergence for D.

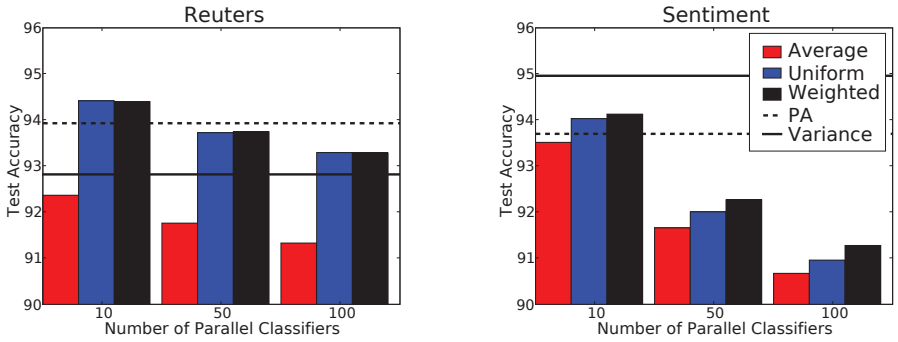


Figure 6: Results for Reuters (800k) and Sentiment (1000k) averaged over 4 runs. Horizontal lines show the test accuracy of a model trained on the entire training set. Vertical bars show the performance of n (10, 50, 100) classifiers trained on disjoint sections of the data as the average performance, uniform combination, or weighted combination.

This minimization leads to the following weighted combination of individual model means:

$$\mu = \left(\sum_{c \in C} \Sigma_c^{-1} \right)^{-1} \sum_{c \in C} \Sigma_c^{-1} \mu_c \qquad \Sigma^{-1} = \sum_{c \in C} \Sigma_c^{-1}.$$

We evaluate classifier combination by training n (10, 50, 100) models by dividing the example stream into n disjoint parts and report the average performance of each of the n classifiers (average), the combined classifier from taking the average of the n sets of weights (L_2) and the combination using the KL divergence on the test data across 4 randomized runs.

Average accuracy on the test sets are reported in Figure 6. As stated above, the PA single model achieves higher accuracy for Reuters, possibly because of the low feature to document ratio. However, combining 10 CW classifiers achieves the best performance. For sentiment, combining 10 classifiers beats PA but is not as good as a single CW model. In every case, combining the classifiers improves over each model individually. On sentiment, the KL combination improves over the L_2 combination and in Reuters the models are equivalent. For comparison, we show the accuracy on the test data for a single run on the CW Variance KL model on sentiment data Figure 7. When trained on all of the data and distributed across 10 machines, the classifier loses 1% of its performance which, using Figure 7 as a guide, corresponds to using 22% of the training data.

Finally, we computed the actual run time of both PA and CW on the large data sets to compare the speed of each model. While CW is more complex, requiring more computation per example, the actual speed is comparable to PA; in all tests the run time of the two algorithms was indistinguishable.

8. Related Work

The idea of using weight-specific variable learning rates has a long history in neural-network learning (Sutton, 1992), although we do not know of a previous model that specifically models confidence in a way that takes into account the frequency of features.

Figure 7: Results from CW Variance KL run on the large scale Sentiment data (1000k) averaged over 4 runs. Accuracy on test data is measured every 10k training examples to demonstrate the improvement with increases in training data.

Online additive algorithms have a long history, from the perceptron (Rosenblatt, 1958) to more recent methods (Kivinen and Warmuth, 1997; Crammer et al., 2006b). Our update has a more general form, in which the input vector x_i is linearly transformed using the covariance matrix, both rotating the input and assigning weight specific learning rates.

The second order perceptron (SOP) (Cesa-Bianchi et al., 2005) demonstrated that second-order techniques can improve first-order online methods. Both SOP and CW maintain second-order information. SOP is mistake driven while CW is passive-aggressive. SOP uses the current example in the correlation matrix for prediction while CW updates after prediction. A variant of `stdev` similar to SOP follows from our derivation if we fix the Lagrange multiplier in (20) to a predefined value $\alpha_i = \alpha$, omit the square root, and use a gradient-descent optimization step. Fundamentally, CW algorithms have a probabilistic motivation, while the SOP is geometric: replace the ball around an example with a refined ellipsoid. Shivaswamy and Jebara (2007) used a similar motivation in batch learning.

Ensemble learning shares the idea of combining multiple classifiers. Gaussian process classification (GPC) maintains a Gaussian distribution over weight vectors (primal) or over regressor values (dual). Our algorithm uses a different update criterion than the standard GPC Bayesian updates (Rasmussen and Williams, 2006, Chapter 3), avoiding the challenge of approximating posteriors. Bayes point machines (Herbrich et al., 2001) maintain a collection of weight vectors consistent with the training data, and use the single linear classifier which best represents the collection. Conceptually, the collection is a non-parametric distribution over the weight vectors. Its online version (Harrington et al., 2003) maintains a set of weight vectors that are updated simultaneously. The relevance vector machine (Tipping, 2001) incorporates probability into the dual formulation of SVMs. As in our work, the dual parameters are random variables distributed according to a diagonal Gaussian with example specific variance. The weighted-majority (Littlestone and Warmuth, 1994) algorithm and later improvements (Cesa-Bianchi et al., 1997) combine the output of multiple arbitrary

classifiers, maintaining a multinomial distribution over the experts. We assume linear classifiers as experts and maintain a Gaussian distribution over their weight vectors.

With the growth of available data there is an increasing need for algorithms that process training data very efficiently. A similar approach to ours is to train classifiers incrementally (Bordes and Bottou, 2005). The extreme case is to use each example once, without repetitions, as in the multiplicative update method of Carvalho and Cohen (2006).

In Bayesian modeling, we note few approaches that use parameterized distributions over weight vectors. Borrowing concepts from support vector machines, Jaakkola et al. (1999) developed maximum entropy discrimination, which models the generation of examples with one generative model for each class. The model consisted of distributions over the weights and over margin thresholds. They used Bayesian prediction and set the weights using the maximum-entropy principle. In a more recent approach, Minka et al. (2009) proposed using additional virtual vectors to allow more expressive power beyond Gaussian prior and posterior.

Passing the output of a linear model through a logistic function has a long-history in the statistical literature, and is extensively covered in many textbooks (e.g., Hastie et al., 2001). Platt (1998) used similar ideas to convert the output of a support vector machine into probabilistic quantities.

Since the conference versions of this work were published, a few algorithms reminiscent of CW were proposed. Duchi et al. (2010) and McMahan and Streeter (2010) proposed to replace the standard Euclidean distance in stochastic gradient decent with general Mahalanobis distance defined by the second order information, captured by the instantaneous second order moment. Crammer et al. (2009a) proposed to replace the hard constraint enforced by the CW algorithm with a relaxed version, formulated using an additional term in the objective function. They call their algorithm AROW for adaptive regularization of weight vectors. Orabona and Crammer (2010) proposed later a framework for online learning, which contains an algorithm close to AROW as a special case, as well as other new algorithms. From a different perspective, Crammer and Lee (2010) proposed a microscopic view for learning, that tracks individual weight-vectors as opposed only to their macroscopic quantities, such as mean and covariance. Their algorithm has similar update form as CW ((11) and (13)), yet with different rates.

Finally, Shivaswamy and Jebara (2010b,a) proposed to use second order information, or the variance in the batch setting where an iid distribution over the examples is assumed. Their algorithm both maximizes the (average) margin and at the same time minimizes its variance. Note, that they do not maintain a distribution over weight vectors, and the probability space is induced using the distribution over training examples.

9. Conclusion

We have presented confidence-weighted linear classifiers, a new learning method designed for NLP problems based on the notion of weight confidence. The algorithm maintains a distribution over weight vectors; online updates both improve the weight estimates and reduce the distribution's variance. Our method improves over both online and batch methods and learns faster on over a dozen NLP data sets. Additionally, our new algorithms allow more intelligent classifier combination techniques, yielding improved performance in distributed learning.

Acknowledgments

Most of the work was performed while the authors were in the Department of Computer and Information Science at the University of Pennsylvania. This research was supported in part by the Israeli Science Foundation grant ISF-1567/10. Koby Crammer is a Horev Fellow, supported by the Taub Foundations.

References

Galen Andrew and Jianfeng Gao. Scalable training of l1-regularized log-linear models. In *ICML '07: Proceedings Of The 24th International Conference On Machine Learning*, pages 33–40, New York, NY, USA, 2007. ACM. ISBN 978-1-59593-793-3. doi: http://doi.acm.org/10.1145/1273496.1273501.

Steffen Bickel. ECML-PKDD discovery challenge overview. In *The ECML-PKDD Discovery Challenge Workshop*, 2006.

John Blitzer, Mark Dredze, and Fernando Pereira. Biographies, Bollywood, boom-boxes and blenders: Domain adaptation for sentiment classification. In *Association For Computational Linguistics (ACL)*, 2007.

Antoine Bordes and Léon Bottou. The Huller: a simple and efficient online SVM. In *European Conference On Machine Learning(ECML), LNAI 3720*, 2005.

Stephen Boyd and Lieven Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004.

Xavier Carreras, Michael Collins, and Terry Koo. Tag, dynamic programming, and the perceptron for efficient, feature-rich parsing. In *Conference On Natural Language Learning (CONLL)*, 2008.

Vitor R. Carvalho and William W. Cohen. Single-pass online learning: Performance, voting schemes and online feature selection. In *KDD-2006*, 2006.

Nicoló Cesa-Bianchi and Gabor Lugosi. *Prediction, learning, and games*. Cambridge University Press, 2006.

Nicoló Cesa-Bianchi, Yoav Freund, David Haussler, David P. Helmbold, Robert E. Schapire, and Manfred K. Warmuth. How to use expert advice. *Journal of the ACM*, 44(3):427–485, May 1997.

Nicoló Cesa-Bianchi, Alex Conconi, and Claudio Gentile. A second-order perceptron algorithm. *Siam Journal Of Commutation*, 34(3):640–668, 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`.

David Chiang, Yuval Marton, and Philip Resnik. Online large-margin training of syntactic and structural translation features. In *Empirical Methods In Natural Language Processing (EMNLP)*, 2008.

Michael Collins. Discriminative training methods for hidden markov models: Theory and experiments with perceptron algorithms. In *Empirical Methods In Natural Language Processing (EMNLP)*, 2002.

Koby Crammer and Daniel D. Lee. Learning via Gaussian herding. In *Advances In Neural Information Processing Systems 24*, 2010.

Koby Crammer, Ofer Dekel, Joseph Keshet, Shai Shalev-Shwartz, and Yoram Singer. Online passive-aggressive algorithms. *Journal Of Machine Learning Research*, 7:551–585, 2006a.

Koby Crammer, Ofer Dekel, Joseph Keshet, Shai Shalev-Shwartz, and Yoram Singer. Online passive-aggressive algorithms. *Journal Of Machine Learning Research*, 7:551–585, 2006b.

Koby Crammer, Mark Dredze, and Fernando Pereira. Exact convex confidence-weighted learning. In *Neural Information Processing Systems (NIPS)*, 2008.

Koby Crammer, Alex Kulesza, and Mark Dredze. Adaptive regularization of weight vectors. In *Advances In Neural Information Processing Systems 23*, pages 414–422, 2009a.

Koby Crammer, Mehryar Mohri, and Fernando Pereira. Gaussian margin machines. In *Proceedings Of The Twelfth Intentional Conference On Artificial Intelligence And Statistics (AISTATS)*, 2009b.

Mark Dredze and Koby Crammer. Active learning with confidence. In *Association For Computational Linguistics (ACL)*, 2008a.

Mark Dredze and Koby Crammer. Online methods for multi-domain learning and adaptation. In *Empirical Methods In Natural Language Processing (EMNLP)*, 2008b.

Mark Dredze, Koby Crammer, and Fernando Pereira. Confidence-weighted linear classification. In *International Conference On Machine Learning (ICML)*, 2008.

Mark Dredze, Alex Kulesza, and Koby Crammer. Multi-domain learning by confidence-weighted parameter combination. *Machine Learning*, 79(1-2):123–149, 2010.

John Duchi, Elad Hazan, and Yoram Singer. Adaptive subgradient methods for online learning and stochastic optimization. In *Proceedings Of The Twenty Third Annual Conference On Learning Theory*, 2010.

Jianfeng Gao, Galen Andrew, Mark Johnson, and Kristina Toutanova. A comparative study of parameter estimation methods for statistical natural language processing. In *Proceedings Of The 45th Annual Meeting Of The Association Of Computational Linguistics*, pages 824–831, Prague, Czech Republic, June 2007. Association for Computational Linguistics. URL `http://www.aclweb.org/anthology/P07-1104`.

Amir Globerson, Terry Y. Koo, Xavier Carreras, and Michael Collins. Exponentiated gradient algorithms for log-linear structured prediction. In *Proceedings Of The 24th International Conference On Machine Learning*, pages 305–312. ACM New York, NY, USA, 2007.

Edward Harrington, Ralf Herbrich, Jyrki Kivinen, John Platt, and Robert C. Williamson. Online Bayes point machines. In *7th Pacific-Asia Conference On Knowledge Discovery And Data Mining (PAKDD)*, 2003.

Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer, 2001.

Ralf Herbrich, Thore Graepel, and Colin Campbell. Bayes point machines. *Journal Of Machine Learning Research*, 1:245–279, 2001.

Jonathan J. Hull. A database for handwritten text recognition research. *Pattern Analysis And Machine Intelligence, IEEE Transactions On*, 16(5):550–554, 1994.

Tommi Jaakkola, Marina Meila, and Tony Jebara. Maximum entropy discrimination, 1999.

Jyrki Kivinen and Manfred K. Warmuth. Exponentiated gradient versus gradient descent for linear predictors. *Information And Computation*, 132(1):1–64, January 1997.

David D. Lewis, Yiming Yang, Tony G. Rose, and Fan Li. Rcv1: A new benchmark collection for text categorization research. *Journal Of Machine Learning Research (JMLR)*, 5:361–397, 2004.

Nick Littlestone. *Mistake bounds and logarithmic linear-threshold learning algorithms*. PhD thesis, U. C. Santa Cruz, March 1989.

Nick Littlestone and Manfred K. Warmuth. The weighted majority algorithm. *Information and Computation*, 108:212–261, 1994.

Justin Ma, Lawrence K. Saul, Stefan Savage, and Geoffrey M. Voelker. Identifying suspicious URLs: An application of large-scale online learning. In *Proc. Of The International Conference On Machine Learning (ICML)*, 2009.

Justin Ma, Alex Kulesza, Koby Crammer, Mark Dredze, Lawrence Saul, and Fernando Pereira. Exploiting feature covariance in high-dimensional online learning. In *AIStats*, 2010.

Andrew McCallum. MALLET: A machine learning for language toolkit. `http://mallet.cs.umass.edu`, 2002.

Ryan McDonald, Koby Crammer, and Fernando Pereira. Large margin online learning algorithms for scalable structured classification. In *NIPS Workshop On Structured Outputs*, 2004.

Ryan McDonald, Koby Crammer, and Fernando Pereira. Flexible text segmentation with structured multilabel classification. In *Empirical Methods In Natural Language Processing (EMNLP)*, 2005a.

Ryan McDonald, Koby Crammer, and Fernando Pereira. Online large-margin training of dependency parsers. In *Association For Computational Linguistics (ACL)*, 2005b.

Ryan McDonald, Keith Hall, and Gideon Mann. Distributed training strategies for the structured perceptron. In *North American Chapter Of The Association For Computational Linguistics (NAACL)*, 2010.

H. Brendan McMahan and Matthew Streeter. Adaptive bound optimization for online convex optimization. In *Proceedings Of The Twenty Third Annual Conference On Learning Theory*, 2010.

Thomas P. Minka, Rongjing Xiang, and Yuan Alan Qi. Virtual vector machine for Bayesian online classification. In *Proceedings Of The Twenty Fifth Conference On Uncertainty In Artificial Intelligence*, 2009.

Francesco Orabona and Koby Crammer. New adaptive algorithms for online classification. In *Advances In Neural Information Processing Systems 24*, 2010.

Kaare B. Petersen and Michael S. Pedersen. The matrix cookbook, oct 2008. URL `http://www2.imm.dtu.dk/pubdb/p.php?3274`. Version 20081110.

John C. Platt. Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. In P.J. Bartlett, B. Schölkopf, D. Schuurmans, and A. J Smola, editors, *Advances In Large Margin Classifiers*. MIT Press, 1998.

Carl E. Rasmussen and Christopher K. I. Williams. *Gaussian Processes for Machine Learning*. The MIT Press, 2006.

Frank Rosenblatt. The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological Review*, 65:386–407, 1958. (Reprinted in *Neurocomputing* (MIT Press, 1988).).

Libin Shen, Giorgio Satta, and Aravind K. Joshi. Guided learning for bidirectional sequence classification. In *Association For Computational Linguistics (ACL)*, 2007.

Pannaga Shivaswamy and Tony Jebara. Ellipsoidal kernel machines. In *Artificial Intelligence And Statistics (AISTATS)*, 2007.

Pannaga Shivaswamy and Tony Jebara. Empirical Bernstein boosting. In Y.W. Teh and M. Titterington, editors, *Proceedings Of The Thirteenth International Conference On Artificial Intelligence And Statistics (AISTATS) 2010*, volume Volume 9 of JMLR: W&CP, pages 733–740, May 13-15 2010a.

Pannagadatta K. Shivaswamy and Tony Jebara. Maximum relative margin and data-dependent regularization. *Journal Of Machine Learning Research*, 11:747–788, 2010b.

Richard S. Sutton. Adapting bias by gradient descent: an incremental version of delta-bar-delta. In *Proceedings Of The Tenth National Conference On Artificial Intelligence*, pages 171–176. MIT Press, 1992.

Michael E. Tipping. Sparse Bayesian learning and the relevance vector machine. *Journal Of Machine Learning Research*, 1:211–244, 2001.

Simon Tong and Daphne Koller. Support vector machine active learning with applications to text classification. *Journal Of Machine Learning Research (JMLR)*, 2001.

Tong Zhang. Solving large scale linear prediction problems using stochastic gradient descent algorithms. In *International Conference On Machine Learning (ICML)*, 2004.

Journal of Machine Learning Research 13 (2012) 1927-1966 Submitted 1/12; Revised 3/12; Published 6/12

Integrating a Partial Model into Model Free Reinforcement Learning

Aviv Tamar AVIVT@TX.TECHNION.AC.IL
Dotan Di Castro DOT@TX.TECHNION.AC.IL
Ron Meir RMEIR@EE.TECHNION.AC.IL
Department of Electrical Engineering
Technion
Haifa 32000, Israel

Editor: Peter Dayan

Abstract

In reinforcement learning an agent uses online feedback from the environment in order to adaptively select an effective policy. Model free approaches address this task by directly mapping environmental states to actions, while model based methods attempt to construct a model of the environment, followed by a selection of optimal actions based on that model. Given the complementary advantages of both approaches, we suggest a novel procedure which augments a model free algorithm with a partial model. The resulting *hybrid* algorithm switches between a model based and a model free mode, depending on the current state and the agent's knowledge. Our method relies on a novel definition for a partially known model, and an estimator that incorporates such knowledge in order to reduce uncertainty in stochastic approximation iterations. We prove that such an approach leads to improved policy evaluation whenever environmental knowledge is available, without compromising performance when such knowledge is absent. Numerical simulations demonstrate the effectiveness of the approach on policy gradient and Q-learning algorithms, and its usefulness in solving a call admission control problem.

Keywords: reinforcement learning, temporal difference, stochastic approximation, markov decision processes, hybrid model based model free algorithms

1. Introduction

In Reinforcement Learning (RL) an agent attempts to improve its performance over time at a given task, based on continual interaction with the (usually unknown) environment, (Bertsekas and Tsitsiklis, 1996; Sutton and Barto, 1998). This improvement takes place by modifying the action selection policy, based on feedback from the environment and prior knowledge available to the agent. Formally, RL is often phrased as the problem of finding a mapping, the so called *policy*, from the environment's states to the agent's actions that maximizes a given functional of a reward function.

Most RL algorithms can be classified into either *model based* (also termed indirect) or *model free* (direct) approaches (Sutton and Barto, 1998; Bertsekas and Tsitsiklis, 1996). In the former setting, taking its inspiration from the field of Adaptive Control (Kumar, 1985), the agent maintains an explicit model of the environmental dynamics, typically in the form of a *Markov Decision Process* (MDP), while interacting with it. Based on this model, a *planning* problem is solved where techniques from *Dynamic Programming* (Bertsekas, 2006) are applied in order to find the optimal policy function. On the other hand, within the model free setting, the agent does not try to build a model of the MDP, but rather attempts to find the optimal policy by directly mapping environmental

states to actions. In this sense, no model of the environmental dynamics is required. While it can be shown that both approaches, under mild conditions, asymptotically reach the same optimal policy on typical MDP's, it is known that each approach possesses distinct merits. Model based methods often make better use of a limited amount of experience and thus achieve a better policy with fewer environmental interactions. On the other hand, model free methods are simpler, require less computational resources, and are not affected by biases in the design (or estimation) of the model.

The view taken in this work is that this dichotomy between algorithmic approaches, although popular, is not necessarily desirable. As an example, consider a scenario where some parts of the environment are known in advance, but computational resources are limited, restricting the use of proper model based approaches. In this case, a *hybrid approach* may allow us to benefit from using parts of the model in the algorithm, without sacrificing its simplicity, thus striking a balance between the merits of each approach. Surprisingly, the concept of combining model free and model based algorithms has received very little attention in the RL literature, and theoretical guarantees to its advantages are lacking.

In this work we pursue such a hybrid approach applicable to cases where partial model information is available in a specific form which we term *partially known MDP*. We provide a method for integrating such information into RL algorithms of the Stochastic Approximation (SA) type (Kushner and Yin, 2003; Borkar, 2008). This class of online model free algorithms includes many standard RL approaches that have been used effectively in practice (e.g., Tesauro, 1995; Crites and Barto, 1996). The method we propose reduces uncertainty in the algorithm trajectory, thereby improving its performance. Our theoretical analysis focuses on a particular model free algorithm - the well known TD(0) policy evaluation algorithm, and we prove that our hybrid method leads to improved performance, as long as sufficiently accurate partial knowledge is available. The effectiveness and generality of our method is further demonstrated in two numerical simulations. In the first, we apply it to a policy gradient type algorithm, and investigate its performance in randomly generated MDPs. In the second, we consider a call admission control problem. As it turns out, our partially known MDP definition is a natural choice for describing plausible partial knowledge in such problems, and performance improvement is demonstrated for a Q-learning algorithm.

1.1 A High-Level Sketch of the Method

Online SA algorithms attempt to optimize some parameter of the system, using "noise corrupted" system measurements as a data stream for an iterative optimization process. These algorithms deal with noise by making only small changes to the parameters at each step, so that over many iterations the noise averages out, and the parameters asymptotically follow a *mean trajectory.* Intuitively, any prior knowledge about the system should reduce our uncertainty about its behavior and thus enable some noise reduction. In this work we propose a method that *reduces the noise at each step*. We do this by observing that the update at each step can be viewed as a simple estimate of the mean update. Using the partially known MDP we propose an improved estimator, thereby reducing the noise variance. A key property of our estimator is that it is unbiased, thus it *preserves the algorithm's mean trajectory.* This assures us that the overall *function* of the algorithm will remain intact, while the reduction in noise variance gives reason to expect an improvement in *performance*.

1.2 Related Work

One difficulty of RL is coping with the stochasticity inherent to RL algorithms. In this work we define a partially known MDP, and use this partial knowledge to improve the asymptotic performance of stochastic approximation type algorithms. Our method is novel, and specifically deals with this difficulty. We note that a different notion of a partially known MDP was used by Kearns and Singh (2002) and Brafman and Tennenholtz (2003) to tackle a different difficulty of RL - the 'exploration exploitation' tradeoff. Thus, the partial model which we use only to reduce stochastic fluctuations may further be used to explore or exploit more efficiently. On the other hand, the advantage of our approach is that it is general, and may be easily applied to a large class of model free algorithms.

When a full model of state transitions is available, applying our method to Q-learning results in an algorithm known as Real Time Dynamic Programming (RTDP) (Barto et al., 1995). Thus, the method presented in this work may be viewed as a bridge between the model free Q-Learning and the model based RTDP.

In Section 4 we analyze the asymptotic fluctuations in a fixed step TD(0) algorithm with a partial model. A similar analysis of TD(0) without partial model was given by Dayan and Sejnowski (1994) for a decreasing step size and without explicit convergence rate results, and by Konda (2002) using a similar technique to bound the convergence rate. Singh and Dayan (1998) provided update equations for the MSE of TD(0), which we use as a measure of convergence rate, though their equations were only solvable by simulation. This work presents explicit *values* of the asymptotic MSE.

On a slightly different note, an early approach towards a hybrid model based - model free RL algorithm is the Dyna architecture (Sutton, 1990), in which interactions with the environment are used both for a direct policy update, using a model free RL algorithm, and for an update of an environmental model. This model is then used to generate simulated trajectories which are fed to the same model free algorithm for further policy improvement. In a more recent work by Abbeel et al. (2006), a hybrid approach is proposed that combines policy search on an inaccurate model, with policy evaluations in the real environment. Finally, we note that the idea of combining model based and model free approaches has been proposed in the context of animal and human learning, suggesting an explanation for behavioral choice experiments (Daw et al., 2005). To the best of our knowledge, our work presents the first formal proof of the *advantage* of a hybrid algorithm over a standard model free algorithm.

1.3 Organization

This paper is organized as follows. In Section 2 we describe our estimator in the context of estimating the expectation of a random variable. This allows us to derive all its important properties without the notational burden of the SA setting. In Section 3 we describe the RL environment and introduce the partially known MDP. We then describe a method that integrates it in model free SA algorithms. Our main results are in Section 4, in which we analyze how our proposed method influences the algorithm's overall performance. Focusing on TD(0), we show that improvement in performance is achieved. In Section 5 we investigate the effects of inaccuracies in the partial model, and extend our results to inaccurate partially known MDP's. In Section 6 we demonstrate through simulation the applicability of our method to other model free RL algorithms. We conclude and discuss future work in Section 7.

2. Estimation of a Random Variable Mean with Partial Knowledge

Our method of using partial knowledge in an SA algorithm is based on constructing a better estimator for the mean update at each step. In this section we describe our estimator in the context of estimating the mean of a random variable. This allows us to derive all its important properties without the notational burden of the SA setting. The results we derive will then easily transfer to the more complicated SA setting.

Let X be a random variable over a finite and discrete set Ω and let $P(\omega) \triangleq \Pr(X=\omega)$ denote the probability distribution of X. Since $P(\omega)$ contains all the information about X, a natural definition for partial knowledge in this setting is information regarding some of its attributes. In particular, we assume that for an a-priori given subset of Ω, the ratios between the probability distribution values are provided. Denote by K this set for which the ratios of P are known,[1]

$$K \triangleq \left\{ \omega : \omega \in \Omega \text{ s.t. } \frac{P(\omega)}{\sum\limits_{\omega' \in K} P(\omega')} \text{ is known} \right\}. \tag{1}$$

We refer to K as the *partial knowledge* set. Suppose we are given one sample of X, denoted by x, and we wish to estimate (without bias) the expectation $\mu = \mathrm{E}[X] \triangleq \sum_{\omega \in \Omega} \omega P(\omega)$. Our estimator can be any function of x, and of values and probability ratios in the partial knowledge set K.

The Maximum Likelihood (ML) estimator of μ is derived by first using x to generate the ML estimate of the complete probability distribution $\hat{P}(\omega)$, and then calculating the expectation $\sum_\omega \omega \hat{P}(\omega)$. For a given known set K, let $\mathcal{P}_K(\omega)$ denote the set of all probability distributions $P(\omega)$ that satisfy the ratios in (1). If the observed sample x is not in K, then the ML estimate for the probability distribution $\hat{P}(\omega; x \notin K)$ is given by

$$\hat{P}(\omega; x \notin K) = \operatorname*{argmax}_{P(\omega) \in \mathcal{P}_K(\omega)} P(x) = \delta_{x,\omega}, \tag{2}$$

where $\delta_{x,\omega}$ is Kronecker's delta. Conversely, if x is in K, the ML estimate $\hat{P}(\omega; x \in K)$ is

$$\hat{P}(\omega; x \in K) = \operatorname*{argmax}_{P(\omega) \in \mathcal{P}_K(\omega)} P(x) = \frac{\mathbf{1}_\omega^K P(\omega)}{\sum\limits_{\omega' \in K} P(\omega')}, \tag{3}$$

where $\mathbf{1}_\omega^K$ denotes the indicator function that equals 1 if $\omega \in K$ and 0 otherwise. Letting $\bar{K}$ denote the complement of the set K, combining (2) and (3) gives the ML estimate for the probability distribution $\hat{P}(\omega)$

$$\hat{P}(\omega; x) = \mathbf{1}_x^K \frac{\mathbf{1}_\omega^K P(\omega)}{\sum\limits_{\omega' \in K} P(\omega')} + \mathbf{1}_x^{\bar{K}} \delta_{x,\omega}. \tag{4}$$

By taking an expectation of (4), we derive the ML estimate for μ given the partial knowledge, which we denote by $\hat{\mu}_K$

$$\hat{\mu}_K(x) = \mathbf{1}_x^K \cdot \frac{\mathrm{E}[X \cdot \mathbf{1}_X^K]}{\mathrm{E}[\mathbf{1}_X^K]} + \mathbf{1}_x^{\bar{K}} \cdot x. \tag{5}$$

1. Note that knowledge of the exact probability distribution values is a special case of this definition.

Note that (5) uses the partial knowledge in a very intuitive way. It 'replaces' samples in the known set with their weighted average, which by (1) is known. An important property of the estimator $\hat{\mu}_K$ is that it is unbiased, as expressed in the following Lemma.

Lemma 1 *The estimator $\hat{\mu}_K$ is unbiased, namely $\mathrm{E}[\hat{\mu}_K] = \mu$.*

Proof By direct calculation

$$\begin{aligned}\mathrm{E}[\hat{\mu}_K] &= \mathrm{E}[\mathbf{1}_X^K] \cdot \frac{\mathrm{E}[X \cdot \mathbf{1}_X^K]}{\mathrm{E}[\mathbf{1}_X^K]} + \mathrm{E}\left[\mathbf{1}_X^{\bar{K}} \cdot X\right] \\ &= \mathrm{E}\left[X\left(\mathbf{1}_X^K + \mathbf{1}_X^{\bar{K}}\right)\right] = \mathrm{E}[X].\end{aligned}$$

■

In the following Lemma the Mean Squared Error (MSE) of $\hat{\mu}_K$ is computed. Let $P(K) = \sum_{\omega \in K} P(\omega)$, and let $P_K(\omega)$ denote the probability measure over the known set K, namely

$$P_K(\omega) \triangleq \mathbf{1}_\omega^K P(\omega) / P(K).$$

Denote by $\mathrm{E}_K[\cdot]$ and $\mathrm{Var}_K[\cdot]$ the expectation and variance under the probability measure P_K.

Lemma 2 *The MSE of $\hat{\mu}_K$ is $\mathrm{E}\left[(\hat{\mu}_K - \mu)^2\right] = \mathrm{Var}[X] - P(K) \cdot \mathrm{Var}_K[X]$.*

Proof Observe that for any function $f(\cdot)$

$$\begin{aligned}\mathrm{E}_K\left[(f(X) - \mu)^2\right] &= \mathrm{E}_K\left[(f(X) - \mathrm{E}_K f(X) + \mathrm{E}_K f(X) - \mu)^2\right] \\ &= \mathrm{Var}_K f(X) + (\mathrm{E}_K[f(X)] - \mu)^2, \qquad (6)\end{aligned}$$

where the cross terms in the second equality vanish.
Next, we have

$$\begin{aligned}\mathrm{E}[\hat{\mu}_K(X) - \mu]^2 &= \mathrm{E}\left[\left(\mathbf{1}_X^K + \mathbf{1}_X^{\bar{K}}\right)(\hat{\mu}_K(X) - \mu)^2\right] \\ &= P(K)\,\mathrm{E}_K\left[(\hat{\mu}_K(X) - \mu)^2\right] + \mathrm{E}\left[\mathbf{1}_X^{\bar{K}}(X - \mu)^2\right] \\ &= P(K)(\mathrm{E}_K[X] - \mu)^2 + \mathrm{E}\left[\mathbf{1}_X^{\bar{K}}(X - \mu)^2\right] \\ &= P(K)\left(\mathrm{E}_K\left[(X - \mu)^2\right] - \mathrm{Var}_K[X]\right) + \mathrm{E}\left[\mathbf{1}_X^{\bar{K}}(X - \mu)^2\right] \\ &= \mathrm{E}\left[(X - \mu)^2\right] - P(K) \cdot \mathrm{Var}_K[X],\end{aligned}$$

where in the fourth equality we used (6) with $f(X) = X$. ■

One could disregard the partial knowledge altogether, and choose to use the sample x itself as an unbiased estimate for μ. Denote this estimator, which will be referred to as the sample estimator, by

$$\hat{\mu}(x) = x. \qquad (7)$$

When no partial information is available, $\hat{\mu}$ seems like the most reasonable choice (actually, it can be shown that $\hat{\mu}$ is the only unbiased estimator in that case). It is easy to see that the MSE of $\hat{\mu}$ is

$\mathrm{Var}[X]$, and from Lemma 2 we deduce that when the cardinality of the known set satisfies $|K| > 1$, and $P(K) > 0$, the MSE of $\hat{\mu}_K$ is smaller than that of $\hat{\mu}$. In parameter estimation parlance, we say that $\hat{\mu}_K$ *dominates* $\hat{\mu}$ (Schervish, 1995).

As will be shown in the next section, the update at each iteration of an SA algorithm can be seen as the estimation of an expected update direction. This estimation is based on one sample, obtained through observation of the system dynamics at that step, and the estimator used is just $\hat{\mu}$. When partial knowledge of these dynamics is available, we propose to use $\hat{\mu}_K$ instead, and benefit from its reduced variance.

An appropriate question at this point is whether a better estimator than $\hat{\mu}_K$ exists. We refer the interested reader to appendix A, where we show that $\hat{\mu}_K$ is admissible. For the following discussion however, the results of Lemmas 1 and 2 suffice.

3. A Stochastic Approximation Algorithm with Partial Model Knowledge

In this section we describe our method of endowing a model free RL algorithm with partial model knowledge. We start with some general definitions of the RL environment and SA algorithms. Then, we consider a situation where partial knowledge of the environment model is available. Based on the estimator developed in the previous section, we propose a general form of SA algorithms that incorporate such knowledge.

3.1 Preliminaries

We describe the notation used throughout the paper, the RL environment, and the stochastic approximation method.

3.1.1 NOTATION

Throughout the rest of the paper the following notation is used. All vectors are column vectors, and $(\cdot)^T$ denotes the transpose operator. The product $A \circ B$ denotes the element-wise product (Hadamard product) of A and B. $\mathrm{Tr}[\cdot]$ is the trace of a matrix. The cardinality of a set K is denoted by $|K|$, and its complement by $\bar{K}$. Unless noted otherwise, a subscript of a variable denotes time. The i-th element of a vector A is denoted by $[A]_i$ or $A(i)$, depending on the context. The (i,j) element of a matrix B is denoted by $[B]_{ij}$.

3.1.2 RL ENVIRONMENT

We consider an agent interacting with an unknown environment, modeled by an MDP in discrete time with a finite state set $\mathcal{X}$ and action set $\mathcal{U}$. Each selected action $u \in \mathcal{U}$ at a state $x \in \mathcal{X}$ determines a stochastic transition to the next state $y \in \mathcal{X}$ with a probability $P_u(y|x)$.

For each state x the agent receives a corresponding deterministic reward $r(x)$, which is bounded by $r_{\max}$, and depends only on the current state.[2] The agent maintains a *policy function,* $\mu_\theta(u|x)$, parametrized by a vector $\theta \in \mathbb{R}^L$, mapping a state x into a probability distribution over the actions $\mathcal{U}$. Under policy μ_θ, the environment and the agent induce a Markovian transition matrix, denoted by P_{μ_θ}, which we assume to be ergodic.[3] This Markovian transition matrix has a stationary distribution

2. Generalizing the results presented here to state-action rewards is straightforward. Generalization to stochastic rewards is also possible by considering mean rewards.
3. That is, aperiodic, recurrent, and irreducible.

over the state space $\mathcal{X}$, denoted by π_{μ_θ}. Let $\Pi_{\mu_\theta} \in \mathbb{R}^{|\mathcal{X}|\times|\mathcal{X}|}$ be a diagonal matrix where its elements are $\Pi_{\mu_\theta} = \mathrm{diag}(\pi_{\mu_\theta})$. Our goal is to optimize θ with respect to some performance criteria. The tuning of θ is performed online in the following fashion. At time n, the current parameter value equals θ_n and the agent is in state x_n. It then chooses an action u_n according to $\mu_{\theta_n}(u|x_n)$, observes x_{n+1}, and updates θ_{n+1} according to some protocol.

3.1.3 STOCHASTIC APPROXIMATION

Stochastic approximation methods (Kushner and Yin, 2003; Borkar, 2008) are a class of iterative stochastic algorithms, to which many model free RL algorithms belong (Bertsekas and Tsitsiklis, 1996). Analysis of SA methods has received considerable attention over the past few decades, and many analysis techniques are available. In particular, the ODE approach introduced by Ljung (1977), is a widely used method for investigating the asymptotic behavior of SA iterates. The algorithms that we deal with in this paper are all cast in the following SA form,[4]

$$\theta_{n+1} = \theta_n + \varepsilon_n F(\theta_n, x_n, u_n, x_{n+1}), \tag{8}$$

where $\{\varepsilon_n\}$ are positive step sizes. The key idea of the technique is the following. Iterate (8) can be decomposed into a deterministic function of the current state, action and parameter, denoted by $g(\theta_n, x_n, u_n)$, and a martingale difference noise term δM_n,

$$\theta_{n+1} = \theta_n + \varepsilon_n (g(\theta_n, x_n, u_n) + \delta M_n), \tag{9}$$

where $g(\theta_n, x_n, u_n) \triangleq \mathrm{E}[F(\theta, x_n, u_n, x_{n+1})|\theta_n, x_n, u_n]$, $\delta M_n \triangleq F(\theta_n, x_n, u_n, x_{n+1}) - g(\theta_n, x_n, u_n)$, and the expectation is taken over the next state x_{n+1}.

Suppose that the effect of the martingale difference noise weakens due to repeated averaging, and further assume that there exists a continuous function $\bar{g}(\theta)$ such that $\frac{1}{m}\sum_{i=n}^{m+n-1} g(\theta, x_i, u_i) \to \bar{g}(\theta)$ w.p.1 as $m, n \to \infty$.[5] Consider the following ordinary differential equation (ODE)

$$d\theta/dt = \bar{g}(\theta). \tag{10}$$

Then, a typical result of the ODE method in the SA setup suggests that the asymptotic limits of (8) and (10) are identical. Another aspect of SA relates to the rate of convergence of such iterates (Kushner and Yin, 2003), an issue that will be elaborated on later.

3.1.4 A NOTE ON TYPES OF CONVERGENCE

The type of convergence to the asymptotic limit depends primarily on the step size used. Let θ^* denote an asymptotically stable fixed point of (10), and assume that it is unique. Then, for a suitably decreasing step size, convergence w.p. 1 of θ_n to θ^* can be established. For a constant step size, θ_n can be shown to converge weakly to a random variable centered on θ^*. In the following we use the term convergence ambiguously, and the precise definition should be inferred from the context. For a detailed and rigorous discussion of the types of convergence in SA the reader is referred to Kushner and Yin (2003).

4. This is not the most general SA form, but one that is cast to the RL setup.

5. Note that for stationary policies, the strong law of large numbers for Markov chains may be used to write $\bar{g}$ explicitly $\bar{g}(\theta) = \mathrm{E}[g(\theta, x, u)|\theta] = \sum_{x\in X} \pi_{\mu_\theta}(x) \sum_{u\in U} \mu_\theta(u|x) g(\theta, x, u)$.

3.2 Partial Model Based Algorithm

A key observation obtained from examining Equations (8-9), is that $F(\theta_n, x_n, u_n, x_{n+1})$ in the SA algorithm is just the sample estimator (7) of $g(\theta_n, x_n, u_n)$, the mean update at each step. The estimation variance in this case stems from the stochastic transition from x_n to x_{n+1}. In the following we assume that we have, prior to running the algorithm, some information about these transitions in the form of partial transition probability ratios. Similarly to Section 2, define the known set for state x and action u as

$$K_{x,u} \triangleq \left\{ y : y \in \mathcal{X} \text{ s.t. } \frac{P_u(y|x)}{\sum_{y' \in K_{x,u}} P_u(y'|x)} \text{ is known} \right\}. \tag{11}$$

We refer to the known sets for all states and actions as the *partially known MDP*.

It is clear that definition (11) is motivated by the theoretical results presented in Section 2, and at this point it may well be questioned whether such a definition has any use in practice. We refer the concerned reader to Section 6, where it is shown that in certain problems definition (11) arises as the *natural* representation of partial model knowledge.

Denote by $\mathbf{1}^K_{n+1}$ an indicator function that equals 1 if $\{x_{n+1}\}$ belongs to K_{x_n,u_n} and 0 otherwise. Based on the estimator introduced in Section 2, we propose the following update rule for the tunable parameter, denoted by θ^K, which we refer to as the *Integrated Partial Model* (IPM) iteration

$$\theta^K_{n+1} = \theta^K_n + \varepsilon_n \left(\mathbf{1}^K_{n+1} F^K_n + \mathbf{1}^{\bar{K}}_{n+1} F(\theta^K_n, x_n, u_n, x_{n+1}) \right), \tag{12}$$

where, abusing notation, $F^K_n = F^K_n(\theta^K_n, x_n, u_n)$, and

$$F^K_n \triangleq \frac{\sum_{y \in K_{x_n,u_n}} P_{u_n}(y|x_n) F(\theta^K_n, x_n, u_n, y)}{\sum_{y \in K_{x_n,u_n}} P_{u_n}(y|x_n)}. \tag{13}$$

Similarly to (9), iterate (12) can also be decomposed into a mean function $g^K(\theta^K_n, x_n, u_n)$ and a martingale difference noise δM^K_n

$$\theta^K_{n+1} = \theta^K_n + \varepsilon_n \left(g^K(\theta^K_n, x_n, u_n) + \delta M^K_n \right),$$

and by Lemma 1 we have $g^K(\theta, x, u) = g(\theta, x, u)$. Similarly, defining $\bar{g}^K(\theta) = \mathrm{E}\left[g^K(\theta, x, u) | \theta \right]$ we get that $\bar{g}^K(\theta) = \bar{g}(\theta)$, and we reach the following important conclusion, which is summarized as a theorem.

Theorem 3 *The IPM iteration defined in* (12) *leads to the same characteristic ODE* $d\theta/dt = \bar{g}(\theta)$ *as the regular SA iteration* (8).

Since the asymptotic behavior of the SA iterate (8) is governed by its ODE, Theorem 3 assures us that using the IPM iteration (12) does not change this behavior, and thus the *function* of the algorithm remains intact. If (8) can be shown to converge to some limit point, iterate (12) can be shown to converge *to the same limit*. Furthermore, from Lemma 2 we have that if the partially known MDP is not null, then on each iteration the variance of the noise term is reduced. This gives us reason to expect an improvement in the overall performance of the algorithm.

3.3 Step Size Considerations

As it turns out, the improvement in performance attained by the IPM iteration is heavily influenced by the step size used. This can be intuitively explained using the following example. Let $\{z_i\}$ be a sequence of i.i.d. bounded random variables, with mean μ_z and variance σ_z^2. Consider the following SA iteration

$$\theta_{n+1} = \theta_n + \varepsilon_n (z_{n+1} - \theta_n).$$

For a decreasing step size of the form $\varepsilon_n = 1/(n+1)$, the value of θ_n is simply the empirical average, which converges w.p. 1 to μ_z. As a performance measure, consider the MSE defined by $\mathrm{E}\|\theta_n - \mu_z\|^2$, which equals σ_z^2/n. Integration of partial knowledge based on (12) in this case is equivalent to averaging variables with the same mean but with a reduced variance, and the MSE still approaches zero at a rate $O(1/n)$. On the other hand, when the step size is constant, θ_n converges in mean to μ_z, but the MSE converges to a non-zero value which, intuitively, is proportional[6] to the variance σ_z^2. Any variance reduction in this case would thus prove valuable.

The use of a constant step size, though clearly undesirable in the preceding example, is quite common in RL applications, as it allows the iterates to quickly reach a neighborhood of the desired solution, and can cope with time varying environments. In the following discussion, we shall thus focus our analysis on algorithms with a constant step size.

4. TD(0) with Partial Model Knowledge

In this section we apply our IPM method of Equation (12) to the well known model free algorithm Temporal Difference (TD(0); Sutton and Barto, 1998). The simplicity of TD(0) allows us to mathematically characterize its performance in terms of *convergence rate*, and to quantify the impact of using the IPM method on it. The mathematical results we derive specifically for TD(0) are also characteristic of more complex algorithms, as will be shown in subsequent sections.

4.1 Definitions

Throughout this section, we assume that the agent's policy μ is deterministic and fixed, mapping a specific action to each state, denoted by $u(x)$.

4.1.1 VALUE FUNCTION ESTIMATION

Letting $0 < \gamma < 1$ denote a discount factor, define the value function for state x under policy μ as the expected discounted return when starting from state x and executing policy μ

$$V^{\mu}(x) \triangleq \mathrm{E}\left[\sum_{t=0}^{\infty} \gamma^t r(x_t) \,\middle|\, x_0 = x\right].$$

Since in this section the policy μ is constant, from now on we omit the superscript μ in $V^{\mu}(x)$, and the subscript μ in P_{μ} , π_{μ}, and Π_{μ}.

The value function is a vector of size $|\mathcal{X}|$. When the state space is large, Function Approximation (FA) is often used to find an approximation to the value function in a subspace of size $L < |X|$. Linear

6. A precise value is given in the next section.

FA is implemented as follows. Given a set of $|X|$ linearly independent basis vectors $\phi(x) \in \mathbb{R}^L$, the goal is to find an approximation to $V(x)$, denoted by $\hat{V}(x,\theta)$ and defined as

$$\hat{V}(x,\theta) = \phi(x)^T \theta.$$

Note that the tunable parameter θ in this case is a vector of L linear weights. In vector form we write $\hat{V}(\theta) = \Phi\theta$, where $\Phi \in \mathbb{R}^{|X|\times L}$ is a matrix composed of rows of basis vectors.

Define the *Temporal Difference* (TD) at time n as

$$d_n \triangleq r(x_n) + \gamma\phi(x_{n+1})^T\theta_n - \phi(x_n)^T\theta_n.$$

For some small step size ε, the fixed step TD(0) algorithm updates θ online in the following manner,

$$\theta_{n+1} = \theta_n + \varepsilon d_n \phi(x_n). \tag{14}$$

This is an SA algorithm as defined in (8), and its associated ODE is (Bertsekas and Tsitsiklis, 1996, Lemma 6.5)

$$\frac{d\theta}{dt} = b + A\theta. \tag{15}$$

where

$$\begin{aligned} A &\triangleq \Phi^T \Pi (\gamma P - I) \Phi, \\ b &\triangleq \Phi^T \Pi r. \end{aligned} \tag{16}$$

Equation (15) is linear and has a fixed point θ^* that satisfies

$$A\theta^* = -b.$$

Furthermore, the eigenvalues of A all have a negative real part (Bertsekas and Tsitsiklis, 1996, Lemma 6.6b), and therefore θ^* is a unique and stable fixed point.

4.1.2 INTEGRATED PARTIAL MODEL TD(0)

We now use the method developed in Section 3 to integrate a partial model into the TD(0) algorithm. Since the policy is deterministic we drop the u subscript in the known set definition. Using (12) and (13) we define IPM-TD(0)

$$\begin{aligned} \theta^K_{n+1} &= \theta^K_n + \varepsilon d^K_n \phi(x_n), \\ d^K_n &\triangleq r(x_n) + \gamma\left(\mathbf{1}^K_{n+1} F^K_n + \mathbf{1}^{\bar{K}}_{n+1} \phi(x_{n+1})^T \theta^K_n\right) - \phi(x_n)^T \theta^K_n, \\ F^K_n &\triangleq \frac{\sum_{y\in K_{x_n}} P(y|x_n)\phi(y)^T\theta^K_n}{\sum_{y\in K_{x_n}} P(y|x_n)}. \end{aligned} \tag{17}$$

Using Theorem 3 we conclude that the IPM-TD(0) iterates have the same characteristic ODE as the TD(0) iterates of Equation (14), and therefore converge to the same fixed point θ^* of (15).

After establishing that the asymptotic trajectory (or, in other words the algorithmic 'function') of the algorithm remains intact, we shall now investigate whether adding the partial knowledge can be guaranteed to improve *performance*.

4.2 Performance Improvement Proof

In this section we prove that the performance of the IPM-TD(0) iteration is superior in terms of asymptotic MSE to regular TD(0). The formal approach we follow here may be carried out for other SA algorithms as well, though the expressions involved may become more complicated.

Recall that the asymptotic limit point of both regular TD(0) and IPM-TD(0) is θ^*. A natural performance measure in this case is the asymptotic MSE defined by

$$\lim_{n\to\infty} \mathrm{E}\left\|\theta_n - \theta^*\right\|^2.$$

The remainder of this section is devoted to showing that integrating a partial model reduces the asymptotic MSE, namely

$$\lim_{n\to\infty} \mathrm{E}\left\|\theta_n^K - \theta^*\right\|^2 < \lim_{n\to\infty} \mathrm{E}\left\|\theta_n - \theta^*\right\|^2,$$

whenever the known set K is not null.

By Lemma 2, at each iteration step we are guaranteed (as long as our partial model is not null) a reduction in the noise variance. This clearly indicates that some improvement in the asymptotic MSE can be expected, but a precise quantification of this is more complicated. A powerful tool for this task is the rate of convergence theory for SA (or a 'limit theorem for fluctuations', as termed by Borkar, 2008). In their treatment of rate of convergence, Kushner and Yin (2003, p. 315) discuss the properties of the sequence

$$\rho_n \triangleq \left(\theta_n - \theta^*\right)/\sqrt{\varepsilon}. \tag{18}$$

Application of their Theorem 10.1.3 to the TD(0) iteration results in the following theorem.

Theorem 4 *The sequence ρ_n converges in distribution, as $\varepsilon \to 0$ and $n \to \infty$ such that $n\varepsilon \to \infty$,*[7] *to a normally distributed random variable, which is the stationary distribution of the stochastic differential equation*

$$dU = AU\,dt + dW. \tag{19}$$

A is defined in (16)*, and W is a Wiener process with covariance matrix $\Sigma = \Sigma_0 + \Sigma_1 + \Sigma_1^T$ where*

$$\Sigma_0 = \lim_{n\to\infty} E\left[\left(d_n\phi\left(x_n\right)\right)\left(d_n\phi\left(x_n\right)\right)^T \middle| \theta_n = \theta^*\right], \tag{20}$$

$$\Sigma_1 = \sum_{j=1}^{\infty} \lim_{n\to\infty} E\left[\left(d_n\phi\left(x_n\right)\right)\left(d_{n+j}\phi\left(x_{n+j}\right)\right)^T \middle| \theta_n = \theta_{n+j} = \theta^*\right].$$

For the IPM iteration (17) *we have Σ_0^K, Σ_1^K where d_n^K replaces d_n in* (20)*.*

The proof of Theorem 4 consists of verifying a lengthy set of technical assumptions required for Theorem 10.1.3 of Kushner and Yin (2003), and is fully described in Appendix E.

The stationary solution to (19) is normally distributed with zero mean and covariance R, which can be easily computed (Papoulis and Pillai, 2002, §9.2) by observing that (19) describes Gaussian white noise filtered through a linear system, leading to

$$R = \lim_{t\to\infty} e^{At}\left\{\int_0^t e^{-As}\Sigma\left(e^{-As}\right)^T ds\right\}\left(e^{At}\right)^T. \tag{21}$$

7. We retain this assumption on ε and n in the sequel.

Let $\{\lambda_i\}_{i=1}^L$ denote the eigenvalues of A, which all have a negative real part (Bertsekas and Tsitsiklis, 1996, Lemma 6.6b), and let Γ be its diagonalizing matrix, that is, $A = \Gamma\Lambda\Gamma^{-1}$ where Λ is diagonal. Also, define a matrix $\chi \in \mathbb{R}^{L\times L}$ such that $[\chi]_{ij} = -1/(\lambda_i + \lambda_j)$. The limit in (21) can be written as

$$R = \lim_{t\to\infty} \Gamma \left\{ \int_0^t e^{\Lambda(t-s)} \Gamma^{-1} \Sigma \left(\Gamma^{-1}\right)^T e^{\Lambda(t-s)} ds \right\} \Gamma^T. \tag{22}$$

Note that the term in the curly brackets in (22) is a matrix, with its (i,j)th component equal to

$$\begin{aligned} & \int_0^t e^{\lambda_i(t-s)} \left[\Gamma^{-1}\Sigma\left(\Gamma^{-1}\right)^T\right]_{i,j} e^{\lambda_j(t-s)} ds \\ = \quad & \left[\Gamma^{-1}\Sigma\left(\Gamma^{-1}\right)^T\right]_{i,j} (\lambda_i+\lambda_j)^{-1} \left(-1 + e^{(\lambda_i+\lambda_j)t}\right). \end{aligned}$$

Substituting in (22) and taking the limit gives

$$R = \Gamma\left(\chi \circ \left(\Gamma^{-1}\Sigma\left(\Gamma^{-1}\right)^T\right)\right)\Gamma^T,$$

and using (18) and Theorem 4, the limit of the MSE is

$$\mathrm{E}\,\|\theta_n - \theta^*\|^2 \to \varepsilon \mathrm{Tr}\left[\Gamma\left(\chi \circ \left(\Gamma^{-1}\Sigma\left(\Gamma^{-1}\right)^T\right)\right)\Gamma^T\right]. \tag{23}$$

The difference in MSE between the original iterate (14) and the IPM iterate (17) lies in the difference between Σ_0, Σ_0^K and Σ_1, Σ_1^K. We now derive explicit expressions for these matrices. In the following, for clarity we adopt the following notation. Let x' denote the state following x. Let $P(K_x) = \sum_{x'\in K_x} P(x'|x)$, and let $P_{K_x}(x')$ denote the probability measure over the known transitions from state x, namely

$$P_{K_x}(x') \triangleq \begin{cases} P(x'|x)/P(K_x) & if \quad x' \in K_x \\ 0 & if \quad x' \notin K_x \end{cases}.$$

Denote by $\mathrm{E}_K[f(x')|x]$, $\mathrm{Var}_K[f(x')|x]$, and $\mathrm{Cov}_K[f(x')|x]$ the expectation, variance, and covariance matrix of some function f of x' given that the current state is x, under the probability measure P_{K_x}.

Lemma 5 *We have* $\Sigma_0 = \Sigma_0^K + \gamma^2 \sum_x [\pi]_x \phi(x) \mathrm{Var}_K\left[\phi(x')^T\theta^* \middle| x\right] \phi(x)^T$.

Proof See Appendix B. ■

In order to simplify calculations, in the remainder of the analysis we deal with a table based algorithm.

Assumption 6 *The algorithm is table based, namely* $\Phi = I$.

Under the table based assumption, the temporal difference terms at subsequent times are not correlated, leading to the following result.

Proposition 7 *Under assumption 6 we have* $\Sigma_1 = \Sigma_1^K = 0$.

Proof For a table based case, θ^* satisfies Bellman's equation for a fixed policy (Bertsekas and Tsitsiklis, 1996)

$$\theta^*(x) = r(x) + \gamma \mathrm{E}\left[\theta^*\left(x'\right)\middle| x\right]. \tag{24}$$

Now, for every j we have

$$\begin{aligned}
& \mathrm{E}\left[\left(d_n \phi(x_n)\right)\left(d_{n+j}\phi(x_{n+j})\right)^T \middle| \theta_n = \theta_{n+j} = \theta^*\right] \\
= \; & \mathrm{E}\left[\left(r(x_n) + \gamma\theta^*(x_{n+1}) - \theta^*(x_n)\right)\left(r(x_{n+j}) + \gamma\theta^*(x_{n+j+1}) - \theta^*(x_{n+j})\right)\right] \\
= \; & \mathrm{E}\left[\mathrm{E}\left[\left(r(x_n) + \gamma\theta^*(x_{n+1}) - \theta^*(x_n)\right)\left(r(x_{n+j}) + \gamma\theta^*(x_{n+j+1}) - \theta^*(x_{n+j})\right)\middle| x_n, \ldots, x_{n+j}\right]\right] \\
= \; & 0,
\end{aligned}$$

where the last equation follows from (24). Thus, in the expression for Σ_1, every element in the sum is zero. For Σ_1^K we can use Lemma 1 to obtain the same result. ■

Generalizing these results to the FA case involves analysis of the correlations in Σ_1, Σ_1^K and is deferred to future work. Nevertheless, we provide numerical simulations with FA that demonstrate similar behavior to the table based case.

Let Δ_Σ denote the diagonal matrix defined by

$$\begin{aligned}
\Delta_\Sigma &\triangleq \Sigma_0 - \Sigma_0^K \\
&= \gamma^2 \sum_x [\pi]_x \phi(x) P(K_x) \mathrm{Var}_K\left[\phi(x')^T \theta^* \middle| x\right] \phi(x)^T.
\end{aligned}$$

Substituting $\Phi = I$ gives a simple expression for the diagonal elements of Δ_Σ

$$[\Delta_\Sigma]_{xx} = \gamma^2 [\pi]_x P(K_x) \mathrm{Var}_K\left[\theta^*\left(x'\right)\middle| x\right].$$

Note that Δ_Σ has no negative elements. We are interested in the difference in asymptotic MSE, which, based on (23) is given by

$$\begin{aligned}
\delta_{MSE} &= \mathrm{E}\|\theta_n - \theta^*\|^2 - \mathrm{E}\|\theta_n^K - \theta^*\|^2 \\
&\to \varepsilon \cdot \mathrm{Tr}\left[\Gamma\left(\chi \circ \left(\Gamma^{-1}\Delta_\Sigma\left(\Gamma^{-1}\right)^T\right)\right)\Gamma^T\right].
\end{aligned} \tag{25}$$

If the known set is not null, then δ_{MSE} is positive (it can be seen as the asymptotic MSE of an iterate with the same matrix A, but with Δ_Σ instead of Σ_0, which by definition is positive), and thus the algorithm's performance improves. We summarize this result in the following theorem.

Theorem 8 *Consider the table based online TD(0) iterate for* θ_n *described by* (14) *with* $\Phi = I$*, and the IPM-TD(0) iterate for* θ_n^K *described by* (17) *with the same requirement on* Φ*. Assuming that there is at least one state* $x \in \mathcal{X}$ *such that* $P(K_x)\mathrm{Var}_K\left[\theta^*(x')|x\right] > 0$*, then the asymptotic MSE of the iterates satisfy* $\lim_{n\to\infty} E\|\theta_n^K - \theta^*\|^2 = \lim_{n\to\infty} E\|\theta_n - \theta^*\|^2 - \delta_{MSE}$*, where* δ_{MSE} *is given in* (25)*, and* $\delta_{MSE} > 0$*.*

Theorem 8 therefore assures us that the reduction in noise variance at each step, guaranteed by Lemma 2, translates into a reduction in the overall error of the algorithm. Note that the simple dependence of the MSE on ε allows for a different interpretation of the performance in terms of *convergence rate* - for some desired MSE, the partial knowledge allows us to use a larger step size ε, and thus *converge faster*. This issue will also be demonstrated in simulation.

We comment on a decreasing step size. For a step size of the form $\varepsilon_n = 1/n^{\alpha}$, $0.5 < \alpha \leq 1$, a similar analysis can be performed with ρ_n defined as $\rho_n = n^{\alpha/2}(\theta_n - \theta^*)$. In this case, θ_n converges to θ^* w.p. 1, and the MSE decreases to zero at a rate $O\left(n^{-\alpha/2}\right)$. Integrating a partial model in this case will reduce fluctuations in the converging path of the system. The performance gain of integrating a partial model is therefore more pronounced when the step size is constant.

4.3 Numerical Simulations of IPM-TD(0)

We conclude this section with a demonstration of the performance of the IPM-TD(0) algorithm, and a comparison with the theory established above.[8]

Our simulations are on a set of abstract randomly constructed MDP's termed Generalized Average Reward Non-stationary Environment Test-bench or in short GARNET (Bhatnagar et al., 2007). GARNET MDP's comprise a class of randomly constructed finite MDP's serving as a test-bench for RL algorithms. A GARNET MDP is characterized in our case by four parameters and is denoted by GARNET$(|\mathcal{X}|, |\mathcal{U}|, B, \sigma)$. The parameter $|\mathcal{X}|$ is the number of states in the MDP, $|\mathcal{U}|$ is the number of actions, B is the branching factor of the MDP, that is, the number of uniformly distributed non-zero entries in each line of the MDP's transition matrices, and the reward in each state is normally distributed with variance σ. For each GARNET MDP we also construct a 'partially known' MDP characterized by a parameter p_K, $0 \leq p_K \leq 1$ such that each transition in the original MDP is known w.p. p_K. The value of p_K therefore indicates our level of knowledge about the MDP, ranging from no knowledge at all ($p_K = 0$) up to knowing the complete MDP ($p_K = 1$).

For a GARNET$(10, 5, 10, 1)$ MDP, a random deterministic policy was chosen and its value function was evaluated using algorithm (17). The error $\|\theta_n^K - \theta^*\|^2$, averaged over 500 different runs with the same initial conditions, is plotted in Figure 1 (left) for different values of p_K. The asymptotic MSE was calculated using (23) and is shown for comparison. In Figure 1 (middle), the step size for an iteration with partial knowledge was set such that the asymptotic MSE would match that of the iteration without partial knowledge. As can be seen, this caused the IPM iteration to converge *faster*.

For the next simulation a linear FA was used, with basis vectors $\phi(x) \in \{0,1\}^L$, where the number of nonzero values in each $\phi(x)$ is l. The nonzero values were chosen uniformly at random, with any two states having different feature vectors. Figure 1 (right) shows the error $\|\theta_n^K - \theta^*\|^2$for a GARNET$(30, 5, 10, 1)$ MDP, where we used linear FA with $L = 10$ and $l = 2$. As can be seen, the behavior observed in the tabular case is characteristic of the FA case as well.

5. Inaccuracy of the Partial Model

Until now, we have assumed that our partial model contained accurate probability ratios. Obviously, such a strong assumption is not realistic, and in any practical situation our partial knowledge would contain some degree of error. In this section we consider the effect of inaccuracies in the partial model on the performance of the IPM-TD(0) method. Specifically, our goal is to show that if the inaccuracy in the partially known model is small enough, then an improvement in performance over regular TD(0) can still be guaranteed, and we seek bounds on the error in the algorithm induced by the inaccuracy in the model. Before we go into mathematical detail we first describe our conceptual approach.

8. The code for generating the results presented here can be found at the author's web site.

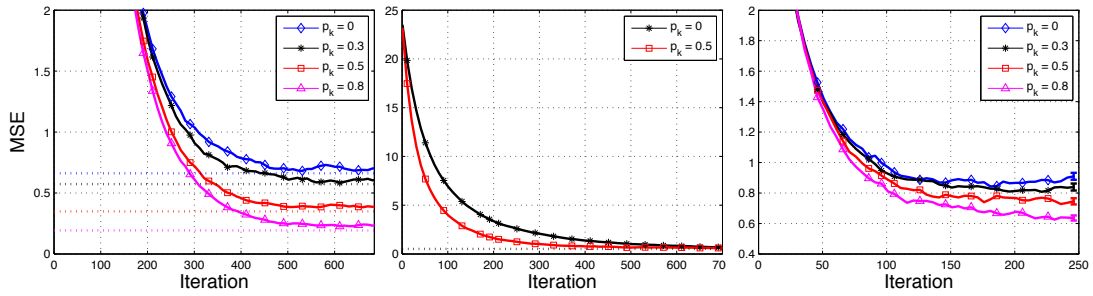


Figure 1: TD(0) with a Partial Model. *Left* : MSE of Table Based IPM-TD(0) on a GARNET(10,5,10,1) MDP with a deterministic random policy, for different values of p_K. Step size is $\varepsilon = 0.2$. Dashed lines show the asymptotic MSE calculated by (23). *Middle* : MSE of Table Based IPM-TD(0) on a GARNET(10,5,10,1) MDP with a deterministic random policy. For $p_K = 0$ (black-solid) a step size $\varepsilon = 0.15$ was used, and the asymptotic MSE was calculated using (23) (black-dashed). For $p_K = 0.5$ (red-solid) a step size was calculated (using (23)) such that its asymptotic MSE would equal that of $p_K = 0$. *Right* : MSE of linear FA IPM-TD(0) on a GARNET(30,5,10,1) MDP with a deterministic random policy, for different values of p_K. Step size is $\varepsilon = 0.15$. The linear FA parameters are $L = 10$ and $l = 2$. A discount factor of $\gamma = 0.7$ was used in all simulations. All results are averaged over 500 different runs with the same initial conditions. Error bars display the standard error of the mean; for clarity of presentation the bars are displayed only for the last iteration.

A key point in the analysis of IPM-TD(0) in Section 4 was that since the estimator $\hat{\mu}_K$ in (5) is unbiased, then the ODE of the stochastic approximation does not change, and asymptotically the algorithm concentrates around its fixed point which is the true value function. This is no longer valid when the partial model is not accurate, as the inaccuracy induces a bias in $\hat{\mu}_K$. Since we use the estimator at every time step, this bias may accumulate, and the crucial question here is how it affects the algorithm asymptotically, and whether it can be guaranteed that small model errors do not cause a large deviation from the true value function.

The improvement in performance of IPM-TD(0) relied on the variance reduction property of $\hat{\mu}_K$. We shall see that if the inaccuracy in the partial model is small enough, then this property can still be guaranteed.

Thus, our analysis consists of investigating the *bias* and the *variance* of IPM-TD(0) with an inaccurate model. As we have done earlier, we first describe some results in the context of estimating the mean of a random variable, and later extend the results to the MDP setting.

5.1 Estimation of a Random Variable Mean

Consider the definitions of Section (2), and let $\left\{\hat{P}(\omega)\right\}_{\omega\in\Omega}$ denote inaccurate probabilities, obtained by some means. For some $\varepsilon > 0$ we define an $\varepsilon-$known set K^ε by

$$K^\varepsilon \triangleq \left\{\omega : \omega \in \Omega \text{ s.t. } \left|\hat{P}_{K^\varepsilon}(\omega) - P_{K^\varepsilon}(\omega)\right| < \varepsilon\right\}, \tag{26}$$

where the probability measures $\hat{P}_{K^\varepsilon}$ and P_{K^ε} are defined by $\hat{P}_{K^\varepsilon}(\omega) \triangleq \hat{P}(\omega) / \sum_{\omega' \in K^\varepsilon} \hat{P}(\omega')$ and $P_{K^\varepsilon}(\omega) \triangleq P(\omega) / \sum_{\omega' \in K^\varepsilon} P(\omega')$, respectively. Also denote by $\hat{\mathrm{E}}_{K^\varepsilon}[\cdot]$ and $\hat{\mathrm{Var}}_{K^\varepsilon}[\cdot]$ the expectation and variance under the measure $\hat{P}_{K^\varepsilon}$, and by $\mathrm{E}_{K^\varepsilon}[\cdot]$ and $\mathrm{Var}_{K^\varepsilon}[\cdot]$ the expectation and variance under the measure P_{K^ε}.

We motivate the definition of the $\varepsilon-$known set with an example. Let $\{x^i\}_{i=1}^n$ denote i.i.d. samples of X. For some set $K \subset \Omega$ let $\hat{P}_K(\omega)$ denote the count ratios in K

$$\hat{P}_K(\omega) = \begin{cases} \frac{\sum_{i=1}^n \mathbf{1}(x^i=\omega)}{\sum_{i=1}^n \mathbf{1}(x^i \in K)} & \text{for } \omega \in K \\ 0 & \text{else} \end{cases},$$

where $\mathbf{1}$ is the indicator function. It can be shown that $\hat{P}_K(\omega)$ is an unbiased estimate of $P_K(\omega)$, and by the law of large numbers we have that for large n, the difference $\left|\hat{P}_K(\omega) - P_K(\omega)\right|$ is small. Furthermore, for a finite n, Chernoff type bounds can be used to bound this difference with high probability by some small ε, motivating definition (26).

An estimator for μ that uses the $\varepsilon-$known set is derived by plugging K^ε instead of K in (5)

$$\hat{\mu}_{K^\varepsilon}(x) = \mathbf{1}_x^{K^\varepsilon} \hat{\mathrm{E}}_{K^\varepsilon}[X] + \mathbf{1}_x^{\bar{K}^\varepsilon} x. \tag{27}$$

Note that since the known set is not accurate, the estimator (27) is no longer unbiased. The following theorem, which we prove in Appendix C, bounds the bias and variance of $\hat{\mu}_{K^\varepsilon}(x)$.

Theorem 9 *The bias of $\hat{\mu}_{K^\varepsilon}(x)$ satisfies*

$$\left|\mathrm{E}\left[\hat{\mu}_{K^\varepsilon}(X)\right] - \mathrm{E}[X]\right| \leq \varepsilon P(K^\varepsilon) \sum_{x \in K^\varepsilon} |x|.$$

The variance of $\hat{\mu}_{K^\varepsilon}(x)$ satisfies

$$\begin{aligned} Var\left[\hat{\mu}_{K^\varepsilon}(X)\right] \leq\ & Var[X] - P(K^\varepsilon) \cdot Var_{K^\varepsilon}[X] \\ & + \varepsilon P(K^\varepsilon) \left(\varepsilon \left(\sum_{x \in K^\varepsilon} |x| \right)^2 + 2 \left(\sum_{x \in K^\varepsilon} |x| \right) \left| E_{K^\varepsilon}[X] - E[X] \right| \right). \end{aligned} \tag{28}$$

5.2 Error Bound for IPM-TD(0)

We now derive asymptotic error bounds for IPM-TD(0) with a constant stepsize $\tilde{\varepsilon}$, when the partial model is inaccurate. We treat only the table based algorithm

$$\begin{aligned} \theta_{n+1}(x_n) &= \theta_n(x_n) + \tilde{\varepsilon} d_n^K, \\ d_n^K &\triangleq r(x_n) + \gamma \left(\mathbf{1}_{n+1}^{K^\varepsilon} \hat{\mathrm{E}}_{K^\varepsilon_{x_n}}\left[\theta_n(x_{n+1}) | x_n\right] + \mathbf{1}_{n+1}^{\bar{K}^\varepsilon} \theta_n(x_{n+1}) \right) - \theta_n(x_n), \end{aligned} \tag{29}$$

where $\hat{\mathrm{E}}_{K^\varepsilon_{x_n}}\left[\theta_n(x_{n+1}) | x_n\right]$ denotes expectation under the probability measure in the $\varepsilon-$known set $K^\varepsilon_{x_n}$

$$K_x^\varepsilon \triangleq \left\{ y : y \in \mathcal{X} \text{ s.t. } \left| \frac{\hat{P}(y|x)}{\sum_{y' \in K_x^\varepsilon} \hat{P}(y'|x)} - \frac{P(y|x)}{\sum_{y' \in K_x^\varepsilon} P(y'|x)} \right| < \varepsilon \right\}. \tag{30}$$

The ODE for (29) can be written as

$$\frac{d\theta}{dt} = \Pi\left(r + \left(\gamma(P+\delta P) - I\right)\theta\right), \tag{31}$$

where

$$\delta P_{ij} = \begin{cases} \left(\sum_{k\in K_i^\varepsilon} P(k|i)\right)\left(\frac{\hat{P}(j|i)}{\sum_{k\in K_i^\varepsilon}\hat{P}(k|i)} - \frac{P(j|i)}{\sum_{k\in K_i^\varepsilon}P(k|i)}\right), & j \in K_i^\varepsilon, \\ 0, & j \notin K_i^\varepsilon. \end{cases} \tag{32}$$

Recalling that the true value function satisfies $\theta^* = (I-\gamma P)^{-1} r$, the asymptotic limit point of the ODE (31) is denoted by $\theta^* + \delta\theta$, and satisfies

$$\theta^* + \delta\theta = (I - \gamma(P+\delta P))^{-1} r. \tag{33}$$

In the next subsection we show how to bound the error term $\delta\theta$.

5.2.1 A BOUND ON THE BIAS

We would like to bound the term $\delta\theta$, which is the error in the value function, and can be seen as the total *bias* induced by the IPM method with the inaccurate model. Note that (33) describes a perturbed linear system (Horn and Johnson, 1985, §5). Using tools for dealing with such systems, we can bound the error as presented in the following theorem.

Theorem 10 *Let $K_{\max}^\varepsilon$ denote the cardinality of the largest $\varepsilon-$known set K_x^ε,*

$$K_{\max}^\varepsilon = \max_x |K_x^\varepsilon|, \tag{34}$$

and let ε satisfy $\varepsilon < \frac{1-\gamma}{\gamma K_{\max}^\varepsilon}$. Then the maximal error in the ODE limit is bounded by

$$\frac{\|\delta\theta\|_\infty}{\|\theta^*\|_\infty} \le \frac{\kappa}{1-\kappa\cdot\left(\frac{K_{\max}^\varepsilon\varepsilon}{1-\gamma}\right)}\left(\frac{K_{\max}^\varepsilon\varepsilon}{1-\gamma}\right), \tag{35}$$

where κ satisfies

$$\kappa \le \frac{1+\gamma}{1-\gamma}. \tag{36}$$

Proof For a matrix norm $\|\cdot\|_p$, if $\left\|(I-\gamma P)^{-1}\right\|_p \|\gamma\delta P\|_p < 1$ we have (Horn and Johnson, 1985, 5.8.8)

$$\frac{\|\delta\theta\|_p}{\|\theta^*\|_p} \le \frac{\kappa(I-\gamma P)}{1-\kappa(I-\gamma P)\left(\|\gamma\delta P\|_p / \|I-\gamma P\|_p\right)} \frac{\|\gamma\delta P\|_p}{\|I-\gamma P\|_p}, \tag{37}$$

where κ is the matrix condition number $\kappa(A) = \left\|A^{-1}\right\|_p \|A\|_p$. We now bound each of the terms on the right hand side of (37). We use the norm $\|\cdot\|_\infty$ which is induced by the max vector norm, and can be alternatively defined (Horn and Johnson, 1985, 5.6.5) as the *maximum row sum* matrix norm

$$\|\delta P\|_\infty \triangleq \max_i \sum_j \left|P_{ij}\right|. \tag{38}$$

From this definition, (32), (34), and (30), it is clear that

$$\|\gamma\delta P\|_\infty < \gamma K^\varepsilon_{\max}\varepsilon. \tag{39}$$

Theorem 5.6.9 in Horn and Johnson (1985) asserts that for any matrix norm we have

$$\|A\| \geq \rho(A),$$

where $\rho(A)$ is the spectral radius of A

$$\rho(A) \triangleq \max\{|\lambda| : \lambda \text{ is an eigenvalue of } A\}.$$

For the matrix $I-\gamma P$ we have

$$\rho(I-\gamma P) > 1-\gamma,$$

since P is stochastic and thus its largest eigenvalue is 1. Using (39) we therefore have

$$\frac{\|\gamma\delta P\|_\infty}{\|I-\gamma P\|_\infty} \leq \frac{\gamma K^\varepsilon_{\max}\varepsilon}{1-\gamma}.$$

We now bound $\kappa(I-\gamma P) = \|I-\gamma P\|_\infty \left\|(I-\gamma P)^{-1}\right\|_\infty$. First, by the triangle equality we have

$$\|I-\gamma P\|_\infty \leq \|I\|_\infty + \gamma\|P\|_\infty = 1+\gamma, \tag{40}$$

since P is a stochastic matrix and by definition (38) we have $\|P\|_\infty = 1$. Next we have by definition of the induced norm

$$\left\|(I-\gamma P)^{-1}\right\|_\infty = \max_{\|r\|_\infty=1} \left\|(I-\gamma P)^{-1} r\right\|_\infty \leq \frac{1}{1-\gamma}, \tag{41}$$

since $(I-\gamma P)^{-1} r$ can be seen as the value function associated with a reward vector r, which can have a maximum value of $r_{\max}/(1-\gamma)$. From (40) and (41) we have

$$\kappa(I-\gamma P) \leq \frac{1+\gamma}{1-\gamma}.$$

All is left is to verify that $\left\|(I-\gamma P)^{-1}\right\|_\infty \|\gamma\delta P\|_\infty < 1$. Using (39) and (41) this is satisfied if

$$\varepsilon < \frac{1-\gamma}{\gamma K^\varepsilon_{\max}}.$$

■

The bound in (35) can be simplified when ε is small, as described in the following corollary.

Corollary 11 *For small enough* ε *we have*

$$\frac{\|\delta\theta\|_\infty}{\|\theta^*\|_\infty} \leq \frac{K^\varepsilon_{\max}\varepsilon(1+\gamma)}{(1-\gamma)^2} + O\left(\varepsilon^2\right).$$

Proof Substitute (36) in (35), where for small ε we have $\varepsilon/\left(1-\frac{\kappa K^\varepsilon_{\max}\varepsilon}{1-\gamma}\right) = \varepsilon + O\left(\varepsilon^2\right)$. ■

The bounds in Theorem 10 and Corollary 11 show that the accumulated bias induced by the model inaccuracies is linear in ε, but also in $K^\varepsilon_{\max}$, thus it is preferred to keep the $\varepsilon-$known set for each state relatively small, as each element in the set contributes to an accumulated error. Note that the term $1/(1-\gamma)^2$ is a consequence of the fact that the estimation bias for each state now accumulates over a whole trajectory.

5.2.2 A BOUND ON THE VARIANCE

We shall now bound the variance of IPM-TD(0). We follow directly the analysis of Section 4 for IPM-TD(0) with an accurate model, and note that the only difference[9] is in the calculation for the term $\Sigma_0^{K^\varepsilon}$, which, using (33), becomes

$$\begin{aligned}\left[\Sigma_0^{K^\varepsilon}\right]_{xx} &= \mathrm{E}\left[(d_n^K)^2 \middle| \theta = \theta^* + \delta\theta, x_n = x\right] \\ &= \gamma^2 \mathrm{Var}\left[\left(\mathbf{1}_{n+1}^{K^\varepsilon} \hat{\mathrm{E}}_{K^\varepsilon_{x_n}}\left[\left[\theta^* + \delta\theta\right](x_{n+1}) | x_n\right] + \mathbf{1}_{n+1}^{\bar{K}^\varepsilon}\left[\theta^* + \delta\theta\right](x_{n+1})\right) \middle| x_n = x\right].\end{aligned}$$

In the following we focus on bounding the term on the right hand side, and show sufficient conditions under which $\left[\Sigma_0^{K^\varepsilon}\right]_{xx} < [\Sigma_0]_{xx}$ for every $x \in |\mathcal{X}|$. For notational simplicity, we drop the dependence on x, and treat a single random variable taking values in $\{[\theta^* + \delta\theta]_i\}_{i=1}^{|\mathcal{X}|}$, with the appropriate probabilities $P(x'|x)$. Thus, we need to bound

$$\gamma^{-2}\left[\Sigma_0^{K^\varepsilon}\right]_{xx} = \mathrm{Var}\left[\mathbf{1}^{K^\varepsilon}\hat{\mathrm{E}}_{K^\varepsilon}\left[\theta^* + \delta\theta\right] + \mathbf{1}^{\bar{K}^\varepsilon}\left(\theta^* + \delta\theta\right)\right], \tag{42}$$

and compare to $[\Sigma_0]_{xx}$. The following theorem, which is proved in Appendix D, bounds $\gamma^{-2}\left[\Sigma_0^{K^\varepsilon}\right]_{xx}$.

Theorem 12 *Let* $b_x \triangleq \sum_{x \in K_x^\varepsilon} |[\theta^*]_x|$, *and* $c_x \triangleq \left|E_{K_x^\varepsilon}[\theta^*] - E[\theta^*]\right|$, *and assume that* $\max\{\delta\theta\} \le \eta$. *Then the elements of the diagonal matrix* Σ_0^K *satisfy*

$$\begin{aligned}\gamma^{-2}\left[\Sigma_0^{K^\varepsilon}\right]_{xx} \le\ & \gamma^{-2}[\Sigma_0]_{xx} - P(K_x^\varepsilon)\cdot \mathrm{Var}_{K_x^\varepsilon}[\theta^*] + \eta^2 + 2\eta\gamma^{-1}\sqrt{[\Sigma_0]_{xx}} \\ & + \varepsilon P(K_x^\varepsilon)\left(\varepsilon b_x^2\left(1 - P(K_x^\varepsilon)\right) + 2b_x c_x\right).\end{aligned} \tag{43}$$

Using our previous bound on the bias we have the following corollary.

Corollary 13 *For small enough* ε *we have*

$$\begin{aligned}\gamma^{-2}\left([\Sigma_0]_{xx} - \left[\Sigma_0^{K^\varepsilon}\right]_{xx}\right) \ge\ & P(K_x^\varepsilon)\cdot \mathrm{Var}_{K_x^\varepsilon}[\theta^*] \\ & - \varepsilon\frac{K_{\max}^\varepsilon(1+\gamma)\|\theta^*\|_\infty}{(1-\gamma)^2}\left(\frac{K_{\max}^\varepsilon\varepsilon(1+\gamma)\|\theta^*\|_\infty}{(1-\gamma)^2} + \frac{2}{\gamma}\sqrt{[\Sigma_0]_{xx}}\right) \\ & - \varepsilon P(K_x^\varepsilon)\left(\varepsilon b_x^2\left(1 - P(K_x^\varepsilon)\right) + 2b_x c_x\right).\end{aligned}$$

Proof Apply Corollary 11 to bound η in Theorem 12. ■

The following corollary translates the previously established bounds to a performance improvement guarantee.

Corollary 14 *For a small enough* ε *an improvement in the asymptotic MSE of IPM-TD(0) can be guaranteed.*

9. Note that the $\Sigma_1^{K^\varepsilon}$ term is still zero, for the same reasons described in Section 4 .

Proof By Corollary 13 ε can be chosen such that for every x we have

$$\left[\Sigma_0^{K^\varepsilon}\right]_{xx} < [\Sigma_0]_{xx},$$

thus we can follow the development of the performance improvement proof in Section 4 and conclude that the sequence $(\theta_n^K - \theta^*)/\sqrt{\tilde{\varepsilon}}$ converges in distribution to a Gaussian centered on $\delta\theta$ and with a covariance $\hat{R}$, which is smaller than R (as defined in Section 4). Since we can use Theorem 10 to also bound the bias $\delta\theta$, we can choose ε small enough such that asymptotically $\mathrm{E}\|\theta_n^K - \theta^*\|^2 < \mathrm{E}\|\theta_n - \theta^*\|^2$. ■

To conclude, from Theorem 10 and Theorem 12 we see that for small enough ε, we get a small bias and a reduction in the variance. The specific terms in the bounds can be used to find a suitable ε that guarantees a performance improvement. If the probabilities $\hat{P}$ are obtained empirically from a trajectory of the MDP, then a Chernoff type bound can be used to further bound the number of observations required for a desired ε. This issue is beyond the scope of this paper.

6. Numerical Experiments

In this section, the performance improvement obtained by the IPM method is demonstrated on two different model free RL algorithms,[10] and two different problems. Our goal is to demonstrate both the *generality* of the method, and its *usefulness* in practical applications. Generality is demonstrated by application of the method to two very different RL algorithms - policy gradient and Q-learning. The only common feature to these two algorithms is their representation as a stochastic approximation. Together with the theory presented in previous sections, these results suggest that the IPM method can be applied succesfully to a wide variety of RL algorithms. The usefulness of the method is demonstrated in the solution of a call admission control problem. In this problem, it is shown that values of the partially known MDP (11) capture meaningful physical quantities of the problem, thus, (11) may be seen as the natural representation for partial knowledge in such problems. The performance improvement obtained by the IPM method suggests that it may be used succesfully in practice.

6.1 IPM Q-Learning for Admission Control

In this section we consider the call admission control problem for a single link, which arises when a telecommunication provider wants to sell the limited bandwidth of the link to different types of customers so as to maximize expected long term revenue. In this scenario, the customers differ in their bandwidth demand, the price they pay for its usage, and the frequency of their requests.

When the link is empty, it is reasonable that every customer request should be accepted, as it generates some revenue. On the other hand, when the link is almost full, a clever policy might decide to save the available bandwidth for the more profitable requests, at the expense of rejecting the less profitable ones. Thus, it is clear that a good policy should take into account both the bandwidth demand and profit of each request type, and its arrival frequency. When some of these quantities are not known in advance, a *learning* policy may be employed. Specifically, in the following we consider a case where these quantities are known only for *some* of the request types, which will naturally lead to the use of a partial model in the learning algorithm. Such a scenario can occur

10. The code for generating the results presented here can be found at the author's web site.

when, for example, new customers are added to a system, or when some of the request types have features that change in time.[11]

One approach to designing an admission policy is to formulate the problem as an MDP, for which an optimal policy is well defined, and solve it using RL approaches, as has been done by Marbach et al. (1998) and Marbach and Tsitsiklis (1998). In the following we present this approach, and show that in this problem our partially known MDP definition emerges as a very natural representation for partial model knowledge.

6.1.1 PROBLEM FORMULATION

Consider a service provider with a bandwidth of B units, which supports a finite set $\{1,2,...,M\}$ of different service types. Each service type is characterized by its bandwidth demand $b(m)$, its call arrival rate $\alpha(m)$, and its average holding time $1/\beta(m)$, where we assume that the calls arrive according to independent Poisson processes, and that the holding times are exponentially and independently distributed. Whenever a call of type m arrives, the controller can decide whether to accept or reject it, and if it is accepted and enough bandwidth is available, an immediate reward $c(m)$ is recieved. The objective is to find an admission controller (policy) that maximizes the average return. This problem can be represented by an MDP as follows.

6.1.2 STATE SPACE, CONTROLS, AND REWARD

The configuration of the link is denoted by $s = (s(1),\ldots,s(M))$, where $s(m) \in \{0,1,2,\ldots\}$ denotes the number of customers of type m currently using the link. Transitions between different configurations are triggered by events which we denote by $\omega = \{\omega(1),\ldots,\omega(M)\}$, where $\omega(m) = 1$ if a new customer of type m requests service, $\omega(m) = -1$ if a customer of type m departs from the system, and $\omega(m) = 0$ otherwise. The state x of the system consists of the link configuration together with the event which triggered a transition,

$$x = (s,\omega),$$

and the complete state space is given by

$$\mathcal{X} = \left\{ x = (s,\omega) | \sum s(m)\, b(m) \leq B,\ \sum |\omega(m)| \leq 1 \text{ and } \omega(m) < 0 \text{ only if } s(m) > 0 \right\}.$$

The possible controls are to accept or reject a call, denoted by $u \in \{u_a, u_r\}$, respectively, and the immediate reward is

$$r(x,u) = \begin{cases} c(m) & \text{if } u = u_a,\ \omega(m) = 1,\ (s+\omega,\omega) \in \mathcal{X}, \\ 0 & \text{else.} \end{cases}$$

The goal is to find the optimal policy with respect to the the *average reward*

$$\eta = \mathrm{E}[r(x)].$$

11. We note that a learning policy may be required even when the model is fully known, as finding the optimal policy is often an intractable problem. This 'fully known' scenario may be seen as a special case of the following presentation.

6.1.3 TRANSITION PROBABILITIES

In order to transform the continuous time process into discrete transitions between events, a uniformization technique (Gallager, 1995, §6.4) is used. Define z to be the maximal transition rate, given by

$$z = \max_{x \in \mathcal{X}} \left\{ \sum_{m=1}^{M} \left(\alpha(m) + s(m) \beta(m) \right) \right\}. \tag{44}$$

For a state $x = (s, \omega)$, the probability that the next event is an arrival of a call of type m is equal to $\alpha(m)/z$. The probability for a departure of a call of type m is $\beta(m) s(m)/z$. By normalization, the probability that in the next event nothing happens is $1 - \sum_{m=1}^{M} \left(\alpha(m) + s(m) \beta(m) \right)/z$.

6.1.4 PARTIALLY KNOWN MDP

For this problem, a natural definition for partial model knowledge is through the arrival and departure rates α, β, namely

$$M_K \triangleq \{ m : m \in 1, \ldots, M \text{ s.t. } \alpha(m), \beta(m) \text{ are known} \}.$$

As an example where such partial knowledge arises in practice, consider a case where new jobs (with unknown rates) are added to an existing system (with previously known rates). Note that generally, the values in M_K do not suffice for calculating z in (44), hence the transition probabilities of the MDP are not known. Nevertheless, the key point here is that in the *ratios* between transition probabilities, the z terms cancel out, therefore the partial MDP definition (11) can be satisfied. In particular, letting $i \in M_K$, we have

$$\frac{P(\text{arrival of type } i)}{\sum_{j \in M_K} P(\text{arrival of type } j)} = \frac{\alpha(i)}{\sum_{j \in M_K} \alpha(j)},$$

and similar expressions hold for probabilities of departures.

6.1.5 IPM Q-LEARNING

The model free RL algorithm we use for this problem is a variant of the popular Q-Learning algorithm for average return.[12] For each state-action pair, a Q value is maintained, and updated according to

$$Q_{n+1}(x_n, u_n) = Q_n(x_n, u_n) + \varepsilon_n \left(r(x_n, u_n) + \max_{u'} Q_n\left(x_{n+1}, u'\right) - Q_n(x_n, u_n) - \frac{1}{|\mathcal{X}||\mathcal{U}|} \sum_{x,u} Q_n(x, u) \right). \tag{45}$$

The greedy deterministic policy $u(x)$ w.r.t. the Q values at time n is

$$u(x) = \operatorname*{argmax}_{u'} Q_n\left(x, u'\right).$$

Update (45) is an SA, and was shown to converge (Abounadi et al., 2001) under suitable step sizes ε_n to a fixed point Q^*, such that the greedy policy w.r.t. Q^* is optimal. Applying the IPM method in

12. This is also known as relative value iteration.

Call Type m	1	2	3	4
$\alpha(m)$	1.8	1.4	1.6	1.4
$\beta(m)$	0.4	0.7	0.5	0.4
$b(m)$	1	1	1	1
$c(m)$	1.4	1	1.6	1

Table 1: Call Types

this case simply amounts to replacing $\max_{u'} Q_n(x_{n+1}, u')$ in (45) with

$$\mathbf{1}_{n+1}^{K} \cdot \frac{\sum_{y \in K_{x_n,u_n}} P_{u_n}(y|x_n) \max_{u'} Q_n(y,u')}{\sum_{y \in K_{x_n,u_n}} P_{u_n}(y|x_n)} + \mathbf{1}_{n+1}^{\bar{K}} \cdot \max_{u'} Q_n\left(x_{n+1}, u'\right). \tag{46}$$

We now report on the results of using IPM Q-learning for optimizing a call admission control policy.

6.1.6 RESULTS

In our experiments, we consider a link with a bandwidth of 7 units, and 4 call types. The parameters for each call are summarized in Table 1, and the size of the state space in this configuration is $|\mathcal{X}| = 2490$. IPM Q-Learning was run with initial values $Q_0(x,u) = r(x,u)$ and a step size $\varepsilon_n = \gamma_0 / (\gamma_1 + v_n(x_n,u_n))$, where $v_n(x,u)$ denotes the number of visits to the state action pair (x,u) up to time n. The values of γ_0, γ_1 were manually tuned for optimal performance, and set to $\gamma_0 = \gamma_1 = 40$. The action selection policy while learning was $\varepsilon-$greedy, with $\varepsilon = 0.1$. The partial model for each experiment is represented by a single parameter k, such that the arrival and departure rates of all calls of type $m \leq k$ are known. Figure 2 shows the average reward η as a function of iteration. As can be seen, incorporation of partial model knowledge by the IPM method resulted in a significant performance improvement.

6.2 IPM Policy Gradient

In this experiment simulations were performed on randomly generated MDP's, as described in Section 4.3. In the experiments, the agent maintains a stochastic policy function parametrized by $\theta \in \mathbb{R}^{L \cdot |\mathcal{U}|}$, and given by

$$\mu_\theta(u|x) = e^{\theta^T \xi(x,u)} / \sum_{u'} e^{\theta^T \xi(x,u')},$$

where the state-action feature vectors $\xi(x,u) \in \{0,1\}^{L \cdot |\mathcal{U}|}$ are constructed from the state features $\phi(x)$ defined in Section 4.3 as follows

$$\xi(x,u) \triangleq (0, \ldots (L \times (u-1)) \text{ zeros}), \phi(x), 0, \ldots (L \times (|\mathcal{U}| - u) \text{ zeros}))^T.$$

The agent's goal is to find the parameter θ which maximizes the *average reward* $\eta = \mathrm{E}[r(x)]$. Policy Gradient algorithms achieve this goal by estimating the gradient w.r.t. θ of the average reward, $\nabla_\theta \eta$, and performing a stochastic gradient ascent on the parameters to reach a local maximum. One such algorithm was proposed by Marbach and Tsitsiklis (1998). At time n we update the parameter vector θ and a scalar λ which is an estimate of η,

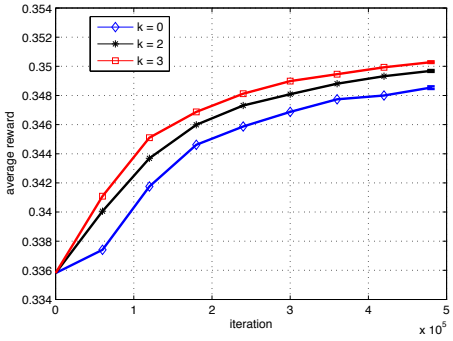


Figure 2: IPM Q-Learning for Admission control. Implementation of IPM Q-Learning, (45) and (46), for the call admission control problem of Table 1. Average reward of the greedy policy is plotted vs. iteration number for different values of k. Results are averaged over 100 different runs with the same initial conditions. Error bars display the standard error of the mean; for clarity of presentation the bars are displayed only for the last iteration.

$$\theta_{n+1} = \theta_n + \varepsilon\left(r\left(x_n\right) - \lambda_n\right) z_n, \tag{47}$$

$$\lambda_{n+1} = \lambda_n + \varepsilon'\left(r\left(x_n\right) - \lambda_n\right), \tag{48}$$

where ε and ε' are step sizes, and $\varepsilon' < \varepsilon$. We then simulate a transition to the next state, and update the vector z by

$$z_{n+1} = z_n + L_{x_n,u_n}\left(\theta_n\right),$$

where $L_{x_n,u_n}\left(\theta_n\right)$ is the likelihood ratio $L_{x,u}\left(\theta\right) = \nabla_\theta \log \mu_\theta(u|x)$. Every time a predefined recurrent state of the MDP is visited, z_{n+1} is reset to zero.

Denote by $\mathbf{1}_n^K$ an indicator function that equals 1 if x_n belongs to $K_{x_{n-1},u_{n-1}}$ and 0 otherwise. Incorporating partial knowledge into the algorithm using (12) simply amounts to replacing $r\left(x_n\right)$ in (47-48) with

$$\mathbf{1}_n^K \cdot \frac{\sum_{y \in K_{x_{n-1},u_{n-1}}} P_{u_{n-1}}(y|x_{n-1}) r(y)}{\sum_{y \in K_{x_{n-1},u_{n-1}}} P_{u_{n-1}}(y|x_{n-1})} + \mathbf{1}_n^{\bar{K}} \cdot r\left(x_n\right).$$

We simulated the policy gradient algorithm on a GARNET$(30,5,10,1)$ MDP. The state features were constructed as described in Section 4.3 with $L = 10$, $l = 2$. Figure 3 shows the average reward η as a function of iteration. These results indicate that the variance reduction in each iteration (guaranteed by Lemma 2) resulted, on average, in a better estimation of the gradient $\nabla_\theta \eta$, and therefore a better policy at each step.

7. Discussion and Future Work

Generally, when devising a solution to a difficult problem, one should incorporate into it all reliably available information. Model free RL algorithms typically operate without explicit knowledge of

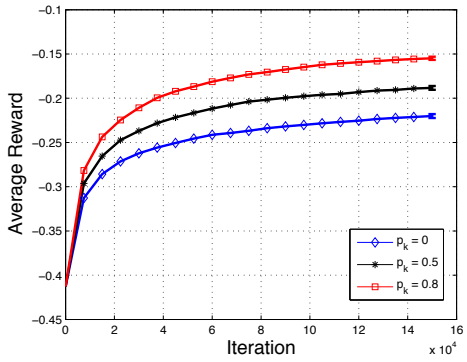


Figure 3: Policy Gradient with a Partial Model. Implementation of the algorithm described in Section 6.2 on a GARNET(30,5,10,1) MDP, with step size parameters $\varepsilon = 0.03$ and $\varepsilon' = 0.003$. The linear FA parameters are $L = 10$ and $l = 2$. Average reward is plotted vs. iteration number for different values of p_K. Results are averaged over 500 different runs with the same initial conditions. Error bars display the standard error of the mean; for clarity of presentation the bars are displayed only for the last iteration.

the underlying environment, and therefore, when such knowledge is available, using these algorithms 'out of the box' is clearly suboptimal. In this work we have presented a general method of integrating partial environmental knowledge into a large class of model free algorithms. Our method improves the asymptotic behavior of the algorithm, and at each iteration reduces the estimation variance due to the uncertainty in the environment. We have proved mathematically (for TD(0)) and demonstrated in simulation (for Policy Gradient and Q-learning) an improvement in the algorithm's overall performance.

From a more conceptual point of view, we have shown that two distinct approaches to RL, the model free and the model based approaches, can be combined in such a way that gains from their respective merits. From this perspective, this work is just a first step towards a theoretical understanding of the combination of different RL approaches.

A few issues are in need of further investigation.

In this work we have not addressed the question of how the partially known model can be acquired. A number of possibilities come to mind. In a transfer learning or tutor learning settings, the partial model can come from an expert who has exact knowledge of a model that is partially similar. In a multi-agent setting with communication, information about different parts of the model can be gathered independently by each of the agents, and combined to create a partial model of the environment.

An interesting possibility is to simultaneously gather information while adapting the policy using some model free algorithm. Using the SA algorithm (8), at the time of the n'th update of θ, we have already encountered a state-action trajectory of size n. Can we use this trajectory to construct an *estimated* partial MDP model, use it as in algorithm (12), and guarantee an improvement in the algorithm's performance? This should be done with caution, since using the same trajectory for updating the parameter and the estimated model may cause overfitting. To see this consider the following example. Let $\{x_i\}$ be a sequence of normally distributed i.i.d. random variables with

mean m, and assume that our goal is to estimate m. A natural approach is to use the empirical mean given by $\theta_n = \frac{1}{n}\sum_{i=1}^{n} x_i$, which can also be calculated recursively using the following SA iterate

$$\begin{aligned} \theta_{n+1} &= \theta_n + \frac{1}{n+1}(x_{n+1} - \theta_n), \\ \theta_0 &= 0. \end{aligned} \tag{49}$$

One may hope, that by the time of the n'th update of θ we could use the $n-1$ values of x_i already observed to build a partial model for x_n, and similarly to (12), use it to manipulate (49) in such a way that guarantees a performance improvement (in the estimation of m). However, it is known that for a normal distribution, the empirical mean is also the minimum variance unbiased estimator for m (Schervish, 1995). Our manipulation of (49) would therefore either add bias or increase the variance. Though this issue deserves careful analysis, we note that when a constant step size is used, the major influences on the current value of the parameter are the most recent measurements, thus older samples can be safely used to construct a partial model, mitigating the severity of this problem.

Finally, we note that the IPM method adds to the algorithm a computational cost of $O(K_{\max})$ evaluations of $F(\theta_n, x_n, u_n, x_{n+1})$ at each iteration. In our experiments, this cost proved to be negligible in comparison to the computational cost of the simulator. However, if the computation of $F(\theta_n, x_n, u_n, x_{n+1})$ is demanding, one may face a tradeoff between the performance of the resulting policy and the computational cost of obtaining it.

Acknowledgments

The authors would like to thank Nahum Shimkin for helpful discussions.

Appendix A. Admissibility of $\hat{\mu}_K$

In this section, based on the definitions of Section 2, we address the following issue. Can a better estimator than $\hat{\mu}_K(x)$ be found?

Since the MSE of any estimator, within a non-Bayesian setting, depends on the unknown μ, comparison of different estimators is a difficult task. A popular comparison framework is that of *admissible* estimators (Schervish, 1995). For a given known set K, an estimator is said to be admissible if there is no other estimator that achieves a smaller MSE for every distribution in $\mathcal{P}_K(\omega)$. Clearly, admissibility is a desirable property for an estimator, since an inadmissible estimator is guaranteed to be sub-optimal. The next theorem states that $\hat{\mu}_K$ is admissible.

Theorem 15 *The estimator $\hat{\mu}_K$ of* (5) *is admissible.*

Proof Let $\tilde{P}(\omega) \in \mathcal{P}_K(\omega)$ be defined as

$$\tilde{P}(\omega) = \frac{\mathbf{1}_\omega^K P(\omega)}{\sum_{\omega' \in K} P(\omega')}.$$

For $X \sim \tilde{P}(\omega)$ it is clear that $\hat{\mu}_K(x) = \mathrm{E}[X]$ for all x, therefore $\mathrm{E}[\hat{\mu}_K(X) - \mu]^2 = 0$, and no other estimator achieves a smaller MSE in this case. ■

Appendix B. Proof of Lemma 5

Proof By the ergodicity of the Markov chain the joint probability for subsequent states is

$$\lim_{n\to\infty} P(x_n, x_{n+1}) \quad = \quad P(x_{n+1}|x_n)\,[\pi]_{x_n}.$$

Now, observe that

$$\begin{aligned}
&\mathrm{E}\left[\left(d_n\phi(x_n)\right)\left(d_n\phi(x_n)\right)^T\middle|\,\theta_n=\theta^*,x_n\right]\\
&=\mathrm{Cov}\left[d_n\phi(x_n)|\,\theta_n=\theta^*,x_n\right]+\mathrm{E}\left[\left(d_n\phi(x_n)\right)|\,\theta_n=\theta^*,x_n\right]\mathrm{E}\left[\left(d_n\phi(x_n)\right)|\,\theta_n=\theta^*,x_n\right]^T\\
&=\gamma^2\phi(x_n)\,\mathrm{Cov}\left[\phi(x_{n+1})^T\theta^*\middle|\,x_n\right]\phi(x_n)^T\\
&\quad+\mathrm{E}\left[d_n\phi(x_n)|\,\theta_n=\theta^*,x_n\right]\mathrm{E}\left[d_n\phi(x_n)|\,\theta_n=\theta^*,x_n\right]^T,
\end{aligned}$$

where the second equality follows from

$$\mathrm{Cov}\left[d_n\phi(x_n)|\,\theta_n,x_n\right]=\mathrm{Cov}\left[d_n\phi(x_n)-r(x_n)+\phi(x_n)^T\theta_n\middle|\,\theta_n,x_n\right],$$

since adding constants does not change the covariance. Using Lemma 1 and Lemma 2 we derive an expression for the IPM iteration

$$\begin{aligned}
&\mathrm{E}\left[\left(d_n^K\phi(x_n)\right)\left(d_n^K\phi(x_n)\right)^T\middle|\,\theta_n=\theta^*,x_n\right]\\
&=\gamma^2\phi(x_n)\left(\mathrm{Cov}\left[\phi(x_{n+1})^T\theta^*\middle|\,x_n\right]-P(K_x)\,\mathrm{Cov}_K\left[\phi(x_{n+1})^T\theta^*\middle|\,x_n\right]\right)\phi(x_n)^T\\
&\quad+\mathrm{E}\left[d_n^K\phi(x_n)|\,\theta_n=\theta^*,x_n\right]\mathrm{E}\left[d_n^K\phi(x_n)|\,\theta_n=\theta^*,x_n\right]^T\\
&=\mathrm{E}\left[\left(d_n\phi(x_n)\right)\left(d_n\phi(x_n)\right)^T\middle|\,\theta_n=\theta^*,x_n\right]-\gamma^2\phi(x_n)\,P(K_x)\,\mathrm{Cov}_K\left[\left(\phi(x_{n+1})^T\theta^*\right)\middle|\,x_n\right]\phi(x_n)^T.
\end{aligned}$$

We therefore have that

$$\begin{aligned}
\Sigma_0 &= \lim_{n\to\infty}\mathrm{E}\left[\left(d_n\phi(x_n)\right)\left(d_n\phi(x_n)\right)^T\middle|\,\theta_n=\theta^*\right]\\
&= \lim_{n\to\infty}\mathrm{E}\left[\mathrm{E}\left[\left(d_n\phi(x_n)\right)\left(d_n\phi(x_n)\right)^T\middle|\,\theta_n=\theta^*,x_n\right]\right]\\
&= \sum_x [\pi]_x\,\mathrm{E}\left[\left(d_n\phi(x_n)\right)\left(d_n\phi(x_n)\right)^T\middle|\,\theta_n=\theta^*,x_n\right]\\
&= \Sigma_0^K+\gamma^2\sum_x [\pi]_x\,\phi(x)\,P(K_x)\,\mathrm{Cov}_K\left[\phi(x')^T\theta^*\middle|\,x\right]\phi(x)^T\\
&= \Sigma_0^K+\gamma^2\sum_x [\pi]_x\,\phi(x)\,P(K_x)\,\mathrm{Var}_K\left[\phi(x')^T\theta^*\middle|\,x\right]\phi(x)^T.
\end{aligned}$$

■

Appendix C. Proof of Theorem 9

Proof First, observe that

$$\left|\hat{\mathrm{E}}_{K^\varepsilon}[X]-\mathrm{E}_{K^\varepsilon}[X]\right|=\left|\sum_{x\in K^\varepsilon} x\left(\hat{P}_{K^\varepsilon}(x)-P_{K^\varepsilon}(x)\right)\right|\le\varepsilon\sum_{x\in K^\varepsilon}|x|. \tag{50}$$

Using (50) a simple bound on the bias of $\hat{\mu}_{K^\varepsilon}$ is derived

$$\begin{aligned}\left|\mathrm{E}\left[\hat{\mu}_{K^\varepsilon}(X)\right]-\mathrm{E}[X]\right| &= \left|\sum_{x\in K^\varepsilon} P(x)\left(\hat{\mathrm{E}}_{K^\varepsilon}[X]-x\right)\right| \\ &= P(K^\varepsilon)\left|\hat{\mathrm{E}}_{K^\varepsilon}[X]-\mathrm{E}_{K^\varepsilon}[X]\right| \leq \varepsilon P(K^\varepsilon)\sum_{x\in K^\varepsilon}|x|.\end{aligned}$$

We now derive a bound on the MSE of $\hat{\mu}_{K^\varepsilon}$.

$$\begin{aligned}\mathrm{E}\left[\left(\hat{\mu}_{K^\varepsilon}(X)-\mathrm{E}[X]\right)^2\right] &= \sum_{x\in K^\varepsilon} P(x)\left(\hat{\mathrm{E}}_{K^\varepsilon}[X]-\mathrm{E}[X]\right)^2+\sum_{x\in\bar{K}^\varepsilon} P(x)\left(x-\mathrm{E}[X]\right)^2 \\ &= P(K^\varepsilon)\left(\hat{\mathrm{E}}_{K^\varepsilon}[X]-\mathrm{E}_{K^\varepsilon}[X]+\mathrm{E}_{K^\varepsilon}[X]-\mathrm{E}[X]\right)^2 \\ &\quad +\sum_{x\in\bar{K}^\varepsilon} P(x)\left(x-\mathrm{E}[X]\right)^2 \\ &\leq \mathrm{Var}[X]-P(K^\varepsilon)\cdot\mathrm{Var}_{K^\varepsilon}[X] \\ &\quad +P(K^\varepsilon)\left(\left(\varepsilon\sum_{x\in K^\varepsilon}|x|\right)^2+2\varepsilon\left(\sum_{x\in K^\varepsilon}|x|\right)\left|\mathrm{E}_{K^\varepsilon}[X]-\mathrm{E}[X]\right|\right)\end{aligned}$$

where in the inequality in the third line we used (50) and Lemma 2. We see that we have an improvement in MSE terms if $\left(\varepsilon\sum_{x\in K^\varepsilon}|x|\right)^2+2\varepsilon\left(\sum_{x\in K^\varepsilon}|x|\right)\left|\mathrm{E}_{K^\varepsilon}[X]-\mathrm{E}[X]\right| < \mathrm{Var}_{K^\varepsilon}[X]$, which is always satisfied as $\varepsilon\to 0$. Similarly, for the variance we have

$$\begin{aligned}\mathrm{E}\left[\left(\hat{\mu}_{K^\varepsilon}(X)-\mathrm{E}[\hat{\mu}_{K^\varepsilon}(X)]\right)^2\right] &= \mathrm{E}\left[\left(\hat{\mu}_{K^\varepsilon}(X)-\mathrm{E}[X]\right)^2\right]-\left(\mathrm{E}[\hat{\mu}_{K^\varepsilon}(X)]-\mathrm{E}[X]\right)^2 \\ &\leq \mathrm{Var}[X]-P(K^\varepsilon)\cdot\mathrm{Var}_{K^\varepsilon}[X] \\ &\quad +P(K^\varepsilon)\left(\left(\varepsilon\sum_{x\in K^\varepsilon}|x|\right)^2+2\varepsilon\left(\sum_{x\in K^\varepsilon}|x|\right)\left|\mathrm{E}_{K^\varepsilon}[X]-\mathrm{E}[X]\right|\right).\end{aligned}$$

■

Appendix D. Proof of Theorem 12

Note that without the $\delta\theta$ terms in (42), the bound on the variance for a random variable (28) could be used to compare $[\Sigma_0^K]_{xx}$ with $[\Sigma_0]_{xx}$. For small $\delta\theta$, the difference in the variance should be small, as is proved in the following Lemma, which we first motivate with a simple example.

Let $X\in\{1,2\}$ and $X'\in\{1+\eta,2+\eta\}$ be two random variables satisfying $P(X=1)=P(X'=1+\eta)$ and $P(X=2)=P(X'=2+\eta)$. We expect that for small η, the difference between $\mathrm{Var}[X]$ and $\mathrm{Var}[X']$ should also be small.

Lemma 16 *Let $X\sim P(\cdot)$ be a random variable over a finite set $\{\Omega_i\}_{i=1}^N$, where $\Omega_i\in\mathbb{R}$. Let $X'\sim P(\cdot)$ be a random variable over a finite set $\{\Omega_i'\}_{i=1}^N$, such that $|\Omega_i-\Omega_i'|<\eta$, $i=1,\ldots,N$. The variance of X' satisfies*

$$\mathrm{Var}\left[X'\right]\leq\mathrm{Var}[X]+\eta^2+2\eta\sqrt{\mathrm{Var}[X]}.$$

Proof First we have

$$\mathrm{Var}\left[X'\right] = \mathrm{E}\left[\left(X' - \mathrm{E}\left[X'\right]\right)^2\right] \leq \mathrm{E}\left[\left(X' - \mathrm{E}\left[X\right]\right)^2\right],$$

since

$$\mathrm{E}\left[\left(X' - \mathrm{E}\left[X\right]\right)^2\right] = \mathrm{Var}\left[X'\right] + \left(\mathrm{E}\left[\left(X' - \mathrm{E}\left[X\right]\right)\right]\right)^2.$$

Next we have

$$\mathrm{E}\left[\left(X' - \mathrm{E}\left[X\right]\right)^2\right] = \sum_{x'} P\left(x'\right)\left(x' - \mathrm{E}\left[X\right]\right)^2,$$

but since $|\Omega_i - \Omega_i'| < \eta,\ \forall i$ then by the triangle equality we have $|x' - \mathrm{E}[X]| \leq |x - \mathrm{E}[X]| + \eta$, so we have

$$\begin{aligned}
\mathrm{E}\left[\left(X' - \mathrm{E}\left[X\right]\right)^2\right] &\leq \sum_x P\left(x\right)\left(|x - \mathrm{E}\left[X\right]| + \eta\right)^2 \\
&= \mathrm{Var}\left[X\right] + \eta^2 + 2\eta \sum_x P\left(x\right)|x - \mathrm{E}\left[X\right]| \\
&\leq \mathrm{Var}\left[X\right] + \eta^2 + 2\eta\sqrt{\mathrm{Var}\left[X\right]},
\end{aligned}$$

where in the last inequality we used Cauchy–Schwartz under the expectation inner product:

$$\left\langle |x - \mathrm{E}\left[X\right]|, 1\right\rangle^2 \leq \left\langle |x - \mathrm{E}\left[X\right]|, |x - \mathrm{E}\left[X\right]|\right\rangle \left\langle 1, 1\right\rangle = \mathrm{Var}\left[X\right].$$

■

We now combine the result of Lemma 16 and the bound on the variance developed for the random variable (28) to prove Theorem 12.

Proof (of Theorem 12) First, we use Lemma 16 to bound (42)

$$\begin{aligned}
\mathrm{Var}\left[\mathbf{1}^{\kappa^\varepsilon}\hat{\mathrm{E}}_{\kappa^\varepsilon}\left[\theta^* + \delta\theta\right] + \mathbf{1}^{\bar{\kappa}^\varepsilon}\left(\theta^* + \delta\theta\right)\right] &\leq \mathrm{Var}\left[\mathbf{1}^{\kappa^\varepsilon}\hat{\mathrm{E}}_{\kappa^\varepsilon}\left[\theta^*\right] + \mathbf{1}^{\bar{\kappa}^\varepsilon}\left(\theta^*\right)\right] \\
&\quad + \eta^2 + \frac{2\eta}{\gamma}\sqrt{\mathrm{Var}\left[\theta^*\right]},
\end{aligned}$$

and substitute $[\Sigma_0]_{xx} = \gamma^2 \mathrm{Var}[\theta^*]$. We now use (28) to bound $\mathrm{Var}\left[\mathbf{1}^{\kappa^\varepsilon}\hat{\mathrm{E}}_{\kappa^\varepsilon}\left[\theta^*\right] + \mathbf{1}^{\bar{\kappa}^\varepsilon}\left(\theta^*\right)\right]$, which results in (43). ■

Appendix E. Proof of Theorem 4

As stated before, Theorem 4 is a consequence of Theorem 10.1.3 in Kushner and Yin (2003), for which a long set of assumptions is required. For the sake of clarity, this section is organized as follows. We first present a constrained version of the IPM-TD(0) algorithm and show that it converges weakly to a unique point. We then present some definitions needed for Theorem 10.1.3 in Kushner and Yin (2003), followed by an explicit statement of the theorem, without the required assumptions. Finally, we present the assumptions one by one, and prove that they are indeed fulfilled.

E.1 Constrained Algorithm

An important issue in the analysis of an SA algorithm (8) is the boundedness of the iterates. For many convergence results, a required condition is that the sequence θ_n be bounded almost surely. This condition is not trivial, and there are several approaches to satisfying it. One simple approach is to truncate the iterate θ_n when it leaves some prespecified constraint set denoted by H (Kushner and Yin, 2003). This will be done by introducing a 'correction' term Z_n

$$\theta_{n+1} = \theta_n + \varepsilon F(\theta_n, x_n, u_n, x_{n+1}) + \varepsilon Z_n, \tag{51}$$

where εZ_n is the vector of shortest Euclidean length needed to take $\theta_n + \varepsilon F(\theta_n, x_n, u_n, x_{n+1})$ back to the constraint set H if it is not in H. Respectively, the correction term needs to be added to the associated ODE

$$\frac{d\theta}{dt} = \bar{g}(\theta) + z_t, \tag{52}$$

where z_t maintains θ in H. Recall the unconstrained ODE for TD(0) (15), and its fixed point θ^*. Since in TD(0) θ^* is bounded by the maximal value of the MDP $r_{\max}/(1-\gamma)$, we can choose H to be large enough such that $\theta^* \in H$. The following Lemma guarantees that in this case, the additional z_t term in (52) does not change the ODE's unique fixed point.

Lemma 17 *Assuming $\theta^* \in H$, the constrained ODE for IPM-TD(0) with linear function approximation $d\theta/dt = b + A\theta + z_t$, with b, A, z_t defined in Section 4.1, has a unique and asymptotically stable fixed point θ^*, which satisfies $A\theta^* = -b$.*

Proof Consider as a Lyapunov function the Euclidean distance to θ^*, $\mathcal{V}(\theta) = (\theta - \theta^*)^T (\theta - \theta^*)$. For the unconstrained ODE (10), we have[13] $\dot{\mathcal{V}}(\theta) = \theta^T \left(A + A^T\right) \theta$, and since A is negative definite we have $\dot{\mathcal{V}}(\theta) < 0$, and $\mathcal{V}(\theta)$ is a valid Lyapunov function. Let $\dot{\mathcal{V}}_H(\theta)$ correspond to the constrained ODE. Since θ^* is in H, by the geometric definition of the correction terms, we have $\dot{\mathcal{V}}_H(\theta) \leq \dot{\mathcal{V}}(\theta) < 0$, therefore $\mathcal{V}$ is also valid for the constrained ODE (52). ■

We now state a convergence theorem that relates between the limit point of the ODE (52) and the asymptotic behavior of algorithm (51). The assumptions needed for this theorem are satisfied by default, by the definition of our RL environment and algorithm, and are thus omitted.

Let $\theta(t)$ denote the piece-wise constant continuous time interpolation θ_n, where $\theta(t) = \theta_n$ on the time interval $[n\varepsilon, n\varepsilon + \varepsilon)$ for $t \geq 0$ and $\theta(t) = \theta_0$ for $t < 0$. Similarly define $Z(t)$ as the piece-wise constant continuous time interpolation of Z_n.

Theorem 18 *(Theorem 8.2.2 in Kushner and Yin, 2003) Consider algorithm* (51)*. For any non decreasing sequence of integers q_ε, for each sub-sequence of $\{\theta(\varepsilon q_\varepsilon + \cdot), Z(\varepsilon q_\varepsilon + \cdot)\}, \varepsilon > 0$, there exist a further sub-sequence and a process $(\theta(\cdot), Z(\cdot))$ such that*

$$(\theta(\varepsilon q_\varepsilon + \cdot), Z(\varepsilon q_\varepsilon + \cdot)) \Rightarrow (\theta(\cdot), Z(\cdot))$$

as $\varepsilon \to 0$ through the convergent sub-sequence, where

$$\theta(t) = \theta(0) + \int_0^t \bar{g}(\theta(s))\, ds + Z(t). \tag{53}$$

Let $\varepsilon q_\varepsilon \to \infty$ as $\varepsilon \to 0$. Then, for almost all ω, the path $\theta(\omega, \cdot)$ lies in a limit set of (53)*.*

13. See derivation in the proof of **20.3.3.**

E.2 Definitions

The following technical definitions are required for the convergence result.

Let $D^L(-\infty,\infty)$ (and $D^L[0,\infty)$, respectively) denote the $L-$fold product space of real valued functions on the interval $(-\infty,\infty)$ (resp. on $[0,\infty)$) that are right continuous and have left-hand limits, with the Skorohod topology used.[14]

Let $\{q_\varepsilon\}$ be a sequence of non-negative integers. In order to investigate the asymptotic behavior we will examine $\theta(\varepsilon q_\varepsilon + \cdot)$, where $\varepsilon q_\varepsilon \to \infty$. We also demand $\varepsilon(q_\varepsilon - p_\varepsilon) \to \infty$ where p_ε are non decreasing and non-negative integers used in assumption 20.3.

Define the normalized error process

$$U_n = (\theta_{q_\varepsilon+n} - \theta^*)/\sqrt{\varepsilon},$$

and let $U^\varepsilon(\cdot)$ denote its piecewise constant right continuous interpolation, with interpolation intervals ε, on $[0,\infty)$. Define $W^\varepsilon(\cdot)$ on $(-\infty,\infty)$ by

$$W^\varepsilon(t) = \begin{cases} \sqrt{\varepsilon}\sum_{i=q_\varepsilon}^{q_\varepsilon+t/\varepsilon-1} F(\theta^*,x_i,u_i,x_{i+1}), & t \geq 0 \\ -\sqrt{\varepsilon}\sum_{i=q_\varepsilon}^{q_\varepsilon+t/\varepsilon-1} F(\theta^*,x_i,u_i,x_{i+1}), & t < 0 \end{cases} \tag{54}$$

E.3 A Theorem on Fluctuations in SA

Theorem 19 *(10.1.3 in Kushner and Yin, 2003) Consider algorithm* (51) *and let assumption 20 hold. Then the sequence* $\{U^\varepsilon(\cdot), W^\varepsilon(\cdot)\}$ *converges weakly in* $D^L[0,\infty) \times D^L(-\infty,\infty)$ *to a limit denoted by* $\{U(\cdot), W(\cdot)\}$*, and*

$$dU = AU\,dt + dW,$$

where the matrix A is defined in 20.8, $W(\cdot)$ *is a Wiener process with covariance matrix* Σ *described in 20.5, and* $U(\cdot)$ *is stationary.*

Theorem 4 is a direct consequence of Theorem 19, with $n = \omega(1/\varepsilon)$ satisfying the requirement on q_ε.

E.4 Assumptions for Theorem 19

The set of assumptions 20 which we describe in the following is designed to fit a wide variety of algorithms, and are thus quite complicated. The IPM-TD(0) algorithm with which we are concerned is a very simple case of this theorem, as it is linear, bounded, and stationary, and the Markovian state transitions are ergodic, and defined over a finite state space.[15] Moreover, many of the assumptions that follow are used in order to reduce a more complicated algorithm to these simpler settings, and to show that the residual that remains is small in some sense. Thus, many complicated terms in the assumptions just vanish, and some assumptions are true by default.

14. See Kushner and Yin (2003, p. 228, 238) for more details on D^L.
15. In Borkar (2008) a simpler result regarding fluctuations with a fixed step size is given, albeit for a martingale difference noise scenario. The Markovian state dependent noise in our case requires the more complicated approach of Kushner and Yin (2003).

Assumption 20 *The following holds*[16]

1. $\left\{F\left(\theta_n, x_n, u_n, x_{n+1}\right) I_{\{|\theta_n - \theta^*| \leq \rho\}}\right\}$ is uniformly integrable for small $\rho > 0$.

   Proof $F\left(\theta_n, x_n, u_n, x_{n+1}\right)$ is uniformly integrable since on every sample path θ_n is bounded (by the constraint), $r\left(x_n\right)$ is bounded by r_{max} and $\phi\left(x_n\right)$ is also bounded by definition. Since this is true for every sample path, $F\left(\theta_n, x_n, u_n, x_{n+1}\right) I_{\{|\theta_n - \theta^*| \leq \rho\}}$ is uniformly integrable for all ρ. ■

2. There is a sequence of non-negative and non decreasing integers N_ε such that $\theta\left(\varepsilon N_\varepsilon + \cdot\right)$ converges weakly to the process with constant value θ^* strictly inside the constraint set.

   Proof By the weak convergence Theorem 18, choosing N_ε such that $\varepsilon N_\varepsilon \to \infty$, and by Lemma 17, we have that IPM-TD(0) converges weakly to the process with constant value θ^* strictly inside the constraint set. ■

3. There are non decreasing and non-negative integers p_ε (that can be taken to be greater than N_ε) such that

   $$\left\{\left(\theta_{p_\varepsilon + n} - \theta^*\right) / \sqrt{\varepsilon}; \varepsilon > 0, n \geq 0\right\}$$

   is tight.

   Proof For the proof of this assumption we use Theorem 10.5.2 in Kushner and Yin (2003), which we now state.

   Theorem 21 *(10.5.2 in Kushner and Yin, 2003) Assume the constrained algorithm 51 with constraint set H, where θ^* is in the interior of H. Assume that 20.2 and 20.3.1-20.3.7 hold in H. Then there are $p_\varepsilon < \infty$ such that $\left\{\left(\theta_{p_\varepsilon + n} - \theta^*\right) / \sqrt{\varepsilon}; \varepsilon > 0, n \geq 0\right\}$ is tight.*

   ■

3.1 θ^* is a globally asymptotically stable (in the sense of Lyapunov) point of the ODE $d\theta/dt = \bar{g}\left(\theta\right) + z_t$.

   Proof This is satisfied by Lemma 17. ■

3.2 The non-negative and continuously differentiable function $V\left(\cdot\right)$ is a Lyapunov function for the ODE. The second order partial derivatives are bounded and $\left|\nabla_\theta V\left(\theta\right)\right|^2 \leq K_1\left(V\left(\theta\right) + 1\right)$, where K_1 is an arbitrary positive number.

   Proof Choose V to be of the form $V\left(\theta\right) = \left(\theta - \theta^*\right)^T \left(\theta - \theta^*\right)$. As was in the proof of Lemma 17, V is a valid Lyapunov function. The second order partial derivatives are zero, and

   $$\nabla_\theta V\left(\theta\right) = 2\left(\theta - \theta^*\right)$$

16. We exclude assumptions which are true by definition of our RL settings.

$$|\nabla_\theta V(\theta)|^2 = 4|\theta-\theta^*|^2 = 4V(\theta).$$

■

3.3 There is a $\lambda > 0$ such that $V_\theta^T(\theta)\bar{g}(\theta) \le -\lambda V(\theta)$

Proof Recalling from (15) and (16) that $\bar{g}(\theta) = b + A\theta$, we have

$$\begin{aligned}
\frac{1}{2}\left((\nabla_\theta V(\theta))^T \bar{g}(\theta) + \left((\nabla_\theta V(\theta))^T \bar{g}(\theta)\right)^T\right) &= (\theta-\theta^*)^T(b+A\theta) + (b+A\theta)^T(\theta-\theta^*) \\
&= (\theta-\theta^*)^T b + b^T(\theta-\theta^*) \\
&\quad + (\theta-\theta^*)^T A\theta + \theta^T A^T(\theta-\theta^*) \\
&= (\theta-\theta^*)^T b + b^T(\theta-\theta^*) \\
&\quad + (\theta-\theta^*)^T\left(A+A^T\right)(\theta-\theta^*) \\
&\quad + (\theta-\theta^*)^T A\theta^* + \theta^{*T} A^T(\theta-\theta^*) \\
&= (\theta-\theta^*)^T(b+A\theta^*) + (b+A\theta^*)^T(\theta-\theta^*) \\
&\quad + (\theta-\theta^*)^T\left(A+A^T\right)(\theta-\theta^*) \\
&= (\theta-\theta^*)^T\left(A+A^T\right)(\theta-\theta^*).
\end{aligned}$$

Let λ' denote the largest eigenvalue of $A+A^T$. We have that $(\theta-\theta^*)^T\left(A+A^T\right)(\theta-\theta^*) \le \lambda'(\theta-\theta^*)^T(\theta-\theta^*)$, and

$$(\nabla_\theta V(\theta))^T \bar{g}(\theta) = (\theta-\theta^*)^T\left(A+A^T\right)(\theta-\theta^*) \le \lambda' V(\theta).$$

■

3.4 For each $K > 0$, $\sup_n \mathrm{E}|F(\theta_n, x_n, u_n, x_{n+1})|^2 I_{\{|\theta_n-\theta^*|\le K\}} \le K_1 \mathrm{E}[V(\theta_n)+1]$, where K_1 does not depend on K.

Proof

Satisfying this requirement is immediate, since $F(\theta_n, x_n, u_n, x_{n+1})$ is bounded on every sample path. This follows from the fact that on every sample path θ_n is bounded (by the constraint), $r(x_n)$ is bounded by r_{max} and $\phi(x_n)$ is also bounded by definition. ■

3.5 The sum $\Gamma_n(\theta) = \varepsilon\sum_{i=n}^{\infty}(1-\varepsilon)^{i-n}\mathrm{E}_n[g(\theta, x_i, u_i) - \bar{g}(\theta)]$, where E_n denotes expectation conditioned on the history up to time n, is well defined in that the sum of the norms of the summands is integrable for each θ, and $\mathrm{E}|\Gamma_n(\theta_n)|^2 = O(\varepsilon^2)$.

Proof From Lemma 6.7 in Bertsekas and Tsitsiklis (1996) (which relies on the exponential mixing time of Markov chains) we have that $|\mathrm{E}_n[g(\theta, x_i, u_i) - \bar{g}(\theta)]| \le c\rho^{i-n}|\theta|$ for some

$c > 0$ and $\rho < 1$. This gives

$$\begin{aligned}\left|\sum_{i=n}^{\infty}(1-\varepsilon)^{i-n}\mathrm{E}_n\left[g\left(\theta,x_i,u_i\right)-\bar{g}\left(\theta\right)\right]\right| &\leq \sum_{i=n}^{\infty}\left|\mathrm{E}_n\left[g\left(\theta,x_i,u_i\right)-\bar{g}\left(\theta\right)\right]\right| \\ &\leq \sum_{i=n}^{\infty}c\rho^{i-n}\left|\theta\right| \\ &= \frac{c\left|\theta\right|}{1-\rho},\end{aligned}$$

and

$$\begin{aligned}\mathrm{E}\left|\Gamma_n\left(\theta_n\right)\right|^2 &\leq \varepsilon^2\mathrm{E}\left|\frac{c\left|\theta_n\right|}{1-\rho}\right|^2 \\ &= \frac{\varepsilon^2 c}{1-\rho}\left|\theta^*\right|^2 \\ &= O\left(\varepsilon^2\right).\end{aligned}$$

■

3.6 $\mathrm{E}\left|\Gamma_{n+1}\left(\theta_{n+1}\right)-\Gamma_{n+1}\left(\theta_n\right)\right|^2 = O\left(\varepsilon^2\right)$.

Proof

We have

$$\begin{aligned}&\Gamma_{n+1}\left(\theta_{n+1}\right)-\Gamma_{n+1}\left(\theta_n\right) \\ = \;& \varepsilon\sum_{i=n+1}^{\infty}(1-\varepsilon)^{i-n-1}\mathrm{E}_{n+1}\left[g\left(\theta_{n+1},x_i,u_i\right)-\bar{g}\left(\theta_{n+1}\right)\right] \\ & -\varepsilon\sum_{i=n+1}^{\infty}(1-\varepsilon)^{i-n-1}\mathrm{E}_{n+1}\left[g\left(\theta_n,x_i,u_i\right)-\bar{g}\left(\theta_n\right)\right] \\ = \;& \varepsilon\sum_{i=n+1}^{\infty}(1-\varepsilon)^{i-n-1}\mathrm{E}_{n+1}\left[g\left(\theta_{n+1},x_i,u_i\right)-g\left(\theta_n,x_i,u_i\right)-\left(\bar{g}\left(\theta_{n+1}\right)-\bar{g}\left(\theta_n\right)\right)\right].\end{aligned}$$

Using the triangle inequality

$$\begin{aligned}&\left|\Gamma_{n+1}(\theta_{n+1})-\Gamma_{n+1}(\theta_n)\right| \leq \\ &\varepsilon\sum_{i=n+1}^{\infty}(1-\varepsilon)^{i-n-1}\left|\mathrm{E}_{n+1}[g\left(\theta_{n+1},x_i,u_i\right)-g\left(\theta_n,x_i,u_i\right)-\left(\bar{g}\left(\theta_{n+1}\right)-\bar{g}\left(\theta_n\right)\right)]\right|\end{aligned}$$

By the linearity of $g,\bar{g}$, and since θ_n is bounded, we have that$\left|g\left(\theta_{n+1},x_i,u_i\right)-g\left(\theta_n,x_i,u_i\right)\right| < k\varepsilon$ for some k, and $\left|\bar{g}\left(\theta_{n+1}\right)-\bar{g}\left(\theta_n\right)\right| < \bar{k}\varepsilon$ for some $\bar{k}$. We therefore have :

$$\left|\mathrm{E}_{n+1}\left[g\left(\theta_{n+1},x_i,u_i\right)-g\left(\theta_n,x_i,u_i\right)\right]\right| \leq \mathrm{E}_{n+1}\left[\left|g\left(\theta_{n+1},x_i,u_i\right)-g\left(\theta_n,x_i,u_i\right)\right|\right] < k\varepsilon,$$

and similarly

$$\left|\mathrm{E}_{n+1}\left[\bar{g}\left(\theta_{n+1}\right)-\bar{g}\left(\theta_{n}\right)\right]\right|<\bar{k}\varepsilon.$$

Now

$$\begin{aligned}\left|\Gamma_{n+1}\left(\theta_{n+1}\right)-\Gamma_{n+1}\left(\theta_{n}\right)\right| &\leq \varepsilon^{2}\left(k+\bar{k}\right)\sum_{i=n+1}^{\infty}\left(1-\varepsilon\right)^{i-n-1}\\ &= \varepsilon\left(k+\bar{k}\right),\end{aligned}$$

and

$$\begin{aligned}\left|\Gamma_{n+1}\left(\theta_{n+1}\right)-\Gamma_{n+1}\left(\theta_{n}\right)\right|^{2} &\leq \varepsilon^{2}\left|k+\bar{k}\right|^{2}\\ &= O\left(\varepsilon^{2}\right).\end{aligned}$$

■

3.7 Let θ^{H} denote the projection of θ onto H. Then for all θ, $V\left(\theta^{H}\right)\leq V\left(\theta\right)$.

Proof This assumption was shown to hold in the proof of Lemma 17. ■

4. For a small $\rho>0$, and any sequence $\varepsilon\to\infty$ and $n\to\infty$ such that $\theta_{n}\to\theta^{*}$ in probability,

$$\mathrm{E}\left[\left|\delta M_{n}-\delta M_{n}\left(\theta^{*}\right)\right|^{2}I_{\{|\theta_{n}-\theta^{*}|\leq\rho\}}\right]\to 0.$$

Proof Recall that we have

$$\begin{aligned}\delta M_{n} &= \mathrm{E}\left[d_{n}^{K}\phi\left(x_{n}\right)|x_{n},u_{n}\right]-d_{n}^{K}\phi\left(x_{n}\right)\\ &= \gamma\sum_{y}P_{u_{n}}\left(y|x_{n}\right)\phi(y)^{T}\theta_{n}\phi\left(x_{n}\right)\\ &\quad -\gamma\left(\mathbf{1}_{n+1}^{K}\frac{\sum_{y\in K_{x_{n},u_{n}}}P_{u_{n}}\left(y|x_{n}\right)\phi(y)^{T}}{\sum_{y\in K_{x_{n},u_{n}}}P_{u_{n}}\left(y|x_{n}\right)}+\mathbf{1}_{n+1}^{\bar{K}}\phi(x_{n+1})^{T}\right)\theta_{n}\phi\left(x_{n}\right)\\ &= \gamma\left(\sum_{y}P_{u_{n}}\left(y|x_{n}\right)\phi(y)^{T}-\mathbf{1}_{n+1}^{K}\frac{\sum_{y\in K_{x_{n},u_{n}}}P_{u_{n}}\left(y|x_{n}\right)\phi(y)^{T}}{\sum_{y\in K_{x_{n},u_{n}}}P_{u_{n}}\left(y|x_{n}\right)}-\mathbf{1}_{n+1}^{\bar{K}}\phi(x_{n+1})^{T}\right)\theta_{n}\phi\left(x_{n}\right).\end{aligned}$$

The difference $\delta M_{n}-\delta M_{n}\left(\theta^{*}\right)$ can therefore be written as

$$\delta M_{n}-\delta M_{n}\left(\theta^{*}\right)=a\left(x_{n}\right)^{T}\left(\theta_{n}-\theta^{*}\right)b\left(x_{n}\right),$$

where a and b are vector valued functions of x_{n}. By the Cauchy–Schwarz inequality, for every x_{n}

$$\left|\delta M_{n}-\delta M_{n}\left(\theta^{*}\right)\right|\leq\left|a\left(x_{n}\right)\right|\left|\theta_{n}-\theta^{*}\right|\left|b\left(x_{n}\right)\right|,$$

and since the state space is finite, a and b are bounded, therefore there exists some constant k such that for every x_n

$$|a(x_n)||b(x_n)| \leq k,$$

and we have that

$$\mathrm{E}\left[|\delta M_n - \delta M_n(\theta^*)|^2 I_{\{|\theta_n-\theta^*|\leq\rho\}}\right] \leq k^2|\theta_n-\theta^*|^2 \to 0.$$

■

5. The sequence of processes $W(\cdot)$ defined on $(-\infty,\infty)$ by (54) converges weakly in $D^L(-\infty,\infty)$ to a Wiener process $W(\cdot)$, with covariance matrix Σ.

Proof

For the proof of this assumption we use Theorem 10.6.2 in Kushner and Yin (2003),which we now state.

Theorem 22 *(10.6.2 in Kushner and Yin, 2003) Assume 20.5.1-20.5.4. Then* $\{W(\cdot)\}$ *defined in* (54) *converges weakly to a Wiener process with covariance matrix* $\Sigma = \Sigma_0 + \Sigma_1 + \Sigma_1^T$.

■

5.1 The following equations hold:

$$\begin{aligned}\lim_{N\to\infty}\sup_n \mathrm{E}\left|\sum_{j=n+N}^{\infty}\mathrm{E}\left(F(\theta^*,x_j,u_j,x_{j+1})\middle|x_n,u_n\right)\right| &= 0,\\ \lim_{N\to\infty}\sup_n \mathrm{E}\left|\sum_{i=n+N}^{\infty}\mathrm{E}\left(F(\theta^*,x_n,u_n,x_{n+1})F(\theta^*,x_i,u_i,x_{i+1})|x_n,u_n\right)^T\right| &= 0.\end{aligned}$$

Proof

Since the transition probabilities at step j converge to the steady state transition probabilities (a property of ergodic Markov chains) exponentially fast in j, and since at the steady state $\mathrm{E}(F(\theta^*,x,u,x')) \equiv \bar{g}(\theta^*) = 0$, we have that for some $\rho < 1$ and some vector c

$$\left|\mathrm{E}\left(F(\theta^*,x_j,u_j,x_{j+1})\middle|x_n,u_n\right)\right| < c\rho^{j-n},$$

therefore for every n

$$\begin{aligned}\lim_{N\to\infty}\left|\sum_{j=n+N}^{\infty}\mathrm{E}\left(F(\theta^*,x_j,u_j,x_{j+1})\middle|x_n,u_n\right)\right| &\leq \lim_{N\to\infty} c\sum_{j=N}^{\infty}\rho^j\\ &= \lim_{N\to\infty}\frac{c\rho^N}{1-\rho}\\ &= 0.\end{aligned}$$

The same goes for the covariance, since there exists some $\rho' < 1$ and some matrix c' such that

$$\mathrm{E}\left(F\left(\theta^*, x_n, u_n, x_{n+1}\right) F\left(\theta^*, x_i, u_i, x_{i+1}\right) \middle| x_n, u_n\right)^T < c' \rho'^{i-n}.$$

■

5.2 The sets $\left\{\left|F\left(\theta^*, x_n, u_n, x_{n+1}\right)\right|^2\right\}$ and $\left\{\left|\sum_{j=i}^{\infty} \mathrm{E}\left(F\left(\theta^*, x_j, u_j, x_{j+1}\right) \middle| x_i, u_i\right)\right|^2\right\}$ are uniformly integrable.

Proof As was shown before, $F\left(\theta^*, x_n, u_n, x_{n+1}\right)$ is bounded, and therefore $\left\{\left|F\left(\theta^*, x_n, u_n, x_{n+1}\right)\right|^2\right\}$ is uniformly integrable. Also, as was shown in the proof of A5.1, for every i

$$\left|\sum_{j=i}^{\infty} \mathrm{E}\left(F\left(\theta^*, x_j, u_j, x_{j+1}\right) \middle| x_i, u_i\right)\right| \leq \frac{c}{1-\rho},$$

which is bounded, and therefore $\left\{\left|\sum_{j=i}^{\infty} \mathrm{E}\left(F\left(\theta^*, x_j, u_j, x_{j+1}\right) \middle| x_i, u_i\right)\right|^2\right\}$ is uniformly integrable.

■

5.3 There is a matrix Σ_0 such that

$$\frac{1}{m} \sum_{j=n}^{n+m-1} \mathrm{E}\left[F\left(\theta^*, x_j, u_j, x_{j+1}\right) F\left(\theta^*, x_j, u_j, x_{j+1}\right)^T \middle| x_n, u_n\right] - \Sigma_0 \to 0$$

in probability as $n, m \to \infty$.

Proof

Since the Markov chain is ergodic, by the law of large numbers this is satisfied by defining

$$\Sigma_0 = \lim_{n \to \infty} \mathrm{E}\left[F\left(\theta^*, x_n, u_n, x_{n+1}\right) F\left(\theta^*, x_n, u_n, x_{n+1}\right)^T\right].$$

■

5.4 There is a matrix Σ_1 such that

$$\frac{1}{m} \sum_{j=n}^{n+m-1} \sum_{k=j+1}^{\infty} \mathrm{E}\left[F\left(\theta^*, x_j, u_j, x_{j+1}\right) F\left(\theta^*, x_k, u_k, x_{k+1}\right)^T \middle| x_n, u_n\right] - \Sigma_1 \to 0$$

in probability as $n, m \to \infty$.

Proof

Since the Markov chain is ergodic, by the law of large numbers this is satisfied by defining

$$\Sigma_1 = \sum_{j=1}^{\infty} \lim_{n \to \infty} \mathrm{E}\left[F\left(\theta^*, x_n, u_n, x_{n+1}\right) F\left(\theta^*, x_{n+j}, u_{n+j}, x_{n+j+1}\right)^T\right].$$

■

6. $g(\cdot,x,u)$ is continuously differentiable for each x,u, and can be expanded as

$$g(\theta,x,u) = g(\theta^*,x,u) + [\nabla_\theta g(\theta^*,x,u)]^T(\theta-\theta^*) + [y(\theta,x,u)]^T(\theta-\theta^*),$$

where

$$[y(\theta,x,u)]^T(\theta-\theta^*) = \int_0^1 [\nabla_\theta g(\theta^*+s(\theta-\theta^*),x,u) - \nabla_\theta g(\theta^*,x,u)]\,ds, \tag{55}$$

and if $\delta \to 0$ as $\varepsilon \to 0$ and $n \to \infty$, then

$$\mathrm{E}\,|y(\theta_n,x_n,u_n)|\,I_{\{|\theta_n-\theta^*|\le\delta\}} \to 0$$

as $\varepsilon \to 0$ and $n \to \infty$.

Proof Recall that for IPM-TD(0)

$$g(\theta,x,u) = \left(r(x) + \left(\gamma\sum_y P_u(y|x)\phi(y)^T - \phi(x)^T\right)\theta\right)\phi(x),$$

which is linear in θ and thus can be expanded as

$$\begin{aligned} g(\theta,x,u) &= \left(r(x) + \left(\gamma\sum_y P_u(y|x)\phi(y)^T - \phi(x)^T\right)(\theta-\theta^*+\theta^*)\right)\phi(x) \\ &= g(\theta^*,x,u) + \left(\gamma\sum_y P_u(y|x)\phi(y)^T - \phi(x)^T\right)(\theta-\theta^*)\phi(x) \\ &= g(\theta^*,x,u) + [\nabla_\theta g(\theta^*,x,u)]^T(\theta-\theta^*). \end{aligned}$$

Since $\nabla_\theta g(\theta,x,u)$ does not depend on θ, the integral in (55) is zero and $y(\theta,x,u)=0$, thus the assumption is satisfied. ■

7. The set $\{\nabla_\theta g(\theta^*,x_n,u_n)\}$ is uniformly integrable.

Proof As was shown above, $\nabla_\theta g(\theta^*,x_n,u_n)$ is clearly bounded, and therefore uniformly integrable. ■

8. There is a Hurwitz matrix A such that

$$\frac{1}{m}\sum_{j=n}^{n+m+1}\left[\mathrm{E}\left[\nabla_\theta g^T(\theta^*,x_j,u_j)\big|x_n,u_n\right] - A\right] \to 0$$

in probability as $\varepsilon \to 0$ and $n,m \to \infty$.

Proof We have,

$$\nabla_\theta g^T(\theta^*, x, u) = \phi(x)\left(\gamma \sum_y P_u(y|x)\,\phi(y) - \phi(x)\right)^T.$$

Recall our definition of A

$$A \triangleq \Phi^T \Pi_\mu (\gamma P_\mu - I)\,\Phi.$$

Then, by the law of large numbers, we have

$$\frac{1}{m}\sum_{j=n}^{n+m+1}\left[\mathrm{E}\left[\nabla_\theta g^T(\theta^*, x_i, u_i)\middle| x_n, u_n\right] - A\right] \to 0.$$

As was stated before, it can be shown (Bertsekas and Tsitsiklis, 1996, Lemma 6.6b) that the eigenvalues of A all have a negative real part, therefore A is Hurwitz. ■

References

P. Abbeel, M. Quigley, and A.Y. Ng. Using inaccurate models in reinforcement learning. In *Proceedings of the 23rd International Conference on Machine Learning*, pages 1–8. ACM, 2006.

J. Abounadi, D. Bertsekas, and V.S. Borkar. Learning algorithms for markov decision processes with average cost. *SIAM Journal on Control and Optimization*, 40(3):681–698, 2001.

A.G. Barto, S.J. Bradtke, and S.P. Singh. Learning to act using real-time dynamic programming. *Artificial Intelligence*, 72(1-2):81–138, 1995.

D.P. Bertsekas. *Dynamic Programming and Optimal Control, Vol I & II*. Athena Scientific, third edition, 2006.

D.P. Bertsekas and J. Tsitsiklis. *Neuro-dynamic Programming*. Athena Scientific, 1996.

S. Bhatnagar, R. Sutton, M. Ghavamzadeh, and M. Lee. Natural actor–critic algorithms. Technical Report TR09-10, Univ. of Alberta, 2007.

V.S. Borkar. *Stochastic Approximation: a Dynamical Systems Viewpoint*. Cambridge University Press, 2008.

R.I. Brafman and M. Tennenholtz. R-max a general polynomial time algorithm for near-optimal reinforcement learning. *The Journal of Machine Learning Research*, 3:213–231, 2003.

R. Crites and A. Barto. Improving elevator performance using reinforcement learning. In *Advances in Neural Information Processing Systems 8*, pages 1017–1023. MIT Press, 1996.

N.D. Daw, Y. Niv, and P. Dayan. Uncertainty-based competition between prefrontal and dorsolateral striatal systems for behavioral control. *Nature Neuroscience*, 8(12):1704–1711, 2005.

P. Dayan and T.J. Sejnowski. TD(λ) converges with probability 1. *Machine Learning*, 14:295–301, 1994.

R.G. Gallager. *Discrete Stochastic Processes*. Kluwer Academic Publishers, 1995.

R. A. Horn and C. R. Johnson. *Matrix Analysis*. Cambridge University Press, 1985.

M. Kearns and S. Singh. Near-optimal reinforcement learning in polynomial time. *Machine Learning*, 49(2):209–232, 2002.

V. Konda. *Actor-Critic Algorithms*. PhD thesis, Massachusetts Institute of Technology, 2002.

P.R. Kumar. A survey of some results in stochastic adaptive control. *SIAM Journal on Control and Optimization*, 23:329–380, 1985.

H.J. Kushner and G. Yin. *Stochastic Approximation and Recursive Algorithms and Applications*. Springer Verlag, 2003.

L. Ljung. Analysis of recursive stochastic algorithms. *IEEE Transactions on Automatic Control*, 22 (4):551 – 575, 1977.

P. Marbach and J. Tsitsiklis. Simulation-based optimization of markov reward processes. *IEEE Transactions on Automatic Control*, 46(2):191–209, 1998.

P. Marbach, O. Mihatsch, M. Schulte, and J.N. Tsitsiklis. Reinforcement learning for call admission control and routing in integrated service networks. In *Advances in Neural Information Processing Systems 10*, pages 922–928. MIT Press, 1998.

A. Papoulis and S.U. Pillai. *Probability, Random Variables, and Stochastic Processes*. McGraw Hill, fourth edition, 2002.

M.J. Schervish. *Theory of Statistics*. Springer, 1995.

S. Singh and P. Dayan. Analytical mean squared error curves for temporal difference learning. *Machine Learning*, 32:5–40, 1998.

R.S. Sutton. Integrated architectures for learning, planning, and reacting based on approximating dynamic programming. In *Proceedings of the Seventh International Conference on Machine Learning*, pages 216–224. Morgan Kaufmann, 1990.

R.S. Sutton and A.G. Barto. *Reinforcement Learning*. MIT Press, 1998.

G. Tesauro. Temporal difference learning and td-gammon. *Commun. ACM*, 38, March 1995.

Journal of Machine Learning Research 13 (2012) 1967-1971 Submitted 2/12; Revised 5/12; Published 6/12

Jstacs: A Java Framework for Statistical Analysis and Classification of Biological Sequences

Jan Grau[*] JAN.GRAU@INFORMATIK.UNI-HALLE.DE
Institute of Computer Science
Martin Luther University Halle–Wittenberg
D-06099 Halle (Saale), Germany

Jens Keilwagen[*] JENS.KEILWAGEN@IPK-GATERSLEBEN.DE
Leibniz Institute of Plant Genetics and Crop Plant Research (IPK)
Corrensstraße 3
D-06466 Stadt Seeland, OT Gatersleben, Germany

André Gohr ANDRE.GOHR@INFORMATIK.UNI-HALLE.DE
Institute of Computer Science
Martin Luther University Halle–Wittenberg
D-06099 Halle (Saale), Germany

Berit Haldemann BERIT.HALDEMANN@INFORMATIK.HU-BERLIN.DE
Department of Computer Science
Humboldt University of Berlin
Unter den Linden 6, D-10099 Berlin, Germany

Stefan Posch STEFAN.POSCH@INFORMATIK.UNI-HALLE.DE
Ivo Grosse IVO.GROSSE@INFORMATIK.UNI-HALLE.DE
Institute of Computer Science
Martin Luther University Halle–Wittenberg
D-06099 Halle (Saale), Germany

Editor: Antti Honkela

Abstract

Jstacs is an object-oriented Java library for analysing and classifying sequence data, which emerged from the need for a standardized implementation of statistical models, learning principles, classifiers, and performance measures. In Jstacs, these components can be used, combined, and extended easily, which allows for a direct comparison of different approaches and fosters the development of new components. Jstacs is especially tailored to biological sequence data, but is also applicable to general discrete and continuous data. Jstacs is freely available at `http://www.jstacs.de` under the GNU GPL license including an API documentation, a cookbook, and code examples.

Keywords: machine learning, statistical models, Java, bioinformatics, classification

1. Introduction

During the last years, machine learning techniques have gained an increasing importance in many fields of science including bioinformatics and computational biology. A plethora of new or improved statistical models, learning principles, and classification approaches has evolved.

∗. Both authors contributed equally.

One critical step in assessing their relevance is the comparison to existing methods. Such direct comparisons are hampered, if the approaches compared are implemented in stand-alone applications or web-servers or if different performance measures are being used. In addition, the same building blocks, such as statistical models or evaluation of performance measures, are implemented repeatedly, which induces an unnecessary implementation overhead slowing down scientific progress.

These observations lead to the development of Jstacs as an object-oriented open-source library. In contrast to other libraries like JavaML (Abeel et al., 2009) or Shogun (Sonnenburg et al., 2010), Jstacs focuses on statistical models and statistical learning principles. Similar to JavaML, Jstacs is mainly targeted at developers who want to use the library in their own code.

In a typical Jstacs application for sequence classification, a user first chooses appropriate statistical models for the data of the different classes. One then combines these models to a classifier, chooses a learning principle for learning the parameters of this classifier, and learns this classifier on training data. Finally, one uses the classifier for predicting class labels for previously unseen data. For assessing the performance of the classifier, one can choose an evaluation schema like cross-validation and choose different performance measures.

At each of these steps, one may use a statistical model, classifier, learning principle, evaluation schema, or performance measure existing in Jstacs, or implement and use new ones. At each level, Jstacs defines interfaces and abstract classes to standardize and ease development, and to achieve modularity. Thus, a replacement of one component does not require a modification of other parts.

Jstacs has been applied to diverse biological problems such as prediction of transcription factor binding sites and splice sites, de-novo motif discovery, analysis of gene expression and Array-CGH data, and classification based on flow cytometry data.

In the following section, we describe the general structure, essential interfaces, and abstract classes of Jstacs. In a case study, we show how these can be used for building a problem-specific application.

2. The Jstacs Library

In Jstacs, data representation is organized at three levels: alphabets, sequences, and data sets. The most prevalent alphabet in Jstacs is the `DNAAlphabet`, while more general implementations can be used for instance to define a three-letter amino acid alphabet. `Sequence`s are defined using such alphabets, while `DataSet`s comprise a collection of sequences over the same alphabet. `DataSet`s are constructed either from an existing array of sequences or from a file. The latter is the standard way of loading data into Jstacs. In addition, `DataSet`s can be sampled from statistical models.

On the algorithmic side, Jstacs is organized around two central types depicted in Figure 1: the abstract class `AbstractClassifier` and the interface `StatisticalModel`, including two subinterfaces `TrainableStatisticalModel` and `DifferentiableStatisticalModel` abbreviated as `TrainSM` and `DiffSM`, respectively. For these interfaces, Jstacs provides abstract classes with standard implementations of many of the specified methods to reduce implementation effort as well as factory classes enabling user-friendly creation of many standard models.

`TrainableStatisticalModel`s provide methods for training the parameters of the model from one data set. For example, this can be accomplished generatively by analytic parameter estimation. Current implementations include inhomogeneous and homogeneous Markov models, Bayesian networks, hidden Markov models, and mixture models accepting any `TrainableStatisticalModel` as mixture components.

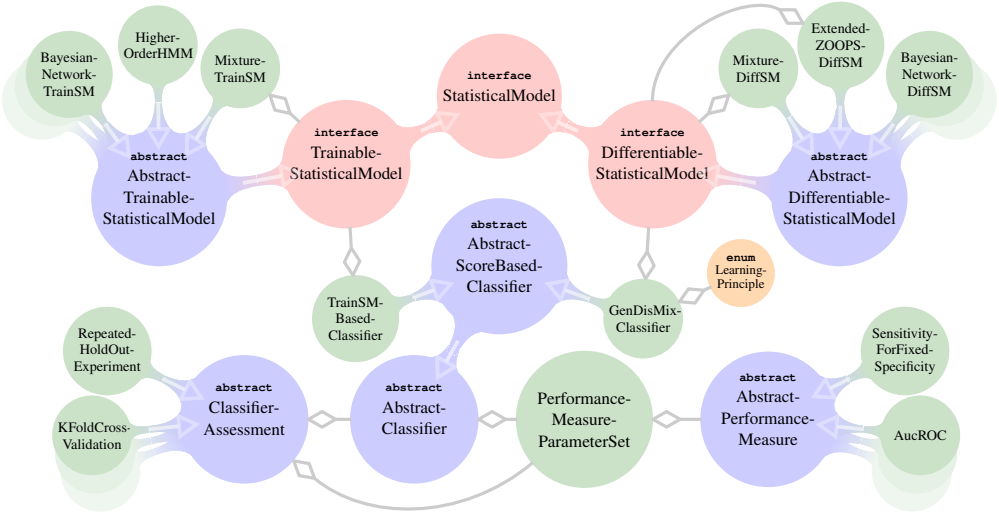


Figure 1: Part of the class structure of Jstacs. The `interfaces` are colored red, `abstract` classes blue, `enums` orange, and concrete classes green without preceeding modifier. Continuous transitions represent inheritance, where arrows indicate the direction of inheritance. Arrows with diamond heads represent usage of a type in the class at the arrow head.

In contrast, `DifferentiableStatisticalModels` provide methods tailored to numerical optimization like the computation of gradients with respect to their parameters. Jstacs provides several `DifferentiableStatisticalModels` including Markov models, Bayesian networks, and mixtures of `DifferentiableStatisticalModels`. In addition, a ZOOPS[1] model for de-novo motif discovery is implemented in `ExtendedZOOPSDiffSM`, which will be the topic of the case study.

`AbstractClassifiers` provide methods for learning internal `StatisticalModels` on training data from different classes and for classifying new input sequences. The `TrainSMBasedClassifier` trains each of the provided `TrainableStatisticalModels` separately on the data set for the corresponding class. The `GenDisMixClassifier` performs a simultaneous numerical parameter estimation for the enclosed `DifferentiableStatisticalModel`, for instance by maximum supervised posterior (MSP) (Grünwald et al., 2002; Cerquides and de Mántaras, 2005), or a unified learning principle (GenDisMix) (Keilwagen et al., 2010).

`ClassifierAssessments` can be used for assessing the performance of any `AbstractClassifier`, for example by k-fold cross validation or repeated holdout sampling. Here, the user may choose one or multiple performance measures such as sensitivity, precision, or the areas under the receiver operating characteristic and precision-recall curve.

3. Case Study

In this section, we describe how we used Jstacs for developing Dispom, a new application for de-novo motif discovery (Keilwagen et al., 2011). Existing approaches for de-novo motif discovery

1. ZOOPS abbreviates *Zero or one occurrence per sequence*.

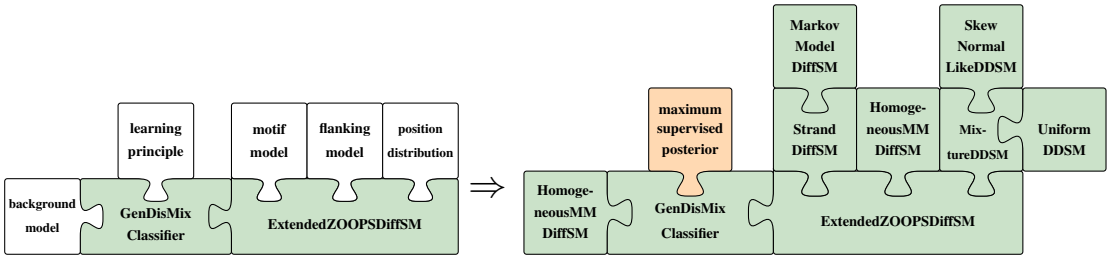


Figure 2: Structure of Dispom within Jstacs. The left side illustrates, how modules can be plugged into the core structure. The right side shows the concrete classes used in the application.

differ in the employed learning principle and in the capability of learning the positional preference of motif occurrences. However, prior to Dispom, no approach existed for learning the motif and the positional preference simultaneously using a discriminative learning principle. The general structure of Dispom in Jstacs is depicted on the left side of Figure 2, where each white piece represents a slot that can be filled with implementations of interfaces defined in Jstacs.

The motif, flanking, and background model are `DifferentiableStatisticalModel`s, the position distribution is a `DurationDiffSM`, and the learning principle is a value from an `enum` type. In the Dispom application illustrated on the right side of Figure 2, we use an inhomogeneous Markov model of order 0 with a mixture over the DNA-strands as motif model. We use homogeneous Markov models of order 0 for both the flanking and background model. All of these models existed before we started developing Dispom. We use a mixture of a skew normal and a uniform distribution as position distribution, and the discriminative MSP learning principle.

This modular structure allowed for an easy adaption to other problems like challenge 2 of DREAM5[2] on protein binding microarray data, where we simply increased the orders of the motif, flanking, and background model, and extended the learning principle to a weighted variant of the MSP principle. These minimal changes were all that was needed for developing a novel application for the analysis of protein binding microarrays and a successful performance in the challenge.

4. Author Contributions

JG, AG, SP and IG initiated the project and JG and AG were main developers. Later, JK joined as a main developer, and AG contributed occasionally. BH contributed to parts of Jstacs and to documentation. All authors contributed to writing and approved the final manuscript.

Acknowledgments

We thank Michael Scharfe and Michael Seifert for their contributions to Jstacs, and Ralf Eggeling, Martin Gleditzsch, Michaela Mohr, Birgit Möller, Martin Porsch, and Marc Strickert for valuable discussions. This work was supported by grant 0312706A by the German Ministry of Education and Research (BMBF) and grant XP3624HP/0606T by the Ministry of Culture of Saxony-Anhalt.

2. A description of the challenge is available at `http://wiki.c2b2.columbia.edu/dream/index.php/D5c2`.

References

Thomas Abeel, Yves Van de Peer, and Yvan Saeys. Java-ML: A machine learning library. *Journal of Machine Learning Research*, 10:931–934, June 2009.

Jesús Cerquides and Ramon López de Mántaras. Robust Bayesian linear classifier ensembles. In *Proceedings of the 16th European Conference on Machine Learning*, volume 3720 of *Lecture Notes in Computer Science*, pages 72–83. Springer, 2005.

Peter Grünwald, Petri Kontkanen, Petri Myllymäki, Teemu Roos, Henry Tirri, and Hannes Wettig. Supervised posterior distributions. Presented at the Seventh Valencia International Meeting on Bayesian Statistics, 2002. URL `http://homepages.cwi.nl/~pdg/presentations/Valencia2.pdf`.

Jens Keilwagen, Jan Grau, Stefan Posch, Marc Strickert, and Ivo Grosse. Unifying generative and discriminative learning principles. *BMC Bioinformatics*, 11(1):98, 2010.

Jens Keilwagen, Jan Grau, Ivan A. Paponov, Stefan Posch, Marc Strickert, and Ivo Grosse. De-novo discovery of differentially abundant transcription factor binding sites including their positional preference. *PLoS Computational Biology*, 7(2):e1001070, 02 2011.

Sören Sonnenburg, Gunnar Rätsch, Sebastian Henschel, Christian Widmer, Jonas Behr, Alexander Zien, Fabio de Bona, Alexander Binder, Christian Gehl, and Vojtech Franc. The SHOGUN machine learning toolbox. *Journal of Machine Learning Research*, 11:1799–1802, August 2010.

Journal of Machine Learning Research 13 (2012) 1973-1998 Submitted 3/11; Revised 1/12; Published 6/12

Variable Selection in High-dimensional Varying-coefficient Models with Global Optimality

Lan Xue XUEL@STAT.OREGONSTATE.EDU
Department of Statistics
Oregon State University
Corvallis, OR 97331-4606, USA

Annie Qu ANNIEQU@ILLINOIS.EDU
Department of Statistics
University of Illinois at Urbana-Champaign
Champaign, IL 61820-3633, USA

Editor: Xiaotong Shen

Abstract

The varying-coefficient model is flexible and powerful for modeling the dynamic changes of regression coefficients. It is important to identify significant covariates associated with response variables, especially for high-dimensional settings where the number of covariates can be larger than the sample size. We consider model selection in the high-dimensional setting and adopt difference convex programming to approximate the L_0 penalty, and we investigate the global optimality properties of the varying-coefficient estimator. The challenge of the variable selection problem here is that the dimension of the nonparametric form for the varying-coefficient modeling could be infinite, in addition to dealing with the high-dimensional linear covariates. We show that the proposed varying-coefficient estimator is consistent, enjoys the oracle property and achieves an optimal convergence rate for the non-zero nonparametric components for high-dimensional data. Our simulations and numerical examples indicate that the difference convex algorithm is efficient using the coordinate decent algorithm, and is able to select the true model at a higher frequency than the least absolute shrinkage and selection operator (LASSO), the adaptive LASSO and the smoothly clipped absolute deviation (SCAD) approaches.

Keywords: coordinate decent algorithm, difference convex programming, L_0- regularization, large-p small-n, model selection, nonparametric function, oracle property, truncated L_1 penalty

1. Introduction

High-dimensional data occur very frequently and are especially common in biomedical studies including genome studies, cancer research and clinical trials, where one of the important scientific interests is in dynamic changes of gene expression, long-term effects for treatment, or the progression of certain diseases.

We are particularly interested in the varying-coefficient model (Hastie and Tibshirani, 1993; Ramsay and Silverman, 1997; Hoover et al., 1998; Fan and Zhang, 2000; Wu and Chiang, 2000; Huang, Wu and Zhou, 2002, 2004; Qu and Li, 2006; Fan and Huang, 2005; among others) as it is powerful for modeling the dynamic changes of regression coefficients. Here the response variables are associated with the covariates through linear regression, but the regression coefficients can vary and are modeled as a nonparametric function of other predictors.

In the case where some of the predictor variables are redundant, the varying-coefficient model might not be able to produce an accurate and efficient estimator. Model selection for significant predictors is especially critical when the dimension of covariates is high and possibly exceeds the sample size, but the number of nonzero varying-coefficient components is relatively small. This is because even a single predictor in the varying-coefficient model could be associated with a large number of unknown parameters involved in the nonparametric functions. Inclusion of high-dimensional redundant variables can hinder efficient estimation and inference for the non-zero coefficients.

Recent developments in variable selection for varying-coefficient models include Wang, Li and Huang (2008) and Wang and Xia (2009), where the dimension of candidate models is finite and smaller than the sample size. Wang, Li and Huang (2008) considered the varying-coefficient model in a longitudinal data setting built on the SCAD approach (Fan and Li, 2001; Fan and Peng, 2004), and Wang and Xia (2009) proposed the use of local polynomial regression with an adaptive LASSO penalty. For the high-dimensional case when the dimension of covariates is much larger than the sample size, Wei, Huang and Li (2011) proposed an adaptive group LASSO approach using B-spline basis approximation. The SCAD penalty approach has the advantages of unbiasedness, sparsity and continuity. However, the SCAD approach involves non-convex optimization through local linear or quadratic approximations (Hunter and Li, 2005; Zou and Li, 2008), which is quite sensitive to the initial estimator. In general, the global minimum is not easily obtained for non-convex function optimization. Kim, Choi and Oh (2008) have improved SCAD model selection using the difference convex (DC) algorithm (An and Tao, 1997; Shen et al., 2003). Still, the existence of global optimality for the SCAD has not been investigated for the case that the dimension of covariates exceeds the sample size. Alternatively, the adaptive LASSO and the adaptive group LASSO approaches are easier to implement due to solving the convex optimization problem. However, the adaptive LASSO algorithm requires the initial estimators to be consistent, and such a requirement could be difficult to obtain in high-dimensional settings.

Indeed, obtaining consistent initial estimators of the regression parameters is more difficult than the model selection problem when the dimension of covariates exceeds the sample size, since if the initial estimator is already close to the true value, then performing model selection is much less challenging. So far, most model selection algorithms rely on consistent LASSO estimators as initial values. However, the irrepresentable assumption (Zhao and Yu, 2006) to obtain consistent LASSO estimators for high-dimensional data is unlikely to be satisfied, since most of the covariates are correlated. When the initial consistent estimators are no longer available, the adaptive LASSO and the SCAD algorithm based on either local linear or quadratic approximations are likely to fail.

To overcome the aforementioned problems, we approximate the L_0 penalty effectively as the L_0 penalty is considered to be optimal for achieving sparsity and unbiasedness, and is optimal even for the high-dimensional data case. However, the challenge of L_0 regularization is computational difficulty due to its non-convexity and non-continuity. We use a newly developed truncated L_1 penalty (TLP, Shen, Pan and Zhu, 2012) for the varying-coefficient model which is piecewise linear and continuous to approximate the non-convex penalty function. The new method intends to overcome the computational difficulty of the L_0 penalty while preserving the optimality of the L_0 penalty. The key idea is to decompose the non-convex penalty function by taking the difference between two convex functions, thereby transforming a non-convex problem into a convex optimization problem.

One of the main advantages of the proposed approach is that the minimization process does not depend on the initial estimator, which could be hard to obtain when the dimension of covariates is high. In addition, the proposed algorithm for the varying-coefficient model is computationally effi-

cient. This is reflected in that the proposed model selection performs better than existing approaches such as SCAD in the high-dimensional case, based on our simulation and as applied to HIV AIDs data, with a much higher frequency of choosing the correct model. The improvement is especially significant when the dimension of covariates is much higher than the sample size.

We derive model selection consistency for the proposed method and show that it possesses the oracle property when the dimension of covariates exceeds the sample size. Note that the theoretical derivation of asymptotic properties and global optimality results are rather challenging for varying-coefficient model selection, as we are dealing with an infinite dimension of the nonparametric component in addition to the high-dimensional covariates. In addition, the optimal rate of convergence for the non-zero nonparametric components can be achieved in high-dimensional varying-coefficient models. The theoretical techniques applied in this project are innovative as there is no existing theoretical result on global optimality for high-dimensional model selection in the varying-coefficient model framework.

The paper is organized as follows. Section 2 provides the background of varying-coefficient models. Section 3 introduces the penalized polynomial spline procedure for selecting varying-coefficient models when the dimension of covariates is high, provides the theoretical properties for model selection consistency and establishes the relationship between the oracle estimator and the global and local minimizers. Section 4 provides tuning parameter selection, and the coordinate decent algorithm for model selection implementation. Section 5 demonstrates simulations and a data example for high-dimensional data. The last section provides concluding remarks and discussion.

2. Varying-coefficient Model

Let $(\mathbf{X}_i, U_i, Y_i), i = 1, \ldots, n$, be random vectors that are independently and identically distributed as $(\mathbf{X}, U, Y)$, where $\mathbf{X} = (X_1, \ldots, X_d)^T$ and a scalar U are predictor variables, and Y is a response variable. The varying-coefficient model (Hastie and Tibshirani, 1993) has the following form:

$$Y_i = \sum_{j=1}^{d} \beta_j(U_i) X_{ij} + \varepsilon_i, \tag{1}$$

where X_{ij} is the jth component of $\mathbf{X}_i$, $\beta_j(\cdot)$'s are unknown varying-coefficient functions, and ε_i is a random noise with mean 0 and finite variance σ^2. The varying-coefficient model is flexible in that the responses are linearly associated with a set of covariates, but their regression coefficients can vary with another variable U. We will call U the index variable and $\mathbf{X}$ the linear covariates. In practice, some of the linear covariates may be irrelevant to the response variable, with the corresponding varying-coefficient functions being zero almost surely. The goal of this paper is to identify the irrelevant linear covariates and estimate the nonzero coefficient functions for the relevant ones.

In many applications, such as microarray studies, the total number of the available covariates d can be much larger than the sample size n, although we assume that the number of relevant ones is fixed. In this paper, we propose a penalized polynomial spline procedure in variable selection for the varying-coefficient model where the number of linear covariates d is much larger than n. The proposed method is easy to implement and fast to compute. In the following, without loss of generality, we assume there exists an integer d_0 such that $0 < E\left[\beta_j^2(U)\right] < \infty$ for $j = 1, \ldots, d_0$, and $E\left[\beta_j^2(U)\right] = 0$ for $j = d_0, \ldots, d$. Furthermore, we assume that only the first d_0 covariates in $\mathbf{X}$ are relevant, and that the rest of the covariates are redundant.

3. Model Selection in High-dimensional Data

In our estimation procedure, we first approximate the smooth functions $\{\beta_j(\cdot)\}_{j=1}^d$ in (1) by polynomial splines. Suppose U takes values in $[a,b]$ with $a<b$. Let υ_j be a partition of the interval $[a,b]$, with N_n interior knots

$$\upsilon_j = \{a = \upsilon_{j,0} < \upsilon_{j,1} < \cdots < \upsilon_{j,N_n} < \upsilon_{j,N_n+1} = b\}.$$

Using υ_j as knots, the polynomial splines of order $p+1$ are functions which are p-degree (or less) of polynomials on intervals $[\upsilon_{j,i}, \upsilon_{j,i+1}), i = 0,\ldots,N_n-1$, and $[\upsilon_{j,N_n}, \upsilon_{j,N_n+1}]$, and have $p-1$ continuous derivatives globally. We denote the space of such spline functions by φ_j. The advantage of polynomial splines is that they often provide good approximations of smooth functions with only a small number of knots.

Let $\{B_{jl}(\cdot)\}_{l=1}^{J_n}$ be a set of B-spline bases of φ_j with $J_n = N_n + p + 1$. Then for $j = 1,\ldots,d$,

$$\beta_j(\cdot) \approx s_j(\cdot) = \sum_{l=1}^{J_n} \gamma_{jl} B_{jl}(\cdot) = \gamma_j^T \mathbf{B}_j(\cdot),$$

where $\gamma_j = (\gamma_{j1},\ldots,\gamma_{jJ_n})^T$ is a set of coefficients, and $\mathbf{B}_j(\cdot) = (B_{j1}(\cdot),\ldots,B_{jJ_n}(\cdot))^T$ are B- spline bases. The standard polynomial spline method (Huang, Wu and Zhou, 2002) estimates the coefficient functions $\{\beta_j(\cdot)\}_{j=1}^d$ by spline functions which minimize the sum of squares

$$\left(\widetilde{\beta}_1,\ldots,\widetilde{\beta}_d\right) = \underset{s_j\in\varphi_j, j=1,\ldots,d}{\operatorname{argmin}} \frac{1}{2n}\sum_{i=1}^n \left[Y_i - \sum_{j=1}^d s_j(U_i) X_{ij}\right]^2.$$

Equivalently, in terms of B-spline basis, it estimates $\gamma = \left(\gamma_1^T,\ldots,\gamma_d^T\right)^T$ by

$$\widetilde{\gamma} = \left(\widetilde{\gamma}_1^T,\ldots,\widetilde{\gamma}_d^T\right)^T = \underset{\gamma_j, j=1,\ldots,d}{\operatorname{argmin}} \frac{1}{2n}\sum_{i=1}^n \left[Y_i - \sum_{j=1}^d \gamma_j^T \mathbf{Z}_{ij}\right]^2, \tag{2}$$

where $\mathbf{Z}_{ij} = \mathbf{B}_j(U_i) X_{ij} = (B_{j1}(U_i) X_{ij},\ldots,B_{jJ_n}(U_i) X_{ij})^T$. However, the standard polynomial spline approach fails to reduce model complexity when some of the linear covariates are redundant, and furthermore is not able to obtain parameter estimation when the dimension of model d is larger than the sample size n. Therefore, to perform simultaneous variable selection and model estimation, we propose minimizing the penalized sum of squares

$$L_n(\mathbf{s}) = \frac{1}{2n}\sum_{i=1}^n \left[Y_i - \sum_{j=1}^d s_j(U_i) X_{ij}\right]^2 + \lambda_n \sum_{j=1}^d p_n\left(\|s_j\|_n\right), \tag{3}$$

where $\mathbf{s} = \mathbf{s}(\cdot) = (s_1(\cdot),\ldots,s_d(\cdot))^T$, and $\|s_j\|_n = \left(\sum_{i=1}^n s_j^2(U_i) X_{ij}^2/n\right)^{1/2}$ is the empirical norm. In terms of the B-spline basis, (3) is equivalent to

$$L_n(\gamma) = \frac{1}{2n}\sum_{i=1}^n \left[Y_i - \sum_{j=1}^d \gamma_j^T \mathbf{Z}_{ij}\right]^2 + \lambda_n \sum_{j=1}^d p_n\left(\|\gamma_j\|_{W_j}\right), \tag{4}$$

where $\|\gamma_j\|_{W_j} = \sqrt{\gamma_j^T \mathbf{W}_j \gamma_j}$ with $\mathbf{W}_j = \sum_{i=1}^{n} \mathbf{Z}_{ij}\mathbf{Z}_{ij}^T/n$. The formulation (3) is quite general. In particular, for a linear model with $\beta_j(u) = \beta_j$ and the linear covariates being standardized with $\sum_{i=1}^n X_{ij}/n = 0$ and $\sum_{i=1}^n X_{ij}^2/n = 1$ for $j = 1,\ldots,d$, (3) reduces to a family of variable selection methods for linear models with the penalty $p_n\left(\|s_j\|_n\right) = p_n\left(|\beta_j|\right)$. For instance, the L_1 penalty $p_n(|\beta|) = |\beta|$ results in LASSO (Tibshirani, 1996), and the smoothly clipped absolute deviation penalty results in SCAD (Fan and Li, 2001). In this paper, we consider a rather different approach for the penalty function such that

$$p_n(\beta) = p(\beta, \tau_n) = \min(|\beta|/\tau_n, 1), \tag{5}$$

which is called a truncated L_1-penalty (TLP) function, as proposed in Shen, Pan and Zhu (2012). In (5), the additional tuning parameter τ_n is a threshold parameter determining which individual components are to be shrunk towards to zero, or not. As pointed out by Shen, Pan and Zhu (2012), the TLP corrects the bias of the LASSO induced by the convex L_1-penalty and also reduces the computational instability of the L_0-penalty. The TLP is able to overcome the computation difficulty for solving non-convex optimization problems by applying difference convex programming, which transforms non-convex problems into convex optimization problems. This leads to significant computational advantages over its smooth counterparts, such as the SCAD (Fan and Li, 2001) and the minimum concavity penalty (MCP, Zhang, 2010). In addition, the TLP works particularly well for high-dimensional linear regression models as it does not depend on initial consistent estimators of coefficients, which could be difficult to obtain when d is much larger than n. In this paper, we will investigate the local and global optimality of the TLP for variable selection in varying-coefficient models in the high-dimensional case when $d \gg n$, and n goes to infinity.

Here we obtain $\widehat{\gamma}$ by minimizing $L_n(\gamma)$ in (4). As a result, for any $u \in [a,b]$, the estimators of the unknown varying-coefficient functions in (1) are given as

$$\widehat{\beta}_j(u) = \sum_{l=1}^{J_n} \widehat{\gamma}_{jl} B_{jl}(u), \ j = 1,\ldots,d. \tag{6}$$

Let $\widetilde{\gamma}^{(o)} = (\widetilde{\gamma}_1,\ldots,\widetilde{\gamma}_{d_0},0,\ldots,0)^T$ be the oracle estimator with the first d_0 elements being the standard polynomial estimator (2) of the true model consisting of only the first d_0 covariates. The following theorems establish the asymptotic properties of the proposed estimator. We only state the main results here and relegate the regularity conditions and proofs to the Appendix.

Theorem 1 *Let $A_n(\lambda_n, \tau_n)$ be the set of local minima of (4). Under conditions (C1-C7) in the Appendix, the oracle estimator is a local minimizer with probability tending to 1, that is,*

$$P\left(\widetilde{\gamma}^{(o)} \in A_n(\lambda_n, \tau_n)\right) \to 1,$$

as $n \to \infty$.

Theorem 2 *Let $\widehat{\gamma} = (\widehat{\gamma}_1,\ldots,\widehat{\gamma}_d)^T$ be the global minima of (4). Under conditions (C1-C6), (C8) and (C9) in the Appendix, the estimator by minimizing (4) enjoys the oracle property, that is,*

$$P\left(\widehat{\gamma} = \widetilde{\gamma}^{(o)}\right) \to 1,$$

as $n \to \infty$.

Theorem 1 guarantees that the oracle estimator must fall into the local minima set. Theorem 2, in addition, provides sufficient conditions such that the global minimizer by solving the non-convex objective function in (4) is also the oracle estimator.

In addition to the results of model selection consistency, we also establish the oracle property for the non-zero components of the varying-coefficients. For any $u\in[a,b]$, let $\widehat{\beta}^{(1)}(u) = \left(\widehat{\beta}_1(u),\ldots,\widehat{\beta}_{d_0}(u)\right)^T$ be the estimator of the first d_0 varying-coefficient functions which are non-zero and are defined in (6) with $\widehat{\gamma}$ being the global minima of (4). Theorem 3 establishes the asymptotic normality of $\widehat{\beta}^{(1)}(u)$ with the optimal rate of convergence.

Theorem 3 *Under conditions (C1) - (C6), (C8) and (C9) given in the Appendix, and if* $\lim N_n \log N_n/n = 0$*, then for any* $u \in [a,b]$,

$$\left\{V\left(\widehat{\beta}^{(1)}(u)\right)\right\}^{-1/2}\left(\widehat{\beta}^{(1)}(u)-\beta_0^{(1)}(u)\right)\to N(0,\mathbf{I})$$

in distribution, where $\beta_0^{(1)}(u) = (\beta_{01}(u),\ldots,\beta_{0d_0}(u))^T$, $\mathbf{I}$ *is a* $d_0\times d_0$ *identity matrix, and*

$$V\left(\widehat{\beta}^{(1)}(u)\right) = \mathbf{B}^{(1)}(u)\left(\sum_{i=1}^n \mathbf{A}_i^{(1)T}\mathbf{A}_i^{(1)}\right)^{-1}\mathbf{B}^{(1)}(u) = O_p(N_n/n),$$

in which $\mathbf{B}^{(1)}(u) = \left(\mathbf{B}_1^T(u),\ldots,\mathbf{B}_{d_0}^T(u)\right)^T$, *and* $\mathbf{A}_i^{(1)} = \left(\mathbf{B}_1^T(U_i)X_{i1},\ldots,\mathbf{B}_{d_0}^T(U_i)X_{id_0}\right)^T$ *with* $\mathbf{B}_j^T(U_i)X_{ij} = (B_{j1}(U_i)X_{ij},\ldots,B_{jJ_n}(U_i)X_{ij})$.

4. Implementation

In this section, we extend the difference convex (DC) algorithm of Shen, Pan and Zhu (2012) to solve the nonconvex minimization in (4) for varying-coefficient models. In addition, we provide the tuning parameter selection criteria.

4.1 An Algorithm

The idea of the DC algorithm is to decompose a non-convex object function into a difference between two convex functions. Then the final solution is obtained iteratively by minimizing a sequence of upper convex approximations of the non-convex objective function. Specifically, we decompose the penalty in (5) as $p_n(\beta) = p_{n1}(\beta) - p_{n2}(\beta)$, where $p_{n1}(\beta) = |\beta|/\tau_n$ and $p_{n2}(\beta) = \max(|\beta|/\tau_n - 1, 0)$. Note that both $p_{n1}(\cdot)$ and $p_{n2}(\cdot)$ are convex functions. Therefore, we can decompose the non-convex objective function $L_n(\gamma)$ in (4) as a difference between two convex functions,

$$L_n(\gamma) = L_{n1}(\gamma) - L_{n2}(\gamma),$$

where

$$\begin{aligned} L_{n1}(\gamma) &= \frac{1}{2n}\sum_{i=1}^n\left[Y_i - \sum_{j=1}^d \gamma_j^T\mathbf{Z}_{ij}\right]^2 + \lambda_n\sum_{j=1}^d p_{n1}\left(\|\gamma_j\|_{W_j}\right), \\ L_{n2}(\gamma) &= \lambda_n\sum_{j=1}^d p_{n2}\left(\|\gamma_j\|_{W_j}\right). \end{aligned}$$

Let $\widehat{\gamma}^{(0)}$ be an initial value. From our experience, the proposed algorithm does not rely on initial consistent estimators of coefficients so we have used $\widehat{\gamma}^{(0)} = \mathbf{0}$ in the implementations. At iteration m, we set $L_n^{(m)}(\gamma)$, an upper approximation of $L_n(\gamma)$, equal to

$$\begin{aligned} & L_{n1}(\gamma) - \left[L_{n2}\left(\widehat{\gamma}^{(m-1)}\right) + \lambda_n \sum_{j=1}^{d} \left(\|\gamma_j\|_{W_j} - \left\|\widehat{\gamma}_j^{(m-1)}\right\|_{W_j} \right) p_{n2}^{'}\left(\left\|\widehat{\gamma}_j^{(m-1)}\right\|_{W_j} \right) \right] \\ \approx & \frac{1}{2n}\sum_{i=1}^{n}\left[Y_i - \sum_{j=1}^{d} \gamma_j^T \mathbf{Z}_{ij} \right]^2 + \frac{\lambda_n}{\tau_n}\sum_{j=1}^{d} \|\gamma_j\|_{W_j} I\left(\left\|\widehat{\gamma}_j^{(m-1)}\right\|_{W_j} \le \tau_n \right) \\ & -L_{n2}\left(\widehat{\gamma}^{(m-1)}\right) + \frac{\lambda_n}{\tau_n}\sum_{j=1}^{d} \left\|\widehat{\gamma}_j^{(m-1)}\right\|_{W_j} I\left(\left\|\widehat{\gamma}_j^{(m-1)}\right\|_{W_j} > \tau_n \right), \end{aligned}$$

where $p_{n2}^{'}\left(\left\|\widehat{\gamma}_j^{(m-1)}\right\|_{W_j} \right) = \frac{1}{\tau_n} I(\left\|\widehat{\gamma}_j^{(m-1)}\right\|_{W_j} > \tau_n)$ is the subgradient of p_{n2}**.** Since the last two terms of the above equation do not depend on γ, therefore at iteration m,

$$\widehat{\gamma}^{(m)} = \underset{\gamma_j, j=1,\ldots,d}{\operatorname{argmin}} \left\{ \frac{1}{2n}\sum_{i=1}^{n}\left[Y_i - \sum_{j=1}^{d} \gamma_j^T \mathbf{Z}_{ij} \right]^2 + \sum_{j=1}^{d} \lambda_{nj} \|\gamma_j\|_{W_j} \right\}, \tag{7}$$

where $\lambda_{nj} = \frac{\lambda_n}{\tau_n} I\left(\left\|\widehat{\gamma}_j^{(m-1)}\right\|_{W_j} \le \tau_n \right)$. Then it reduces to a group lasso with component-specific tuning parameter λ_{nj}. It can be solved by applying the coordinate-wise descent (CWD) algorithm as in Yuan and Lin (2006). To be more specific, let $\mathbf{Z}_{ij}^* = \mathbf{W}_j^{-1/2}\mathbf{Z}_{ij}$ and $\gamma_j^* = \mathbf{W}_j^{1/2}\gamma_j$. Then the minimization problem in (7) reduces to

$$\widehat{\gamma}^{*(m)} = \underset{\gamma_j^*, j=1,\ldots,d}{\operatorname{argmin}} \left\{ \frac{1}{2n}\sum_{i=1}^{n}\left[Y_i - \sum_{j=1}^{d} \gamma_j^{*T} \mathbf{Z}_{ij}^* \right]^2 + \lambda_{nj}\sum_{j=1}^{d} \|\gamma_j^*\|_2 \right\}. \tag{8}$$

Then the CWD algorithm minimizes (8) in each component while fixing the remaining components at their current value. For the jth component, $\widehat{\gamma}_j^{*(m)}$ is updated by

$$\gamma_j^{*(m)} = \left(1 - \frac{\lambda_{nj}}{\|S_j\|_2} \right)_+ S_j, \tag{9}$$

where $S_j = \mathbf{Z}_j^{*T}\left(\mathbf{Y} - \mathbf{Z}^* \gamma_{-j}^{*(m)} \right)$ with $\gamma_{-j}^{*(m)} = \left(\gamma_1^{*(m)T}, \ldots, \gamma_{j-1}^{*(m)T}, \mathbf{0}^T, \gamma_{j+1}^{*(m)T}, \ldots, \gamma_d^{*(m)T} \right)^T$, $\mathbf{Z}_j^* = \left(\mathbf{Z}_{1j}^*, \ldots, \mathbf{Z}_{nj}^* \right)^T$, $\mathbf{Z}^* = \left(\mathbf{Z}_1^*, \ldots, \mathbf{Z}_d^* \right)$ and $(x)_+ = xI_{\{x \ge 0\}}$. The solution to (8) can therefore be obtained by iteratively applying Equation (9) to $j = 1, \ldots, d$ until convergence.

The above algorithm is piece-wise linear and therefore it is computationally efficient. The penalty part in (7) only involves a large L_2-norm of the varying-coefficient function, implying that there is no shrinkage for the non-zero components with a large magnitude of coefficients. In addition, the above algorithm can capture weak signals of varying-coefficients, and meanwhile is able to

obtain the sparsest solution through tuning the additional thresholding parameters τ_n. The involvement of the additional tuning of τ_n makes the TLP a flexible optimization procedure.

The minimization in (4) can achieve the global minima if the leading convex function can be approximated, and it is called the outer approximation method (Breiman and Cutler, 1993). However, it has a slower convergence rate. Here we approximate the trailing convex function with fast computation, and it leads to a good local minimum if it is not global (Shen, Pan and Zhu, 2012). It can achieve the global minimizer if it is combined with the branch-and-bound method (Liu, Shen and Wong, 2005), which searches through all the local minima with an additional cost in computation. This contrasts to the SCAD or adaptive LASSO approaches which are based on local approximation. Achieving the global minimum is particularly important if the dimension of covariates is high, as the number of possible local minima increases dramatically as p increases. Therefore, any local approximation algorithm which relies on initial values likely fails.

4.2 Tuning Parameter Selection

The performance of the proposed spline TLP method crucially depends on the choice of tuning parameters. One needs to choose the knot sequences in the polynomial spline approximation and λ_n, τ_n in the penalty function. For computation convenience, we use equally spaced knots with the number of interior knots $N_n = [n^{1/(2p+3)}]$, and select only λ_n, τ_n. A similar strategy for knot selection can also be found in Huang, Wu and Zhou (2004), and Xue, Qu and Zhou (2010). Let $\theta_n = (\lambda_n, \tau_n)$ be the parameters to be selected. For faster computation, we use K-fold cross-validation to select θ_n, with $K = 5$ in the implementation. The full data T is randomly partitioned into K groups of about the same size, denoted as T_v , for $v = 1, \ldots, K$. Then for each v, the data $T - T_v$ is used for estimation and T_v is used for validation. For any given θ_n, let $\hat{\beta}_j^{(v)}(\cdot, \theta_n)$ be the estimators of $\beta_j(\cdot)$ using the training data $T - T_v$ for $j = 1, \ldots, d$. Then the cross-validation criterion is given as

$$\mathrm{CV}(\theta_n) = \sum_{v=1}^{K} \sum_{i \in T_v} \left\{ Y_i - \sum_{j=1}^{d} \hat{\beta}_j^{(v)}(U_i, \theta_n) X_{ij} \right\}^2 .$$

We select $\hat{\theta}_n$ by minimizing $\mathrm{CV}(\theta_n)$.

5. Simulation and Application

In this section, we conduct simulation studies to demonstrate the finite sample performance of the proposed method. We also illustrate the proposed method with an analysis of an AIDS data set. The total average integrated squared error (TAISE) is evaluated to assess estimation accuracy. Let $\widehat{\beta}^{(r)}$ be the estimator of a nonparametric function β in the r-th ($1 \le r \le R$) replication and $\{u_m\}_{m=1}^{n_{\mathrm{grid}}}$ be the grid points where $\widehat{\beta}^{(r)}$ is evaluated. We define $\mathrm{AISE}\left(\widehat{\beta}\right) = \frac{1}{R}\sum_{r=1}^{R} \frac{1}{n_{\mathrm{grid}}} \sum_{m=1}^{n_{\mathrm{grid}}} \left\{ \beta(u_m) - \widehat{\beta}^{(r)}(u_m) \right\}^2$, and $\mathrm{TAISE} = \sum_{l=1}^{d} \mathrm{AISE}\left(\widehat{\beta}_l\right)$. Let $\mathcal{S}$ and $\mathcal{S}_0$ be the selected and true index sets containing significant variables, respectively. We say $\mathcal{S}$ is correct if $\mathcal{S} = \mathcal{S}_0$; $\mathcal{S}$ overfits if $\mathcal{S}_0 \subset \mathcal{S}$ but $\mathcal{S}_0 \neq \mathcal{S}$; and $\mathcal{S}$ underfits if $\mathcal{S}_0 \not\subset \mathcal{S}$. In all simulation studies, the total number of simulations is 500.

5.1 Simulated Example

We consider the following varying-coefficient model

$$Y_i = \sum_{j=1}^{d} \beta_j(U_i) X_{ij} + \varepsilon_i,\ i = 1, \ldots, 200, \tag{10}$$

where the index variables U_i are generated from a Uniform $[0,1]$, and the linear covariates $\mathbf{X}_i$ are generated from a multivariate normal distribution with mean $\mathbf{0}$ and $Cov(X_{ij}, X_{ij'}) = 0.5^{|j-j'|}$, the noises ε_i are generated from a standard normal distribution, and the coefficient functions are of the forms

$$\beta_1(u) = \sin(2\pi u),\ \ \beta_2(u) = (2u-1)^2 + 0.5,\ \ \beta_3(u) = \exp(2u-1) - 1,$$

and $\beta_j(u) = 0$ for $j = 4, \ldots, d$. Therefore only the first three covariates are relevant for predicting the response variable, and the rest are null variables and do not contribute to the model prediction. We consider the model (10) with $d = 10$, 100, 200, or 400 to examine the performance of model selection and estimation when d is smaller than, close to, or exceeds the sample size.

We apply the proposed varying-coefficient TLP with a linear spline. The simulation results based on the cubic spline are not provided here as they are quite similar to those based on the linear spine. The tuning parameters are selected using the five-fold cross-validation procedure as described in Section 4.2. We compare the TLP approach to a penalized spline procedure with the SCAD penalty, the group LASSO (LASSO) and the group adaptive LASSO (AdLASSO) as described in Wei, Huang and Li (2011). For the SCAD penalty, the first order derivative of $p_n(\cdot)$ in (4) is given as $p_n'(\theta) = I(\theta \le \lambda_n) + \frac{(a\lambda_n - \theta)_+}{(a-1)\lambda_n} I(\theta > \lambda_n)$, and we set $a = 3.7$ as in Fan and Li (2001). For all procedures, we select the tuning parameters using a five-fold cross-validation procedure for fair comparison. To assess the estimation accuracy of the penalized methods, we also consider the standard polynomial spline estimations of the oracle model (ORACLE). The oracle model only contains the first three relevant variables and is only available in simulation studies where the true information is known.

Table 1 summarizes the simulation results. It gives the relative TAISEs (RTAISE) of the penalized spline methods (TLP, SCAD, LASSO, AdLASSO) to the ORACLE estimator. It also reports the percentage of correct fitting(C), underfitting(U) and overfitting(O) over 200 simulation runs for the penalized methods. When $d = 10$, the performance of the TLP, SCAD, LASSO and AdLASSO are comparable, with TLP being slightly better the rest. But as the dimension d increases, Table 1 clearly shows that the TLP outperforms the other procedures. The percentage of correct fitting for SCAD, LASSO and AdLASSO decreases significantly more when d increases, while the performance of the TLP is relatively stable as d increases. For example, when $d = 400$, the correct fitting is 82.5% for TLP versus 58.5% for SCAD, 18% for LASSO, and 59.5% for AdLASSO in the linear spline. In addition, SCAD, LASSO and AdLASSO also tend to over-fit the model when d increases, for example, when $d = 400$, the over-fitting rate is 37% for SCAD, 81% for LASSO, and 39.5% for AdLASSO versus 14.5% for TLP in the linear spline.

In terms of estimation accuracy, Table 1 shows that the RTAISE of the TLP is close to 1 when d is small. This indicates that the TLP can estimate the nonzero components as accurately as the oracle. But RTAISE increases as d increases, since variable selection becomes more challenging as d increases. Figure 1 plots the typical estimated coefficient functions from ORACLE, TLP and SCAD using linear splines ($p = 1$) when $d = 100$. The typical estimated coefficient functions are

Penalty	d	RTAISE	C	U	O
TLP	10	1.049	0.925	0.005	0.070
SCAD		1.051	0.875	0.010	0.125
LASSO		1.080	0.640	0.000	0.360
AdLASSO		1.061	0.895	0.000	0.105
TLP	100	1.230	0.890	0.030	0.080
SCAD		1.282	0.710	0.030	0.260
LASSO		1.391	0.410	0.000	0.590
AdLASSO		1.283	0.720	0.000	0.280
TLP	200	1.404	0.895	0.035	0.070
SCAD		1.546	0.705	0.035	0.260
LASSO		1.856	0.330	0.015	0.655
AdLASSO		1.509	0.710	0.015	0.275
TLP	400	1.715	0.825	0.030	0.145
SCAD		1.826	0.585	0.045	0.370
LASSO		2.364	0.180	0.010	0.810
AdLASSO		1.879	0.595	0.010	0.395

Table 1: Simulation results for model selection based on various penalty functions: Relative total averaged integrated squared errors (RTAISEs) and the percentages of correct-fitting (C), under-fitting (U) and over-fitting (O) over 200 replications.

those with TAISE being the median of the 200 TAISEs from the simulations. Also plotted are the point-wise 95% confidence intervals from the ORACLE estimation, with the point-wise lower and upper bounds being the 2.5% and 97.5% sample quantiles of the 200 ORACLE estimates. Figure 1 shows that the proposed TLP method estimates the coefficient functions reasonably well. Compared with the SCAD, LASSO and AdLASSO, the TLP method gives better estimation in general, which is consistent with the RTAISEs reported in Table 1.

5.2 Application to AIDS Data

In this subsection, we consider the AIDs data in Huang, Wu and Zhou (2004). The data set consists of 283 homosexual males who were HIV positive between 1984 and 1991. Each patient was scheduled to undergo measurements related to their disease at a semi-annual base visit, but some of them missed or rescheduled their appointments. Therefore, each patient had different measurement times during the study period. It is known that HIV destroys CD4 cells, so by measuring CD4 cell counts and percentages in the blood, patients can be regularly monitored for disease progression. One of the study goals is to evaluate the effects of cigarette smoking status (Smoking), with 1 as smoker and 0 as nonsmoker; pre-HIV infection CD4 cell percentage (Precd4); and age at HIV infection (age), on the CD4 percentage after infection. Let t_{ij} be the time in years of the jth measurement for the ith individual after HIV infection, and y_{ij} be the CD4 percentage of patient i at time t_{ij}. We consider the following varying-coefficient model

$$y_{ij} = \beta_0(t_{ij}) + \beta_1(t_{ij})\text{Smoking} + \beta_2(t_{ij})\text{Age} + \beta_3(t_{ij})\text{Precd4} + \varepsilon_{ij}. \tag{11}$$

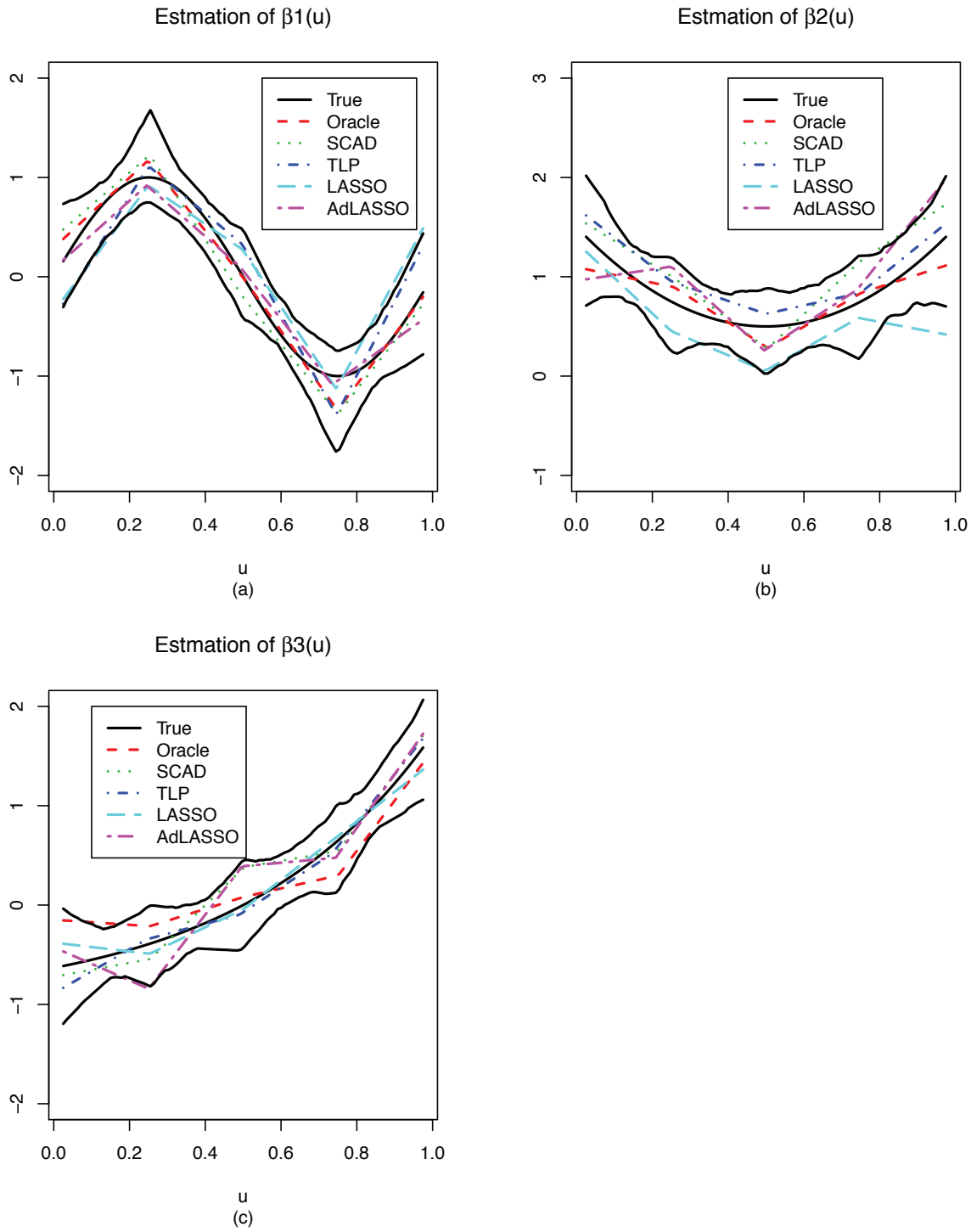


Figure 1: Simulated example: Plots of the estimated coefficient functions for (a) $\beta_1(u)$, (b) $\beta_2(u)$ and (c) $\beta_3(u)$ based on Oracle, SCAD, TLP, LASSO and AdLASSO approaches using linear spline when $d = 100$. In each plot, also plotted are the true curve and the pointwise 95% confidence intervals from the ORACLE estimation.

We apply the proposed penalized cubic spline ($p = 3$) with TLP, SCAD, LASSO and Adaptive LASSO penalties to identify the non-zero coefficient functions. We also consider the standard polynomial spline estimation of the coefficient functions. All four procedures selected two non-zero coefficient functions $\beta_0(t)$ and $\beta_3(t)$, indicating that Smoking and Age have no effect on the CD4 percentage. Figure 2 plots the estimated coefficient functions from the standard cubic spline, SCAD, TLP, LASSO and Adaptive LASSO approaches. For the standard cubic spline estimation, we also calculated the 95% point-wise bootstrap confidence intervals for the coefficient functions based on 500 bootstrapped samples.

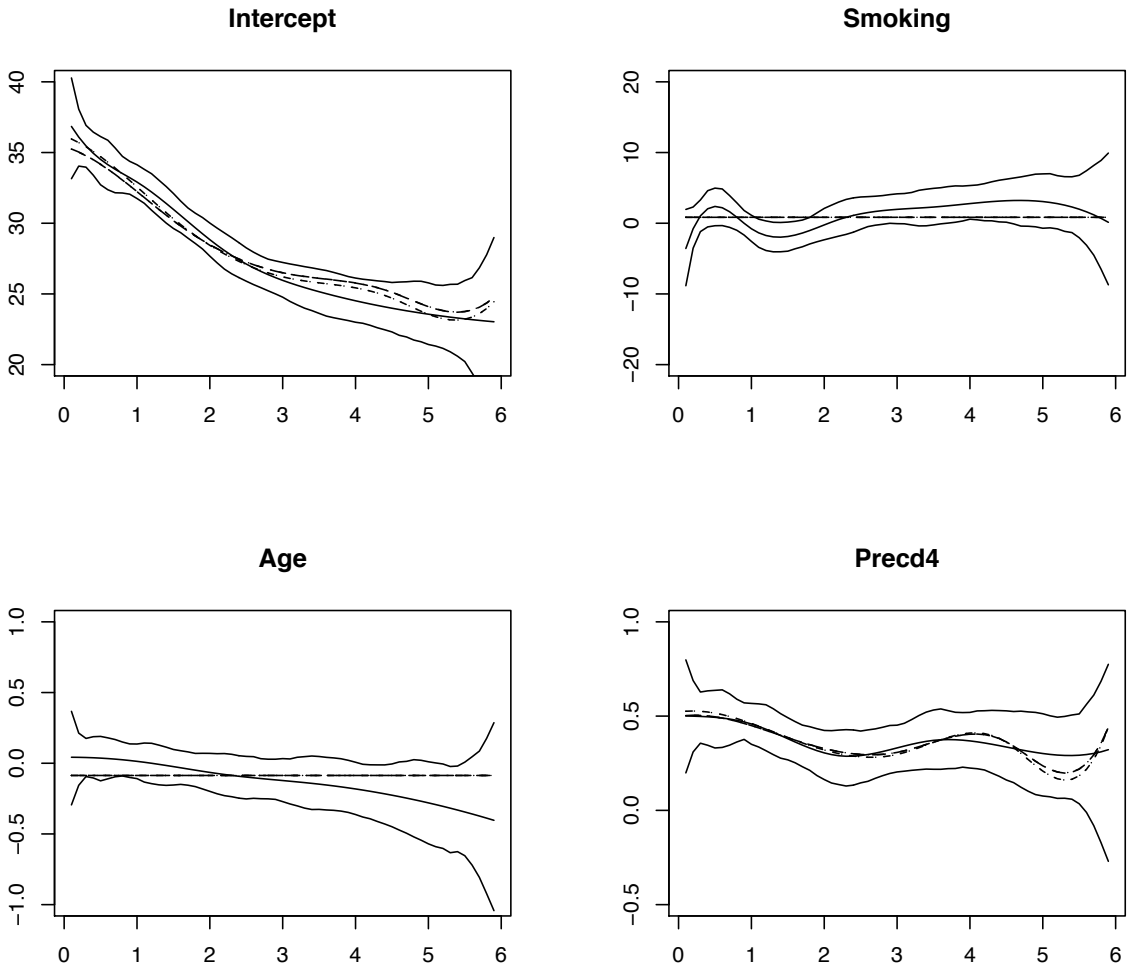


Figure 2: AIDs data: Plots of the estimated coefficient functions using standard cubic spline (line), penalized cubic spline with TLP (dotted), SCAD (dashed), LASSO (dotdash), Adaptive LASSO (long dash) penalties, together with the point-wise 95% bootstrap confidence intervals from the standard cubic spline estimation.

In this example, the dimension of the linear covariates is rather small. In order to evaluate a more challenging situation with higher dimension of d, we introduced an additional 100 redundant linear covariates, which are artificially generated from a Uniform $[0,1]$ distribution independently. We then apply the penalized spline with TLP, SCAD, LASSO or Adaptive LASSO penalties to the augmented data set. We repeated this procedure 100 times. For the three observed variables in model (11), all four procedures always select the Precd4 and never select Smoking and Age. For the 100 artificial covariates, the TLP selects at least one of these artificial covariates only 8 times, while LASSO, Adaptive LASSO, and SCAD select 28, 27, and 42 times respectively. Clearly, LASSO, Adaptive LASSO and SCAD tend to overfit the model and select many more null variables in this data example. Note that our analysis does not incorporate the dependent structure of the repeated measurements. Using the dependent structure of correlated data for high-dimensional settings will be further investigated in our future research.

6. Discussion

We propose simultaneous model selection and parameter estimation for the varying-coefficient model in high-dimensional settings where the dimension of predictors exceeds the sample size. The proposed model selection approach approximates the L_0 penalty effectively, while overcoming the computational difficulty of the L_0 penalty. The key idea is to decompose the non-convex penalty function by taking the difference between two convex functions, therefore transforming a non-convex problem into a convex optimization problem. The main advantage is that the minimization process does not depend on the initial consistent estimators of coefficients, which could be hard to obtain when the dimension of covariates is high. Our simulation and data examples confirm that the proposed model selection performs better than the SCAD in the high-dimensional case.

The model selection consistency property is derived for the proposed method. In addition, we show that it possesses the oracle property when the dimension of covariates exceeds the sample size. Note that the theoretical derivation of asymptotic properties and global optimality results are rather challenging for varying-coefficient model selection, as the dimension of the nonparametric component is also infinite in addition to the high-dimensional covariates.

Shen, Pan and Zhu (2012) provide stronger conditions under which a local minimizer can also achieve the objective of a global minimizer through the penalized truncated L_1 approach. The derivation is based on the normality assumption and the projection theory. For the nonparametric varying-coefficient model, these assumptions are not necessarily satisfied and the projection property cannot be used due to the curse of dimensionality. In general, whether a local minimizer can also hold the global optimality property for the high-dimensional varying-coefficient model requires further investigation. Nevertheless, the DC algorithm yields a better local minimizer compared to the SCAD, and can achieve the global minimum if it is combined with the branch-and-bound method (Liu, Shen and Wong, 2005), although this might be more computationally intensive.

Acknowledgments

Xue's research was supported by the National Science Foundation (DMS-0906739). Qu's research was supported by the National Science Foundation (DMS-0906660). The authors are grateful to

Xinxin Shu's computing support, and the three reviewers and the Action Editor for their insightful comments and suggestions which have improved the manuscript significantly.

Appendix A. Assumptions

To establish the asymptotic properties of the spline TLP estimators, we introduce the following notation and technical assumptions. For a given sample size n, let $\mathbf{Y}_n = (Y_1,\ldots,Y_n)^T$, $\mathbb{X}_n = (\mathbf{X}_1,\ldots,\mathbf{X}_n)^T$ and $\mathbf{U}_n = (U_1\ldots,U_n)^T$. Let $\mathbf{X}_{nj}$ be the j-th column of $\mathbb{X}_n$. Let $\|\cdot\|_2$ be the usual L_2 norm for functions and vectors and $\mathcal{C}^p([a,b])$ be the space of p-times continuously differentiable functions defined on $[a,b]$. For two vectors of the same length $a = (a_1,\ldots,a_d)^T$ and $b = (b_1,\ldots,b_d)^T$, denote $a \circ b = (a_1b_1,\ldots,a_db_d)^T$. For any scalar function $g(\cdot)$ and a vector $a = (a_1,\ldots,a_d)^T$, we denote $g(a) = (g(a_1),\ldots,g(a_d))^T$.

(C1) *The number of relevant linear covariates d_0 is fixed and there exists $\beta_{0j}(\cdot) \in C^p[a,b]$ for some $p \geq 1$ and $j = 1,\ldots,d_0$, such that $E(Y|\mathbf{X},U) = \sum_{j=1}^{d_0} \beta_{0j}(U) X_j$. Furthermore there exists a constant $c_1 > 0$ such that $\min_{1\leq j\leq d_0} E\left[\beta_{0j}^2(U)\right] > c_1$.*

(C2) *The noise ε satisfies $E(\varepsilon) = 0$, $V(\varepsilon) = \sigma^2 < \infty$, and its tail probability satisfies $P(|\varepsilon| > x) \leq c_2 \exp\left(-c_3 x^2\right)$ for all $x \geq 0$ and for some positive constants c_2 and c_3.*

(C3) *The index variable U has a compact support on $[a,b]$ and its density is bounded away from 0 and infinity.*

(C4) *The eigenvalues of matrix $E\left(\mathbf{X}\mathbf{X}^T|U=u\right)$ are bounded away from 0 and infinity uniformly for all $u \in [a,b]$.*

(C5) *There exists a constant $c > 0$ such that $\left|X_j\right| < c$ with probability 1 for $j = 1,\ldots,d$.*

(C6) *The d sets of knots denoted as $\upsilon_j = \left\{a = \upsilon_{j,0} < \upsilon_{j,1} < \cdots < \upsilon_{j,N_n} < \upsilon_{j,N_n+1} = b\right\}, j = 1,\ldots,d$, are quasi-uniform, that is, there exists $c_4 > 0$, such that*

$$\max_{j=1,\ldots,d} \frac{\max\left(\upsilon_{j,l+1} - \upsilon_{j,l}, l = 0,\ldots,N_n\right)}{\min(\upsilon_{j,l+1} - \upsilon_{j,l}, l = 0,\ldots,N_n)} \leq c_4.$$

(C7) *The tuning parameters satisfy*

$$\begin{aligned} \frac{\tau_n}{\lambda_n}\sqrt{\frac{\log(N_nd)}{nN_n}} + \frac{\tau_n N_n^{-(p+2)}}{\lambda_n} &= o(1) \\ \frac{N_n \log(N_nd)}{n} + \tau_n &= o(1). \end{aligned}$$

(C8) *The tuning parameters satisfy*

$$\begin{aligned} \frac{\log(N_nd)N_n}{n\lambda_n} + \frac{n}{\log(N_nd)N_n^{2p+3}} &= o(1) \\ \frac{n\lambda_n}{\log(N_nd)dN_n} + \frac{d\log(n)\tau_n^2}{\lambda_n} &= o(1). \end{aligned}$$

(C9) *For any subset A of* $\{1,\ldots,d\}$, *let*

$$\Delta_n(A) = \min_{\beta_j \in \varphi_j, j \in A} \left\| \sum_{j \in A} \beta_j(\mathbf{U}_n) \circ \mathbf{X}_{nj} - \sum_{j \in A_0} \beta_{0j}(\mathbf{U}_n) \circ \mathbf{X}_{nj} \right\|_2^2.$$

We assume that the model (1) is empirically identifiable in the sense that,

$$\lim_{n\to\infty} \min \left\{ (\log(N_n d) N_n d)^{-1} \Delta_n(A) : A \neq A_0, |A| \leq \alpha d_0 \right\} = \infty,$$

where $\alpha > 1$ is a constant, $|A|$ *denotes the cardinality of* A, and $A_0 = \{1,\ldots,d_0\}$.

The above conditions are commonly assumed in the polynomial spline and variable selection literature. Conditions similar to (C1) and (C2) are also assumed in Huang, Horowitz and Wei (2010). Conditions similar to (C3)-(C6) can be found in Huang, Wu and Zhou (2002) and are needed for estimation consistency even when the dimension of linear covariates d is fixed. Conditions (C7) and (C8) are two different sets of conditions on tuning parameters for the local and global optimality of the spline TLP, respectively. Condition (C9) is analogous to the "degree-of-separation" condition assumed in Shen, Pan and Zhu (2012), and is weaker than the sparse Riesz condition assumed in Wei, Huang and Li (2011).

Appendix B. Outline of Proofs

To establish the asymptotic properties of the proposed estimator, we first investigate the properties of spline functions for high-dimensional data in Lemmas 4-5 and properties of the oracle spline estimators of the coefficient functions in Lemma oracle. The approximation theory for spline functions (De Boor, 2001) plays a key role in these proofs. When the true model is assumed to be known, it reduces to the estimation of the the varying-coefficient model with fixed dimensions. The asymptotic properties of the resulting oracle spline estimators of the coefficient functions have been discussed in the literature.Specifically, Lemma 6 follows directly from Theorems 2 and 3 of Huang, Wu and Zhou (2004).

To prove Theorem 1, we first provide the sufficient conditions for a solution to be a local minimizer for the object function by differentiating the objective function through regular subdifferentials. We then establish Theorem 1 by showing that the oracle estimator satisfies those conditions with probability approaching 1. In Theorem 2, we show that the oracle estimator minimizes the objective function globally with probability approaching 1, thereby establishing that the oracle estimator is also the global optimizer. This is accomplished by showing that the sum of the probabilities of all the other misspecified solutions minimizing the objective function converges to zero as $n \to \infty$.

Appendix C. Technical Lemmas

For any set $A \subset \{1,\ldots,d\}$, we denote $\widetilde{\beta}^{(A)}$ the standard polynomial spline estimator of the model A, that is, $\widetilde{\beta}_j^{(A)} = 0$ if $j \notin A$, and

$$\left(\widetilde{\beta}_j^{(A)}, j \in A\right) = \underset{s_j \in \varphi_j}{\operatorname{argmin}} \frac{1}{2n} \sum_{i=1}^{n} \left[Y_i - \sum_{j \in A} s_j(U_i) X_{ij} \right]^2. \tag{12}$$

In particular, $\widetilde{\beta}^{(o)} = \widetilde{\beta}^{(A_0)}$, with $A_0 = \{1,\ldots,d_0\}$ being the standard polynomial spline estimator of the oracle model.

We first investigate the property of splines. Here we use B-spline basis in the proof, but the results still hold true for other choices of basis. For any $s^{(1)}(u) = \left(s_1^{(1)}(u),\ldots,s_d^{(1)}(u)\right)^T$ and $s^{(2)}(u) = \left(s_1^{(2)}(u),\ldots,s_d^{(2)}(u)\right)^T$ with each $s_j^{(1)}(u), s_j^{(2)}(u) \in S_j$, define the empirical inner product as

$$\left\langle s^{(1)}, s^{(2)} \right\rangle_n = \frac{1}{n}\sum_{i=1}^{n}\left(\sum_{j=1}^{d} s_j^{(1)}(U_i)X_{ij}\right)\left(\sum_{j=1}^{d} s_j^{(2)}(U_i)X_{ij}\right),$$

and theoretical inner product as

$$\left\langle s^{(1)}, s^{(2)} \right\rangle = E\left[\left(\sum_{j=1}^{d} s_j^{(1)}(U)X_j\right)\left(\sum_{j=1}^{d} s_j^{(2)}(U)X_j\right)\right].$$

Denote the induced empirical and theoretical norms as $\|\cdot\|_n$ and $\|\cdot\|$ respectively. Let $\|g\|_\infty = \sup_{x\in[a,b]} g(u)$ be the supremum norm.

Lemma 4 *For any $s_j(u) \in \varphi_j$, write $s_j(u) = \sum_{l=1}^{J_n}\gamma_{jl}B_{jl}(u)$ for $\gamma_j = (\gamma_{j1},\ldots,\gamma_{jJ_n})^T$. Let $\gamma = \left(\gamma_1^T,\ldots,\gamma_d^T\right)^T$ and $\mathbf{s}(u) = (s_1(u),\ldots,s_d(u))^T$. Then there exist constants $0 < c \le C$ such that*

$$c\|\gamma\|_2^2/N_n \le \|\mathbf{s}\|^2 \le C\|\gamma\|_2^2/N_n.$$

Proof: Note that

$$\begin{aligned}\|\mathbf{s}\|^2 &= E\left[\left(\sum_{j=1}^{d} s_j(U)X_j\right)^2\right] = E\left[\mathbf{s}^T(U)\mathbf{X}\mathbf{X}^T\mathbf{s}(U)\right] \\ &= E\left[\mathbf{s}^T(U)E\left\{\mathbf{X}\mathbf{X}^T|U\right\}\mathbf{s}(U)\right].\end{aligned}$$

Therefore by (C4), there exist $0 < c_1 \le c_2$, such that

$$c_1 E\left[\mathbf{s}^T(U)\mathbf{s}(U)\right] \le \|\mathbf{s}\|^2 \le c_2 E\left[\mathbf{s}^T(U)\mathbf{s}(U)\right],$$

in which, by properties of B-spline basis functions, there exist $0 < c_1^* \le c_2^*$, such that

$$c_1^*\sum_{j=1}^{d}\left\|\gamma_j\right\|_2^2/N_n \le E\left[\mathbf{s}^T(U)\mathbf{s}(U)\right] = \sum_{j=1}^{d} E\left[s_j^2(U)\right] \le c_2^*\sum_{j=1}^{d}\left\|\gamma_j\right\|_2^2/N_n.$$

The conclusion follows by taking $c = c_1c_1^*$, and $C = c_2c_2^*$.

For any $A \subset \{1,\ldots,d\}$, let $|A|$ be the cardinality of A. Denote $\mathbf{Z}_A = (\mathbf{Z}_j, j \in A)$ and $\mathbf{D}_A = \mathbf{Z}_A^T\mathbf{Z}_A/n$. Let $\rho_{\min}(\mathbf{D}_A)$ and $\rho_{\max}(\mathbf{D}_A)$ be the minimum and maximum eigenvalues of $\mathbf{D}_A$ respectively. ■

Lemma 5 *Suppose that $|A|$ is bounded by a fixed constant independent of n and d. Then under conditions (C3)-(C5), one has*

$$c_1/N_n \le \rho_{\min}(\mathbf{D}_A) \le \rho_{\max}(\mathbf{D}_A) \le c_2/N_n,$$

for some constants $c_1, c_2 > 0$.

Proof: Without loss of generality, we assume $A = \{1, \ldots, k\}$ for some constant k which does not depend on n nor d. Note that for any $\gamma_A = (\gamma_j, j \in A)$, the triangular inequality gives

$$\gamma_A^T \mathbf{D}_A \gamma_A = \frac{1}{n} \left\| \sum_{j \in A} \mathbf{Z}_j \gamma_j \right\|_2^2 \leq \frac{2}{n} \sum_{j \in A} \left\| \mathbf{Z}_j \gamma_j \right\|_2^2 = 2 \sum_{j \in A} \gamma_j^T \mathbf{D}_j \gamma_j,$$

where $\mathbf{D}_j = \mathbf{Z}_j^T \mathbf{Z}_j / n$. By Lemma 6.2 of Zhou, Shen and Wolfe (1998), there exist constants $c_3, c_4 > 0$ $c_3/N_n \leq \rho_{\min}(\mathbf{D}_j) \leq \rho_{\max}(\mathbf{D}_j) \leq c_4/N_n$. Therefore $\gamma_A^T \mathbf{D}_A \gamma_A \leq 2c_4 \gamma_A^T \gamma_A / N_n$. That is $\rho_{\max}(\mathbf{D}_A) \leq 2c_4/N_n = c_2/N_n$. The lower bound follows from Lemma A.5 in Xue and Yang (2006) with $d_2 = 1$.

Now we consider properties of the oracle spline estimators of the coefficient functions when the true model is known. That is, $\widehat{\beta}^{(o)} = \left(\widehat{\beta}_1^{(o)}, \ldots, \widehat{\beta}_{d_0}^{(o)}, 0, \ldots, 0 \right)$ is the polynomial spline estimator of coefficient functions knowing only that the first d_0 covariates are relevant. That is

$$\left(\widehat{\beta}_1^{(o)}, \ldots, \widehat{\beta}_{d_0}^{(o)} \right)^T = \underset{s_j \in \varphi_j}{\operatorname{argmin}} \sum_{i=1}^{n} \left[Y_i - \sum_{j=1}^{d_0} s_j(U_i) X_{ij} \right]^2.$$

■

Lemma 6 *Suppose conditions (C1)-(C6) hold. If* $\lim N_n \log N_n / n = 0$*, then for* $j = 1, \ldots, d_0$,

$$\begin{aligned} E\left(\beta_j(U) - \widehat{\beta}_j^{(o)}(U) \right)^2 &= O_p\left(\frac{N_n}{n} + N_n^{-2(p+1)} \right), \\ \frac{1}{n} \sum_{i=1}^{n} \left(\beta_j(U_i) - \widehat{\beta}_j^{(o)}(U_i) \right)^2 &= O_p\left(\frac{N_n}{n} + N_n^{-2(p+1)} \right), \end{aligned}$$

and

$$\left\{ V\left(\widehat{\beta}^{(o,1)}(u) \right) \right\}^{-1/2} \left(\widehat{\beta}^{(o,1)}(u) - \beta^{(1)}(u) \right) \to N(0, \mathbf{I})$$

in distribution, where $\widehat{\beta}^{(o,1)}(u) = \left(\widehat{\beta}_1^{(o)}(u), \ldots, \widehat{\beta}_{d_0}^{(o)}(u) \right)^T$*, and* $\beta^{(1)}(u) = (\beta_1(u), \ldots, \beta_{d_0}(u))^T$*, and*

$$V\left(\widehat{\beta}^{(o,1)}(u) \right) = \mathbf{B}^{(1)}(u) \left(\sum_{i=1}^{n} \mathbf{A}_i^{(1)T} \mathbf{A}_i^{(1)} \right)^{-1} \mathbf{B}^{(1)}(u) = O_p(N_n/n),$$

where $\mathbf{B}^{(1)}(u) = \left(\mathbf{B}_1^T(u), \ldots, \mathbf{B}_{d_0}^T(u) \right)^T$*, and* $\mathbf{A}_i^{(1)} = \left(\mathbf{B}_1^T(U_i) X_{i1}, \ldots, \mathbf{B}_{d_0}^T(U_i) X_{id_0} \right)^T$ *in which* $\mathbf{B}_j^T(U_i) X_{ij} = (B_{j1}(U_i) X_{ij}, \ldots, B_{jJ_n}(U_i) X_{ij})$.

Proof: It follows from Theorems 2 and 3 of Huang, Wu and Zhou (2004). ■

Lemma 7 *Suppose conditions (C1)-(C6) hold. Let* $T_{jl} = \sqrt{N_n/n} \sum_{i=1}^{n} B_{jl}(U_i) X_{ij} \varepsilon_i$*, for* $j = 1, \ldots, d$*, and* $l = 1, \ldots, J_n$*. Let* $T_n = \max_{1 \leq j \leq d, 1 \leq l \leq J_n} |T_{jl}|$*. If* $N_n \log(N_n d)/n \to 0$*, then*

$$E(T_n) = O\left(\sqrt{\log(N_n d)} \right).$$

Proof: Let $m_{jl}^2 = \sum_{i=1}^{n} B_{jl}^2(U_i) X_{ij}^2$, and $m_n^2 = \max_{1\le j\le d, 1\le l\le J_n} m_{jl}^2$. By condition (C2) and the maximal inequality for gaussian random variables, there exists a constant $C_1 > 0$ such that

$$E(T_n) = E\left(\max_{1\le j\le d, 1\le l\le J_n} |T_{jl}|\right) \le C_1\sqrt{N_n/n}\sqrt{\log(N_n d)}E(m_n). \tag{13}$$

Furthermore, by the definition of B-spline basis and (C5), there exists a $C_2 > 0$, such that for each $1 \le j \le d, 1 \le l \le J_n$,

$$\left|B_{jl}^2(U_i) X_{ij}^2\right| \le C_2, \text{ and } E\left[B_{jl}^2(U_i) X_{ij}^2\right] \le C_2 N_n^{-1}.$$

As a result,

$$\sum_{i=1}^{n} E\left[B_{jl}^2(U_i) X_{ij}^2 - E\left(B_{jl}^2(U_i) X_{ij}^2\right)\right]^2 \le 4C_2 n N_n^{-1},$$

and

$$\max_{1\le j\le d, 1\le l\le J_n} E m_{jl}^2 = \max_{1\le j\le d, 1\le l\le J_n} \sum_{i=1}^{n} E\left(B_{jl}^2(U_i) X_{ij}^2\right) \le C_2 n N_n^{-1}. \tag{14}$$

Then by Lemma A.1 of Van de Geer (2008), one has

$$\begin{aligned} & E\left(\max_{1\le j\le d, 1\le l\le J_n} \left|m_{jl}^2 - E m_{jl}^2\right|\right) \\ = \; & E\left(\max_{1\le j\le d, 1\le l\le J_n} \left|\sum_{i=1}^{n} B_{jl}^2(U_i) X_{ij}^2 - E\left(B_{jl}^2(U_i) X_{ij}^2\right)\right|\right) \\ \le \; & \sqrt{2C_2 n N_n^{-1}\log(N_n d)} + 4\log(2N_n d). \end{aligned} \tag{15}$$

Therefore (14) and (15) give that

$$\begin{aligned} E m_n^2 &\le \max_{1\le j\le d, 1\le l\le J_n} E m_{jl}^2 + E\left(\max_{1\le j\le d, 1\le l\le J_n} \left|m_{jl}^2 - E m_{jl}^2\right|\right) \\ &\le C_2 n N_n^{-1} + \sqrt{2C_2 n N_n^{-1}\log(N_n d)} + 4\log(2N_n d). \end{aligned}$$

Furthermore, $E m_n \le \sqrt{E m_n^2} \le \left(\sqrt{2C_2 n N_n^{-1}\log(N_n d)} + 4\log(2dN_n) + C_2 n N_n^{-1}\right)^{1/2}$. Together with (13) and $N_n \log(N_n d)/n \to 0$, one has

$$\begin{aligned} E(T_n) &\le C_1\sqrt{N_n/n}\sqrt{\log(N_n d)}\left(\sqrt{2C_2 n N_n^{-1}\log(N_n d)} + 4\log(2N_n d) + C_2 n N_n^{-1}\right)^{1/2} \\ &= O\left(\sqrt{\log(N_n d)}\right). \end{aligned}$$

■

Lemma 8 *Suppose conditions (C1)-(C7) hold. Let* $\mathbf{Z}_j = (\mathbf{Z}_{1j}, \ldots, \mathbf{Z}_{nj})^T$, $\mathbf{Y} = (Y_1, \ldots, Y_n)^T$, *and* $\mathbf{Z}_{(1)} = (\mathbf{Z}_1, \ldots, \mathbf{Z}_{d_0})$. *Then*

$$P\left(\left\|\frac{1}{n}\mathbf{Z}_j^T\left(\mathbf{Y} - \mathbf{Z}_{(1)}\widehat{\gamma}^{(o,1)}\right)\right\|_{W_j} > \frac{\lambda_n}{\tau_n}, \; \exists\, j = d_0+1, \ldots, d\right) \to 0.$$

Proof: By the approximation theory (de Boor 2001, p. 149), there exist a constant $c > 0$ and spline functions $s_j^0 = \sum_{l=1}^{J_n} \gamma_{jl}^0 B_{jl}(t) \in S_j$, such that

$$\max_{1 \le j \le d_0} \left\| \beta_j - s_j^0 \right\|_\infty \le c N_n^{-(p+1)}. \tag{16}$$

Let $\delta_i = \sum_{j=1}^{d_0} \left[\beta_j(U_i) - s_j^0(U_i) \right] X_{ij}$, $\delta = (\delta_1, \ldots, \delta_n)^T$, and $\varepsilon = (\varepsilon_1, \ldots, \varepsilon_n)^T$. Then one has

$$\mathbf{Z}_j^T \left(\mathbf{Y} - \mathbf{Z}_{(1)} \widehat{\gamma}^{(o,1)} \right) = \mathbf{Z}_j^T \mathbf{H}_n \mathbf{Y} = \mathbf{Z}_j^T \mathbf{H}_n \varepsilon + \mathbf{Z}_j^T \mathbf{H}_n \delta,$$

where $\mathbf{H}_n = \mathbf{I} - \mathbf{Z}_{(1)} \left(\mathbf{Z}_{(1)}^T \mathbf{Z}_{(1)} \right)^{-1} \mathbf{Z}_{(1)}^T$. By Lemma 7, there exists a $c > 0$ such that

$$E \left(\max_{d_0+1 \le j \le d} \left\| \mathbf{Z}_j^T \mathbf{H}_n \varepsilon \right\|_{W_j} \right) \le c \sqrt{n \log(N_n d) / N_n}.$$

Therefore by Markov's inequality, one has

$$\begin{aligned} & P \left(\left\| \mathbf{Z}_j^T \mathbf{H}_n \varepsilon \right\|_{W_j} > \frac{n \lambda_n}{2 \tau_n}, \ \exists \ j = d_0 + 1, \ldots, d \right) = P \left(\max_{d_0+1 \le j \le d} \left\| \mathbf{Z}_j^T \mathbf{H}_n \varepsilon \right\|_{W_j} > \frac{n \lambda_n}{2 \tau_n} \right) \\ \le & \quad \frac{2c\tau_n}{\lambda_n} \sqrt{\frac{\log(N_n d)}{n N_n}} \to 0, \end{aligned} \tag{17}$$

as $n \to \infty$, by condition (C7). On the other hand, let ρ_j and ρ_{H_n} be the largest eigenvalue of $\mathbf{Z}_j^T \mathbf{Z}_j / n$ and $\mathbf{H}_n$. Then Lemma (5) entails that $\max_{d_0+1 \le j \le d} \rho_j = O_p(1/N_n)$. Together with (16) and condition (C7), one has

$$\begin{aligned} \max_{d_0+1 \le j \le d} \frac{1}{n} \left\| \mathbf{Z}_j^T \mathbf{H}_n \delta \right\|_{W_j} & \le (n N_n)^{-1/2} \sqrt{\max_{d_0+1 \le j \le d} \rho_j \rho_{H_n}} \|\delta\|_2 \\ & = O_p \left(N_n^{-(p+1)} / N_n \right) = o_p \left(\frac{\lambda_n}{2 \tau_n} \right). \end{aligned} \tag{18}$$

Then the lemma follows from (17) and (18) and by noting that

$$\begin{aligned} & P \left(\left\| \frac{1}{n} \mathbf{Z}_j^T \left(\mathbf{Y} - \mathbf{Z}_{(1)} \widehat{\gamma}^{(o,1)} \right) \right\|_{W_j} > \frac{\lambda_n}{\tau_n}, \ \exists \ j = d_0 + 1, \ldots, d \right) \\ \le & \quad P \left(\max_{d_0+1 \le j \le d} \frac{1}{n} \left\| \mathbf{Z}_j^T \mathbf{H}_n \varepsilon \right\|_{W_j} > \frac{\lambda_n}{2 \tau_n} \right) + P \left(\max_{d_0+1 \le j \le d} \frac{1}{n} \left\| \mathbf{Z}_j^T \mathbf{H}_n \delta \right\|_{W_j} > \frac{\lambda_n}{2 \tau_n} \right). \end{aligned}$$

■

Appendix D. Proof of Theorem 1

For notation simplicity, let $\mathbf{Z}_{ij}^* = \mathbf{W}_j^{-1/2} \mathbf{Z}_{ij}$ and $\gamma_j^* = \mathbf{W}_j^{1/2} \gamma_j$. Then the minimization problem in (4) becomes

$$L_n(\gamma^*) = \frac{1}{2n} \sum_{i=1}^n \left[Y_i - \sum_{j=1}^d \gamma_j^{*T} \mathbf{Z}_{ij}^* \right]^2 + \lambda_n \sum_{j=1}^d p_n \left(\left\| \gamma_j^* \right\|_2 \right).$$

For $i=1,\ldots,n$, and $j=1,\ldots,d$, write $\mathbf{Z}_i^*=\left(\mathbf{Z}_{i1}^{*T},\ldots,\mathbf{Z}_{id}^{*T}\right)^T$, $\gamma^*=\left(\gamma_1^{*T},\ldots,\gamma_d^{*T}\right)^T$ and $c_j^*(\gamma^*)=-\frac{1}{n}\sum_{i=1}^n\mathbf{Z}_{ij}^*\left(Y_i-\mathbf{Z}_i^{*T}\gamma^*\right)$. Differentiate $L_n(\gamma^*)$ with respect to γ_j^* through regular subdifferentials, we obtain the local optimality condition for $L_n(\gamma^*)$ as $c_j^*(\gamma^*)+\frac{\lambda_n}{\tau_n}\zeta_j=\mathbf{0}$, where $\zeta_j=\gamma_j^*/\left\|\gamma_j^*\right\|_2$ if $0<\left\|\gamma_j^*\right\|_2<\tau_n$; $\zeta_j=\{\gamma_j^*,\left\|\gamma_j^*\right\|_2\le 1\}$ if $\left\|\gamma_j^*\right\|_2=0$;$\zeta_j=\mathbf{0}$, if $\left\|\gamma_j^*\right\|_2>\tau_n$; and $\zeta_j=\emptyset$, if $\left\|\gamma_j^*\right\|_2=\tau_n$, where $\emptyset$ is an empty set. Therefore any γ^* that satisfies

$$\begin{aligned} c_j^*(\gamma^*) &= \mathbf{0}, \qquad \left\|\gamma_j^*\right\|>\tau_n \text{ for } j=1,\ldots,d_0.\\ \left\|c_j^*(\gamma^*)\right\|_2 &\le \frac{\lambda_n}{\tau_n}, \qquad \left\|\gamma_j^*\right\|=0 \text{ for } j=d_0+1,\ldots,d, \end{aligned}$$

is a local minimizer of $L_n(\gamma^*)$. Or equivalently, any γ that satisfies

$$c_j(\gamma) = \mathbf{0}, \qquad \left\|\gamma_j\right\|_{W_j}>\tau_n \text{ for } j=1,\ldots,d_0. \tag{19}$$

$$\left\|c_j(\gamma)\right\|_{W_j} \le \frac{\lambda_n}{\tau_n}, \qquad \left\|\gamma_j\right\|_{W_j}=0 \text{ for } j=d_0+1,\ldots,d, \tag{20}$$

is a local minimizer of $L_n(\gamma)$, in which $c_j(\gamma)=-\frac{1}{n}\sum_{i=1}^n\mathbf{Z}_{ij}\left(Y_i-\mathbf{Z}_i^T\gamma\right)$. Therefore it suffices to show that $\widehat{\gamma}^{(o)}$ satisfies (19) and (20).

For $j=1,\ldots,d_0$, $c_j\left(\widehat{\gamma}^{(o)}\right)=\mathbf{0}$ trivially by the definition of $\widehat{\gamma}^{(o)}$. On the other hand, conditions (C1), (C7) and Lemma 6 give that

$$\lim_{n\to\infty}P\left(\left\|\widehat{\gamma}_j^{(o)}\right\|_{W_j}>\tau_n, j=1,\ldots,d_0\right)=1.$$

Therefore $\widehat{\gamma}^{(o)}$ satisfies (19). For (20), note that, by definition $\widehat{\gamma}_j^{(o)}=0$, for $j=d_0+1,\ldots,d$. Furthermore, for $j=d_0+1,\ldots,d$,

$$c_j\left(\widehat{\gamma}^{(o)}\right)=-\frac{1}{n}\mathbf{Z}_j^T\left(\mathbf{Y}-\mathbf{Z}_{(1)}\widehat{\gamma}^{(o,1)}\right).$$

By Lemma 8,

$$P\left(\left\|c_j\left(\widehat{\gamma}^{(o)}\right)\right\|_{W_j}>\frac{\lambda_n}{\tau_n},\ \exists\ j=d_0+1,\ldots,d\right)\to 0.$$

Therefore $\widehat{\gamma}_j^{(o)}$ also satisfies (20) with probability approaching to 1. As a result, $\widehat{\gamma}^{(o)}$ is a local minimum of $L_n(\gamma)$ with probability approaching to 1. ■

Appendix E. Proof of Theorem 2

Note that for any $\gamma=\left(\gamma_1^T,\ldots,\gamma_d^T\right)^T$, one can write

$$\begin{aligned} L_n(\gamma) &= \frac{1}{2n}\sum_{i=1}^n\left[Y_i-\sum_{j=1}^d\gamma_j^T\mathbf{Z}_{ij}\right]^2+\lambda_n\sum_{j=1}^d\min\left(\left\|\gamma_j\right\|_{W_j}/\tau_n,1\right)\\ &= \frac{1}{2n}\sum_{i=1}^n\left[Y_i-\sum_{j=1}^d\gamma_j^T\mathbf{Z}_{ij}\right]^2+\lambda_n|A|+\frac{\lambda_n}{\tau_n}\sum_{j\in A^c}\left\|\gamma_j\right\|_{W_j}, \end{aligned}$$

where $A = A(\gamma) = \left\{ j : \left\|\gamma_j\right\|_{W_j} \geq \tau_n \right\}$, $A^c = \left\{ j : \left\|\gamma_j\right\|_{W_j} < \tau_n \right\}$, and $|A|$ denotes the cardinality of A. For a given set A, let $\widetilde{\gamma}^{(A)}$ be the coefficient from the standard polynomial spline estimation of the model A as defined in (12). Then for $a = \lambda_n / \left(d\tau_n^2 \log n\right) + 1 > 1$, one has

$$\begin{aligned}
& L_n(\gamma) - \lambda_n |A| \\
= & \frac{1}{2n} \sum_{i=1}^{n} \left[Y_i - \sum_{j \in A} \gamma_j^T \mathbf{Z}_{ij} - \sum_{j \in A^c} \gamma_j^T \mathbf{Z}_{ij} \right]^2 + \frac{\lambda_n}{\tau_n} \sum_{j \in A^c} \left\|\gamma_j\right\|_{W_j} \\
\geq & \frac{a-1}{2an} \sum_{i=1}^{n} \left[Y_i - \sum_{j \in A} \gamma_j^T \mathbf{Z}_{ij} \right]^2 - \frac{a-1}{2n} \sum_{i=1}^{n} \left[\sum_{j \in A^c} \gamma_j^T \mathbf{Z}_{ij} \right]^2 + \frac{\lambda_n}{\tau_n} \sum_{j \in A^c} \left\|\gamma_j\right\|_{W_j} \\
\geq & \frac{a-1}{2an} \sum_{i=1}^{n} \left[Y_i - \sum_{j=1}^{d} \widetilde{\gamma}_j^{(A)T} \mathbf{Z}_{ij} \right]^2 - \frac{d(a-1)}{2n} \sum_{i=1}^{n} \sum_{j \in A^c} \left(\gamma_j^T \mathbf{Z}_{ij}\right)^2 + \frac{\lambda_n}{\tau_n} \sum_{j \in A^c} \left\|\gamma_j\right\|_{W_j} \\
\geq & \frac{a-1}{2an} \sum_{i=1}^{n} \left[Y_i - \sum_{j=1}^{d} \widetilde{\gamma}_j^{(A)T} \mathbf{Z}_{ij} \right]^2 + \left(\frac{\lambda_n}{\tau_n} - \frac{a-1}{2} d\tau_n \right) \sum_{j \in A^c} \left\|\gamma_j\right\|_{W_j}.
\end{aligned}$$

Note that $\frac{\lambda_n}{\tau_n} - \frac{a-1}{2} d\tau_n > 0$ for sufficiently large n by the definition of a. Therefore,

$$L_n(\gamma) \geq \frac{a-1}{2an} \sum_{i=1}^{n} \left[Y_i - \sum_{j=1}^{d} \widetilde{\gamma}_j^{(A)T} \mathbf{Z}_{ij} \right]^2 + \lambda_n |A|. \tag{21}$$

Let $\Gamma_1 = \{A : A \subset \{1,\ldots,d\}, A_0 \subset A, \text{ and } A \neq A_0\}$ be the set of overfitting models and $\Gamma_2 = \{A : A \subset \{1,\ldots,d\}, A_0 \not\subset A \text{ and } A \neq A_0\}$ be the set of underfitting models. For any γ, $A(\gamma)$ must fall into one of Γ_j, $j = 1,2$. We now show that

$$\sum_{A \in \Gamma_j} P\left(\min_{\gamma : \mathbf{A}(\gamma) = A} L_n(\gamma) - L_n\left(\widetilde{\gamma}^{(o)}\right) \leq 0 \right) \to 0,$$

as $n \to \infty$, for $j = 1,2$.

Let $\mathbf{Z}(A) = (\mathbf{Z}_j, j \in A)$ and $H_n(A) = \mathbf{Z}(A)\left[\mathbf{Z}^T(A)\mathbf{Z}(A)\right]^{-1}\mathbf{Z}(A)$. Let $\mathbf{E} = (\varepsilon_1,\ldots,\varepsilon_n)^T$, $\mathbf{Y} = (Y_1,\ldots,Y_n)^T$, $m(\mathbf{X}_i, U_i) = \sum_{j=1}^{d} \beta_j(U_i) X_{ij}$ and $\mathbf{M} = (m(\mathbf{X}_1,U_1),\ldots,m(\mathbf{X}_n,U_n))^T$. Lemma 6 entails that $P\left(\min_{j=1,\ldots,d_0} \left\|\widetilde{\gamma}_j^{(o)}\right\|_{W_j} \geq \tau_n \right) \to 1$, as $n \to \infty$. Therefore it follows from (21) that, with probability approaching to one,

$$\begin{aligned}
& 2n\left\{ L_n(\gamma) - L_n\left(\widetilde{\gamma}^{(o)}\right) - \lambda_n(|A| - d_0) \right\} \\
\geq & -\mathbf{Y}^T(\mathbf{H}_n(A) - \mathbf{H}_n(A_0))\mathbf{Y} - \frac{1}{a}\mathbf{Y}^T(\mathbf{I}_n - \mathbf{H}_n(A))\mathbf{Y} \\
= & -\mathbf{E}^T(\mathbf{H}_n(A) - \mathbf{H}_n(A_0))\mathbf{E} - \mathbf{M}^T(\mathbf{H}_n(A) - \mathbf{H}_n(A_0))\mathbf{M} \\
& -2\mathbf{E}^T(\mathbf{H}_n(A) - \mathbf{H}_n(A_0))\mathbf{M} - \frac{1}{a}\mathbf{Y}^T(\mathbf{I}_n - \mathbf{H}_n(A))\mathbf{Y} \\
= & -\mathbf{E}^T(\mathbf{H}_n(A) - \mathbf{H}_n(A_0))\mathbf{E} + I_{n1} + I_{n2} + I_{n3}.
\end{aligned}$$

Let $r(A)$ and $r(A_0)$ be the ranks of $\mathbf{H}_n(A)$ and $\mathbf{H}_n(A_0)$ respectively, and $I_n = I_{n1} + I_{n2} + I_{n3}$. Also note that if $T_m \sim \chi^2_m$, then the Cramer-Chernoff bound gives that $P(T_m - m > km) \leq \exp\left\{-\frac{m}{2}(k - \log(1+k))\right\}$ for some constant $k > 0$. Then one has,

$$\begin{aligned}
& P\left\{L_n(\gamma) - L_n\left(\widetilde{\gamma}^{(o)}\right) < 0\right\} \\
= \; & P\left\{\mathbf{E}^T\left(\mathbf{H}_n(A) - \mathbf{H}_n(A_0)\right)\mathbf{E} > I_n + 2n\lambda_n\left(|A| - d_0\right)\right\} \\
= \; & P\left\{\chi^2_{r(A)-r(A_0)} > I_n + 2n\lambda_n\left(|A| - d_0\right)\right\} \\
\leq \; & \exp\left\{-\frac{r(A) - r(A_0)}{2}\left[\frac{I_n + 2n\lambda_n\left(|A| - d_0\right)}{r(A) - r(A_0)} - 1 - \log\frac{I_n + 2n\lambda_n\left(|A| - d_0\right)}{r(A) - r(A_0)}\right]\right\} \\
\leq \; & \exp\left\{-\frac{r(A) - r(A_0)}{2}\left[\frac{I_n + 2n\lambda_n\left(|A| - d_0\right)}{r(A) - r(A_0)} - 1\right]\frac{1+c}{2}\right\} \qquad (22)
\end{aligned}$$

for some $0 < c < 1$. To bound (22), we consider the following two cases. Case 1 (overfitting): $A = A(\gamma) \in \Gamma_1$. Let $k = |A| - d_0$. By the spline approximation theorem (de Boor, 2001), there exist spline functions $s_j \in \varphi_j$ and constant c such that $\max_{1 \leq j \leq d_0} \left\|\beta_j - s_j\right\|_\infty \leq cN_n^{-(p+1)}$. Let $m^*(\mathbf{X}, U) = \sum_{j=1}^{d_0} s_j(U) X_j$, and $\mathbf{M}^* = \left(m^*(\mathbf{X}_1, U_1), \ldots, m^*(\mathbf{X}_n, U_n)\right)^T$. Then by the definition of projection

$$\frac{1}{n}\mathbf{M}^T\left(\mathbf{I}_n - \mathbf{H}_n(A_0)\right)\mathbf{M} \leq \|m - m^*\|_n^2 \leq cd_0N_n^{-2(p+1)}.$$

Similarly, one can show $\frac{1}{n}\mathbf{M}^T\left(\mathbf{I}_n - \mathbf{H}_n(A)\right)\mathbf{M} \leq c|A|N_n^{-2(p+1)}$. Therefore, by condition (C8)

$$I_{n1} = \mathbf{M}^T\left(\mathbf{I}_n - \mathbf{H}_n(A)\right)\mathbf{M} - \mathbf{M}^T\left(\mathbf{I}_n - \mathbf{H}_n(A_0)\right)\mathbf{M} \leq ckN_n^{-2(p+1)}n = o_p\left(k\log\left(dN_n\right)N_n\right).$$

Furthermore, the Cauchy-Schwartz inequality gives that,

$$\begin{aligned}
|I_{n2}| \; & \leq \; 2\sqrt{\mathbf{E}^T\left(\mathbf{H}_n(A) - \mathbf{H}_n(A_0)\right)\mathbf{E}}\sqrt{\mathbf{M}^T\left(\mathbf{H}_n(A) - \mathbf{H}_n(A_0)\right)\mathbf{M}} \\
& = \; O_p\left(k\sqrt{\log\left(dN_n\right)N_n n}N_n^{-(p+1)}\right) = o_p\left(k\log\left(dN_n\right)N_n\right).
\end{aligned}$$

Finally $I_{n3} = -\frac{1}{a}\mathbf{Y}^T\left(\mathbf{I}_n - \mathbf{H}_n(A)\right)\mathbf{Y} = o_p\left(k\log\left(dN_n\right)N_n\right)$, since $a \to \infty$ as $n \to \infty$ by condition (C8). Therefore, $I_n = I_{n1} + I_{n2} + I_{n3} = o_p\left(k\log\left(dN_n\right)N_n\right)$. As a result, (22) gives that,

$$\begin{aligned}
& \sum_{A(\gamma)\in\Gamma_1} P\left(\min_\gamma L_n(\gamma) - L_n\left(\widetilde{\gamma}^{(o)}\right) \leq 0\right) \\
\leq \; & \sum_{k=1}^{d-d_0}\left(\begin{array}{c} d-d_0 \\ k \end{array}\right)\exp\left\{-\frac{r(A) - r(A_0)}{2}\left[\frac{I_n + 2n\lambda_n k}{r(A) - r(A_0)} - 1\right]\frac{1+c}{2}\right\} \\
\leq \; & \sum_{k=1}^{d-d_0} d^k \exp\left\{-\frac{1+c}{4}\left[I_n + 2n\lambda_n k - \left(r(A) - r(A_0)\right)\right]\right\} \\
= \; & \sum_{k=1}^{d-d_0} \exp\left\{-\frac{1+c}{4}\left[I_n + 2n\lambda_n k - \left(r(A) - r(A_0)\right)\right] + k\log d\right\}
\end{aligned}$$

in which $2n\lambda_n k$ is the dominated term inside of the exponential under condition (C8). Therefore,

$$\begin{aligned}
&\sum_{A(\gamma)\in\Gamma_1} P\left(\min_{\gamma} L_n(\gamma) - L_n\left(\widetilde{\gamma}^{(o)}\right) \le 0\right) \\
\le\ & \sum_{k=1}^{d-d_0} \exp\left\{-\frac{n\lambda_n k}{2}\right\} = \exp\left\{-\frac{n\lambda_n}{2}\right\} \frac{1-\exp\left(-\frac{n(d-d_0)\lambda_n}{2}\right)}{1-\exp\left(-\frac{n\lambda_n}{2}\right)} \to 0 \qquad (23)
\end{aligned}$$

as $n\to\infty$, by condition (C8).

Case 2 (underfitting): $A = A(\gamma) \in \Gamma_2$. Note that,

$$I_{n1} = \mathbf{M}^T(\mathbf{I}_n - \mathbf{H}_n(A))\mathbf{M} - \mathbf{M}^T(\mathbf{I}_n - \mathbf{H}_n(A_0))\mathbf{M} = I_{n1}^{(1)} - I_{n1}^{(2)},$$

in which

$$I_{n1}^{(1)} = \mathbf{M}^T(\mathbf{I}_n - \mathbf{H}_n(A))\mathbf{M} \ge \Delta_n(A).$$

Therefore for any γ with $A_0 \not\subset A$ and $|A| \le \alpha d_0$ where $\alpha > 1$ is a constant as given in condition (C9), the empirically identifiable condition entails that, $(\log(N_n d) N_n d)^{-1} I_{n1}^{(1)} \to \infty$,as $n\to\infty$. On the other hand, similar arguments for Case 1 give that $I_{n1}^{(2)} = O_p\left(d_0 N_n^{-2(p+1)} n\right) = o_p(\log(N_n d) N_n d)$, and $I_{n2} + I_{n3} = O_p(\log(N_n d) N_n d)$. Therefore $I_{n1}^{(1)}$ is the dominated term in I_n. As a result, together with (22), one has

$$P\left\{L_n(\gamma) - L_n\left(\widetilde{\gamma}^{(o)}\right) < 0\right\} \le \exp\left\{-\frac{1+c}{4}\left[\frac{I_{n1}^{(1)}}{2} + 2n\lambda_n(|A| - d_0) - (r(A) - r(A_0))\right]\right\}.$$

Furthermore, note that for n large enough,

$$\begin{aligned}
2n\lambda_n(|A| - d_0) - (r(A) - r(A_0)) &\ge (2n\lambda_n - N_n - p - 1)(|A| - d_0) \\
&\ge n\lambda_n(|A| - d_0) \ge -n\lambda_n d_0 = o(\log(N_n d) N_n d)
\end{aligned}$$

by assumption (C8). Therefore $I_{n1}^{(1)}$ is the dominated term inside of the exponential. Thus, when n is large enough, one has,

$$P\left\{L_n(\gamma) - L_n\left(\widetilde{\gamma}^{(o)}\right) < 0\right\} \le \exp\left\{-\frac{I_{n1}^{(1)}}{8}\right\} \le \exp\left\{-\frac{\Delta_n(A)}{8}\right\}. \qquad (24)$$

For any γ with $A_0 \not\subset A$ and $|A| > \alpha d_0$, we show that, $I_n = L_1(A) + L_2(A) + L_3(A)$, where $L_1(A) = -\frac{1}{a}(\mathbf{E} - (a-1)(\mathbf{I}_n - \mathbf{H}_n(A))\mathbf{M})^T(\mathbf{I}_n - \mathbf{H}_n(A))(\mathbf{E} - (a-1)(\mathbf{I}_n - \mathbf{H}_n(A))\mathbf{M})$,
$L_2(A) = (a-1)\mathbf{M}^T(\mathbf{I}_n - \mathbf{H}_n(A))\mathbf{M}$, and

$$L_3(A) = -\mathbf{M}^T(\mathbf{I}_n - \mathbf{H}_n(A_0))\mathbf{M} - 2\mathbf{E}^T(\mathbf{I}_n - \mathbf{H}_n(A_0))\mathbf{M}.$$

Here, $-aL_1(A)/\sigma^2$ follows a noncentral χ^2 distribution with the degree of freedom $n - \min(r(A), n)$ and noncentral parameter $(a-1)\mathbf{M}^T(\mathbf{I}_n - \mathbf{H}_n(A))\mathbf{M}/\sigma^2$. Furthermore, as in Case 1, one can show

that $L_3(A) = o_p(\log(dN_n)N_n d_0)$. Therefore $L_2(A)$ is the dominated term in I_n, by noting that $a \to \infty$ by assumption (C8). Thus, for n sufficiently large,

$$\begin{aligned} & P\left\{L_n(\gamma) - L_n\left(\widetilde{\gamma}^{(o)}\right) < 0\right\} \\ \leq & \exp\left\{-\frac{1+c}{4}\left[I_n + 2n\lambda_n(|A| - d_0) - (r(A) - r(A_0))\right]\right\} \\ \leq & \exp\left\{-\frac{1+c}{4}\left[2n\lambda_n(|A| - d_0) - (r(A) - r(A_0))\right]\right\}. \end{aligned} \tag{25}$$

Therefore, (24) and (25) give that,

$$\begin{aligned} & \sum_{A(\gamma)\in\Gamma_2} P\left(\min_{\gamma} L_n(\gamma) - L_n\left(\widetilde{\gamma}^{(o)}\right) \leq 0\right) \\ \leq & \sum_{i=1}^{[\alpha d_0]}\sum_{j=0}^{d_0-1}\left(\begin{array}{c} d_0 \\ j \end{array}\right)\left(\begin{array}{c} d-d_0 \\ i-j \end{array}\right)\exp\left\{-\frac{\min\Delta_n(A)}{8}\right\} \\ & + \sum_{i=[\alpha d_0]+1}^{d}\sum_{j=0}^{d_0-1}\left(\begin{array}{c} d_0 \\ j \end{array}\right)\left(\begin{array}{c} d-d_0 \\ i-j \end{array}\right)\exp\left\{-\frac{1+c}{4}\left[2n\lambda_n(i-d_0) - (r(A) - r(A_0))\right]\right\} \\ = & II_1 + II_2, \end{aligned}$$

where, by noting that $\binom{a}{b} \leq a^b$ for any two integers $a, b > 0$,

$$\begin{aligned} II_1 \leq & \sum_{i=1}^{[\alpha d_0]}\sum_{j=0}^{d_0-1} d_0^j (d-d_0)^{i-j}\exp\left(-\frac{\min\Delta_n(A)}{8}\right) \\ \leq & (N_n d)^{-N_n d/8} d_0^{[\alpha d_0]}(d-d_0)^{[\alpha d_0]}[\alpha d_0] d_0 \to 0, \end{aligned}$$

as $n \to \infty$, since d_0 is fixed and $N_n \to \infty$. Furthermore,

$$\begin{aligned} II_2 \leq & \sum_{i=[\alpha d_0]+1}^{d}\sum_{j=0}^{d_0-1}\left(\begin{array}{c} d_0 \\ j \end{array}\right)\left(\begin{array}{c} d-d_0 \\ i-j \end{array}\right)\exp\left\{-\frac{1+c}{4}\left[2n\lambda_n(i-d_0) - (r(A) - r(A_0))\right]\right\} \\ \leq & \sum_{i=[\alpha d_0]+1}^{d}\sum_{j=0}^{d_0-1} d_0^j (d-d_0)^{i-j}\exp\left\{-\frac{n\lambda_n(i-d_0)}{4}\right\} \\ \leq & \sum_{i=[\alpha d_0]+1}^{d} d_0\exp\left\{-\frac{n\lambda_n(i-d_0)}{4} + i\log(d)\right\} \to 0, \end{aligned}$$

as $n \to \infty$, by assumption (C8). Therefore, as $n \to \infty$,

$$\sum_{A\in\Gamma_2} P\left(\min_{\gamma:\mathbf{A}(\gamma)=A} L_n(\gamma) - L_n\left(\widetilde{\gamma}^{(o)}\right) \leq 0\right) \to 0. \tag{26}$$

Note that for the global minima $\widehat{\gamma}$ of (4), one has

$$P\left(\widehat{\gamma} \neq \widetilde{\gamma}^{(o)}\right) \leq \sum_{j=1}^{2}\sum_{A\in\Gamma_j} P\left(\min_{\gamma:\mathbf{A}(\gamma)=A} L_n(\gamma) - L_n\left(\widetilde{\gamma}^{(o)}\right) \leq 0\right).$$

Therefore, Theorem 2 follows from (23) and (26). ■

Appendix F. Proof of Theorem 3

Theorem 3 follows immediately from Lemma 6 and Theorem 2. ■

References

L. An and P. Tao. Solving a class of linearly constrained indefinite quadratic problems by D.C. algorithms. *Journal of Global Optimization*, 11:253-285, 1997.

L. Breiman and A. Cutler. A deterministic algorithm for global optimization. *Mathematical Programming*, 58:179-199, 1993.

C. de Boor. *A Practical Guide to Splines*. Springer, New York, 2001.

J. Fan and T. Huang. Profile likelihood inferences on semiparametric varying-coefficient partially linear models. *Bernoulli*, 11:1031-1057, 2005.

J. Fan and R. Li. Variable selection via nonconcave penalized likelihood and its oracle properties. *Journal of the American Statistical Association*, 96:1348-1360, 2001.

J. Fan and H. Peng. Nonconcave penalized likelihood with a diverging number of parameters. *Annals of Statistics*, 32:928-961, 2004.

J. Fan and J. Zhang. Two-step estimation of functional linear models with applications to longitudinal data. *Journal of the Royal Statistical Society, Series B*, 62:303-322, 2000.

T. Hastie and R. Tibshirani. Varying-coefficient models. *Journal of the Royal Statistical Society, Series B*, 55:757-796, 1993.

D. R. Hunter and R. Li. Variable selection using MM algorithms. *Annals of Statistics*, 33:1617-1642, 2005.

D. R. Hoover, J. A. Rice, C. O. Wu, and L. Yang. Nonparametric smoothing estimates of time-varying coefficient models with longitudinal data. *Biometrika*, 85:809-822, 1998.

J. Huang, J. L. Horowitz, and F. Wei. Variable selection in nonparametric additive models. *Annals of Statistics*, 38:2282-2313, 2010.

J. Z. Huang, C. O. Wu, and L. Zhou. Varying-coefficient models and basis function approximations for the analysis of repeated measurements. *Biometrika*, 89:111-128, 2002.

J. Z. Huang, C. O. Wu, and L. Zhou. Polynomial spline estimation and inference for varying coefficient models with longitudinal data. *Statistica Sinica*, 14:763-788, 2004.

Y. Kim, H. Choi, and H. Oh. Smoothly clipped absolute deviation on high dimensions. *Journal of the American Statistical Association*, 103:1665-1673, 2008.

Y. Liu, X. Shen, and W. Wong. Computational development of *psi*-learning. *Proc SIAM 2005 Int. Data Mining Conf.*, 1-12, 2005.

A. Qu, and R. Li. Quadratic inference functions for varying-coefficient models with longitudinal data. *Biometrics*, 62:379-391, 2006.

J. O. Ramsay, and B. W. Silverman. *Functional Data Analysis.* Springer-Verlag: New York, 1997.

X. Shen, W. Pan, Y. Zhu. Likelihood-based selection and sharp parameter estimation. *Journal of the American Statistical Association*, 107:223-232, 2012.

X. Shen, G. C. Tseng, X. Zhang, and W. H. Wong. On ψ-learning. *Journal of the American Statistical Association*, 98:724-734, 2003.

R. Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society, Series B*, 58:267-288, 1996.

S. Van de Geer. High-dimensional generalized linear models and the Lasso. *Annals of Statistics*, 36:614-645, 2008.

H. Wang, and Y. Xia. Shrinkage estimation of the varying coefficient model. *Journal of the American Statistical Association*, 104:747-757, 2009.

L. Wang, H. Li, and J. Z. Huang. Variable selection in nonparametric varying-coefficient models for analysis of repeated measurements. *Journal of the American Statistical Association*, 103:1556-1569, 2008.

F. Wei, J. Huang, and H. Li. Variable selection and estimation in high-dimensional varying coefficient models. *Statistica Sinica*, 21:1515-1540, 2011.

C. O. Wu, and C. Chiang. Kernel smoothing on varying coefficient models with longitudinal dependent variable. *Statistica Sinica*, 10:433-456, 2000.

L. Xue, A. Qu, and J. Zhou. Consistent model selection for marginal generalized additive model for correlated data. *Journal of the American Statistical Association*, 105:1518-1530, 2010.

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.

C. H. Zhang. Nearly unbiased variable selection under minimax concave penalty. *Annals of Statistics*, 38:894-942, 2010.

P. Zhao, and B. Yu. On model selection consistency of Lasso. *Journal of Machine Learning Research*, 7:2541-2563, 2006.

S. Zhou, X. Shen, and D. A. Wolfe. Local asymptotics for regression splines and confidence regions. *Annals of Statistics*, 26:1760-1782, 1998.

H. Zou, and R. Li. One-step sparse estimates in nonconcave penalized likelihood models. *Annals of Statistics*, 36:1509-1533, 2008.

Journal of Machine Learning Research 13 (2012) 1999-2030 Submitted 5/11; Revised 1/12; Published 6/12

An Improved GLMNET for L1-regularized Logistic Regression

Guo-Xun Yuan R96042@CSIE.NTU.EDU.TW
Chia-Hua Ho B95082@CSIE.NTU.EDU.TW
Chih-Jen Lin CJLIN@CSIE.NTU.EDU.TW
Department of Computer Science
National Taiwan University
Taipei 106, Taiwan

Editor: S. Sathiya Keerthi

Abstract

Recently, Yuan et al. (2010) conducted a comprehensive comparison on software for L1-regularized classification. They concluded that a carefully designed coordinate descent implementation CDN is the fastest among state-of-the-art solvers. In this paper, we point out that CDN is less competitive on loss functions that are expensive to compute. In particular, CDN for logistic regression is much slower than CDN for SVM because the logistic loss involves expensive exp/log operations.

In optimization, Newton methods are known to have fewer iterations although each iteration costs more. Because solving the Newton sub-problem is independent of the loss calculation, this type of methods may surpass CDN under some circumstances. In L1-regularized classification, GLMNET by Friedman et al. is already a Newton-type method, but experiments in Yuan et al. (2010) indicated that the existing GLMNET implementation may face difficulties for some large-scale problems. In this paper, we propose an improved GLMNET to address some theoretical and implementation issues. In particular, as a Newton-type method, GLMNET achieves fast local convergence, but may fail to quickly obtain a useful solution. By a careful design to adjust the effort for each iteration, our method is efficient for both loosely or strictly solving the optimization problem. Experiments demonstrate that our improved GLMNET is more efficient than CDN for L1-regularized logistic regression.

Keywords: L1 regularization, linear classification, optimization methods, logistic regression, support vector machines

1. Introduction

Logistic regression and support vector machines (SVM) are popular classification methods in machine learning. Recently, L1-regularized logistic regression and SVM are widely used because they can generate a sparse model. Given a set of instance-label pairs $(\mathbf{x}_i, y_i)$, $i = 1, \ldots, l$, $\mathbf{x}_i \in \mathbf{R}^n$, $y_i \in \{-1, +1\}$, an L1-regularized classifier solves the following unconstrained optimization problem:

$$\min_{\mathbf{w}} \quad f(\mathbf{w}), \tag{1}$$

where

$$f(\mathbf{w}) \equiv \|\mathbf{w}\|_1 + L(\mathbf{w}),$$

$\|\cdot\|_1$ denotes the 1-norm, and $L(\mathbf{w})$ indicates training losses

$$L(\mathbf{w}) \equiv C\sum_{i=1}^{l}\xi(\mathbf{w};\mathbf{x}_i,y_i). \tag{2}$$

For logistic regression and L2-loss SVM, we have the following loss functions.

$$\begin{aligned}\xi_{\log}(\mathbf{w};\mathbf{x},y) &= \log(1+e^{-y\mathbf{w}^T\mathbf{x}}) \quad \text{and}\\ \xi_{\text{svm}}(\mathbf{w};\mathbf{x},y) &= \max(0,1-y\mathbf{w}^T\mathbf{x})^2.\end{aligned} \tag{3}$$

The regularization term $\|\mathbf{w}\|_1$ is used to avoid overfitting the training data. The user-defined parameter $C>0$ is used to balance regularization and loss terms. Different from the 2-norm regularization, the 1-norm regularization gives a sparse solution of (1).

It is difficult to solve (1) because $\|\mathbf{w}\|_1$ is not differentiable. Many optimization approaches have been proposed and an earlier comparison is by Schmidt et al. (2009). Recently, Yuan et al. (2010) made a comprehensive comparison among state-of-the-art algorithms and software for L1-regularized logistic regression and SVM. For L1-regularized logistic regression, they compared CDN (Yuan et al., 2010), BBR (Genkin et al., 2007), SCD (Shalev-Shwartz and Tewari, 2009), CGD (Tseng and Yun, 2009), IPM (Koh et al., 2007), BMRM (Teo et al., 2010), OWL-QN (Andrew and Gao, 2007). Lassplore (Liu et al., 2009), TRON (Lin and Moré, 1999), and GLMNET (Friedman et al., 2010). Other existing approaches include, for example, Shevade and Keerthi (2003), Lee et al. (2006) and Shi et al. (2010). Yuan et al. (2010) conclude that carefully designed coordinate descent (CD) methods perform better than others for large sparse data (e.g., document data). As a result, their CD method (called CDN) was included in a popular package LIBLINEAR as the solver of L1-regularized logistic regression.

However, we point out in Section 2 that CDN becomes inefficient if the loss function is expensive to compute. An example is L1-regularized logistic regression, where exp/log operations are more expensive than other basic operations. We investigate this problem in detail to show that CDN suffers from frequent loss-function computation.

In Section 3, we show that for expensive loss functions, Newton-type methods are more suitable. A Newton method needs not compute the loss function when finding the Newton direction, which is the most time consuming part. Based on this point, we attempt to obtain an appropriate Newton-type method for L1-regularized logistic regression. We introduce an existing Newton-type algorithm GLMNET (Friedman et al., 2010). In Yuan et al.'s comparison, GLMNET, although inferior to CDN, performs reasonably well. However, GLMNET failed to train some large-scale data used in their experiments. In Sections 4 and 5, we improve GLMNET in theoretical and practical aspects, respectively. We call the improved method newGLMNET. Based on the modification in Section 4, we establish the asymptotic convergence of newGLMNET. By a careful design in Section 5 to adjust the effort for each iteration, newGLMNET is efficient for both loosely and strictly solving the optimization problem. Note that our discussion in Sections 2–4 is generic to all differentiable loss functions, although the focus is on logistic regression.

Experiments in Section 6 show that newGLMNET is more efficient than CDN, which was considered the state of the art for L1-regularized logistic regression. In particular, newGLMNET is much faster for dense problems. While logistic regression is an example of problems with expensive loss functions, to check the situation of cheap loss functions, in Section 7, we extend newGLMNET to solve L2-loss SVM. Experiments show that, contrary to logistic regression, CDN is slightly

better. Therefore, our investigation in this work fully demonstrate that expensive loss functions need a different design of training algorithms from that of cheap loss functions. Section 8 concludes this work. A supplementary file including additional analysis and experiments is available at `http://www.csie.ntu.edu.tw/~cjlin/papers/l1_glmnet/supplement.pdf`.

Because the proposed newGLMNET is faster for logistic regression, we replace the CDN solver in the package LIBLINEAR with newGLMNET after version 1.8. This paper is an extension of an earlier conference paper (Yuan et al., 2011). In addition to a thorough reorganization of the main results, more analysis and theoretical results are included.

2. Coordinate Descent (CD) Method and Its Weakness

CD is a commonly-used optimization approach by iteratively solving one-variable sub-problems. For L1-regularized classification, past works (e.g., Genkin et al., 2007; Yuan et al., 2010) have shown that CD methods can quickly obtain a useful model. In this section, we first discuss a specific CD method called CDN (Yuan et al., 2010) and follow by showing its weakness.

2.1 CDN

At the kth iteration, a CD method cyclically selects a dimension $j \in \{1,2,\ldots,n\}$ and solves the following one-variable sub-problem.

$$\min_d \quad f(\mathbf{w}^{k,j}+d\mathbf{e}_j)-f(\mathbf{w}^{k,j}), \tag{4}$$

where

$$f(\mathbf{w}^{k,j}+d\mathbf{e}_j)-f(\mathbf{w}^{k,j}) = \|\mathbf{w}^{k,j}+d\mathbf{e}_j\|_1 - \|\mathbf{w}^{k,j}\|_1 + L(\mathbf{w}^{k,j}+d\mathbf{e}_j)-L(\mathbf{w}^{k,j}).$$

In (4), we define

$$\mathbf{w}^{k,j} \equiv [w_1^{k+1},\ldots,w_{j-1}^{k+1},w_j^k,\ldots,w_n^k]^T \tag{5}$$

and the indicator vector

$$\mathbf{e}_j \equiv [\underbrace{0,\ldots,0}_{j-1},1,0,\ldots,0]^T.$$

Let $\mathbf{w}^k = \mathbf{w}^{k,1} = \mathbf{w}^{k-1,n+1}$ at the beginning of each iteration. If d is an optimal solution of (4), then $\mathbf{w}^{k,j}$ is updated to $\mathbf{w}^{k,j+1}$ by

$$w_t^{k,j+1} = \begin{cases} w_t^{k,j}+d & \text{if } t=j, \\ w_t^{k,j} & \text{otherwise.} \end{cases}$$

For logistic regression, the one-variable sub-problem (4) does not have a closed-form solution, so Yuan et al. (2010) approximately solve it using the second-order approximation of $L(\mathbf{w}^{k,j}-d\mathbf{e}_j)-L(\mathbf{w}^{k,j})$.

$$\min_d \quad \nabla_j L(\mathbf{w}^{k,j})d+\frac{1}{2}\nabla^2_{jj}L(\mathbf{w}^{k,j})d^2+|w_j^k+d|-|w_j^k|. \tag{6}$$

For logistic regression,

$$\nabla L(\mathbf{w}) = C\sum_{i=1}^{l}(\tau(y_i\mathbf{w}^T\mathbf{x}_i)-1)y_i\mathbf{x}_i \quad \text{and} \quad \nabla^2 L(\mathbf{w}) = CX^TDX, \tag{7}$$

Algorithm 1 CDN framework in Yuan et al. (2010). Some implementation details are omitted.

1. Given $\mathbf{w}^1$.
2. For $k = 1,2,3,\ldots$ // iterations
   - For $j = 1,2,3,\ldots,n$ // n CD steps
     - Compute the optimum d of sub-problem (6) by (9).
     - Find the step size $\bar{\lambda}$ by (10).
     - $\mathbf{w}^{k,j+1} \leftarrow \mathbf{w}^{k,j} + \bar{\lambda} d \mathbf{e}_j$.

where $\tau(s)$ is the derivative of the logistic loss function $\log(1+e^s)$:

$$\tau(s) = \frac{1}{1+e^{-s}},$$

$D \in \mathbf{R}^{l \times l}$ is a diagonal matrix with

$$D_{ii} = \tau(y_i \mathbf{w}^T \mathbf{x}_i)\left(1 - \tau(y_i \mathbf{w}^T \mathbf{x}_i)\right), \tag{8}$$

and

$$X \equiv \begin{bmatrix} \mathbf{x}_1^T \\ \vdots \\ \mathbf{x}_l^T \end{bmatrix} \in \mathbf{R}^{l \times n}.$$

It is well known that (6) has a simple closed-form solution

$$d = \begin{cases} -\frac{\nabla_j L(\mathbf{w}^{k,j})+1}{\nabla^2_{jj} L(\mathbf{w}^{k,j})} & \text{if } \nabla_j L(\mathbf{w}^{k,j}) + 1 \le \nabla^2_{jj} L(\mathbf{w}^{k,j}) w_j^{k,j}, \\ -\frac{\nabla_j L(\mathbf{w}^{k,j})-1}{\nabla^2_{jj} L(\mathbf{w}^{k,j})} & \text{if } \nabla_j L(\mathbf{w}^{k,j}) - 1 \ge \nabla^2_{jj} L(\mathbf{w}^{k,j}) w_j^{k,j}, \\ -w_j^{k,j} & \text{otherwise.} \end{cases} \tag{9}$$

Because (9) considers a Newton direction, Yuan et al. (2010) refer to this setting as CDN (CD method using one-dimensional Newton directions). For convergence, Yuan et al. follow Tseng and Yun (2009) to apply a line search procedure. The largest step size $\lambda \in \{\beta^i \mid i = 0,1,\ldots\}$ is found such that λd satisfies the following sufficient decrease condition.

$$f(\mathbf{w}^{k,j} + \lambda d \mathbf{e}_j) - f(\mathbf{w}^{k,j}) \le \sigma\lambda\left(\nabla_j L(\mathbf{w}^{k,j}) d + |w_j^k + d| - |w_j^k|\right), \tag{10}$$

where $0 < \beta < 1$ and $0 < \sigma < 1$ are pre-specified parameters.

The basic structure of CDN is in Algorithm 1. To make CDN more efficient, Yuan et al. have considered some implementation tricks, but details are omitted here.

We discuss the computational complexity of CDN. While solving (6) by (9) takes a constant number of operations, calculating $\nabla_j L(\mathbf{w}^{k,j})$ and $\nabla^2_{jj} L(\mathbf{w}^{k,j})$ for constructing the sub-problem (6) is expensive. From (7), we need $O(nl)$ operations for obtaining $\mathbf{w}^T \mathbf{x}_i, \forall i$. A common trick to make CD methods viable for classification problems is to store and maintain $\mathbf{w}^T \mathbf{x}_i, \forall i$. Yuan et al. (2010) store $e^{\mathbf{w}^T \mathbf{x}_i}$ instead and update the values by

$$e^{\mathbf{w}^T \mathbf{x}_i} \leftarrow e^{\mathbf{w}^T \mathbf{x}_i} \cdot e^{\bar{\lambda} d x_{ij}}, \forall i, \tag{11}$$

Data set	exp/log	Total
epsilon	64.25 (73.0%)	88.18
webspam	72.89 (66.6%)	109.39

Table 1: Timing analysis of the first CD cycle of CDN. Time is in seconds.

where $\bar{\lambda}$ is the step size decided by the line search procedure and d is the optimal solution of (6). If $e^{\mathbf{w}^T\mathbf{x}_i}, \forall i$ are available, the evaluation of $\nabla_j L(\mathbf{w}^{k,j})$ and $\nabla^2_{jj} L(\mathbf{w}^{k,j})$ in (6) and $f(\mathbf{w}^{k,j} + \lambda d\mathbf{e}_j)$ in the sufficient decrease condition (10) takes $O(l)$ operations. Therefore, with n CD steps in one iteration, the complexity of each iteration is:

$$n \cdot (1+ \text{\# steps of line search}) \cdot O(l). \tag{12}$$

For sparse data, in (11), only $e^{\mathbf{w}^T\mathbf{x}_i}$ with $x_{ij} \neq 0$ needs to be updated. Then, $n \cdot O(l)$ in Equation (12) can be reduced to $O(\text{nnz})$, where nnz is the total number of non-zero elements in X (i.e., training data). In Algorithm 1, one CD iteration contains n CD steps to update $w_1, \ldots, w_n$ as a cycle. This concept of CD cycles will be frequently used in our subsequent analysis and experiments.

2.2 Weakness of CDN

Although CDN is reported as the best method in the comparison by Yuan et al. (2010), for the same data set, CDN's training time for logistic regression is more than L2-loss SVM. Motivated from this observation, we point out that CDN suffers from expensive exp/log operations of logistic regression.

In Table 1, we conduct an experiment on two data sets, epsilon and webspam.[1] We check the proportion of time for exp/log operations in the first CD cycle of CDN. The results clearly show that exp/log operations dominate the training time. We present results of more data sets in Section 6 and have similar observations.

Exp/log operations occur in two places (11) and (10), each of which costs $O(l)$. From (12), we can see that the complexity of exp/log operations is the same as that of all operations.[2] Because each exp/log is much more expensive than a basic operation like multiplication, a significant portion of running time is spent on exp/log operations.

3. GLMNET: A Method that Less Frequently Computes the Loss Function

Based on the observation in Section 2, to reduce the number of exp/log operations, we should consider methods which less frequently compute the loss function. In this section, we identify such methods and present one of them named GLMNET.

3.1 Algorithms that may Have Less Loss Computation

For logistic regression, exp/log operations occur in computing the function, gradient, and Hessian of the loss. To avoid frequent exp/log operations, we hope an optimization method could conduct enough basic (e.g., multiplication or division) operations between two function, gradient, or Hessian

1. Details of the data sets are in Section 6.1.
2. For binary-valued data, only $e^{\bar{\lambda} d x_{ij}}$ becomes $e^{\bar{\lambda} d}$ in (11), so $O(nl)$ exp/log operations can be reduced to $O(n)$. This has been pointed out in Huang et al. (2010).

evaluations. However, these basic operations should also be useful for minimizing the optimization problem. We find that methods involving second-order approximation of $f(\mathbf{w})$ may fulfill the requirements. At the kth iteration, consider the following sub-problem to find a direction $\mathbf{d}$:

$$\min_{\mathbf{d}} \quad q_k(\mathbf{d}), \tag{13}$$

where

$$q_k(\mathbf{d}) \equiv \nabla L(\mathbf{w}^k)^T\mathbf{d} + \frac{1}{2}\mathbf{d}^T H^k \mathbf{d} + \|\mathbf{w}^k + \mathbf{d}\|_1 - \|\mathbf{w}^k\|_1,$$

and H^k is either $\nabla^2 L(\mathbf{w}^k)$ or its approximation. Then, $\mathbf{w}$ is updated by

$$\mathbf{w}^{k+1} \leftarrow \mathbf{w}^k + \mathbf{d}. \tag{14}$$

Between two iterations, H^k and $\nabla f(\mathbf{w}^k)$ are constants, so (13) is a quadratic program without involving exp/log operations. Further, (13) is not a trivial sub-problem to solve.

If $H^k = \nabla^2 L(\mathbf{w}^k)$, we have a Newton-type method that usually enjoys a small number of iterations. At each iteration, obtaining $\nabla f(\mathbf{w}^k)$ and $\nabla^2 f(\mathbf{w}^k)$ via (8) requires at least $O(nl)$ operations because of calculating $\sum_{i=1}^{l}(\tau(y_i\mathbf{w}^T\mathbf{x}) - 1)y_i\mathbf{x}_i$. However, the number of exp/log operations is only $O(l)$. With the cost of solving the sub-problem (13), the total cost of one iteration is at least $O(nl)$, but only a small portion, $O(l)$, is for exp/log computation. This situation is much better than CDN, which requires $O(nl)$ exp/log operations in $O(nl)$ overall operations. An existing Newton-type method for L1-regularized classification is GLMNET by Friedman et al. (2010). We will discuss its details in Section 3.2.[3]

Note that CDN also applies second-order approximation for solving the one-variable sub-problem (4). However, once a variable is changed in CDN, the gradient and Hessian become different. In contrast, for GLMNET, gradient and Hessian remain the same while (13) is being solved. The reason is that GLMNET applies second-order approximation on the whole objective function $f(\mathbf{w})$. Such differences explain why GLMNET needs less exp/log computation than CDN.

If we use an approximate Hessian as H^k, the analysis of $O(l)$ exp/log operations versus at least $O(nl)$ total operations per iteration still holds.[4] However, because minimizing $q_k(\mathbf{d})$ in (13) becomes easier, exp/log operations may play a more important role in the whole procedure. In the extreme situation, H^k is a constant diagonal matrix, so we have a gradient descent method. Existing approaches of using such H^k include ISTA (Daubechies et al., 2004), FISTA (Beck and Teboulle, 2009), and others.

For L2-regularized logistic regression, Chang et al. (2008) have pointed out that Newton methods less frequently conduct exp/log operations than CD methods, but they did not conduct detailed analysis and comparisons.

3.2 GLMNET

We pointed out in Section 3.1 that GLMNET is an existing Newton-type method for L1-regularized classification. At each iteration, it solves the sub-problem (13) with $H^k = \nabla^2 L(\mathbf{w}^k)$ and updates

3. The GLMNET code by Friedman et al. (2010) supports using an approximate Hessian matrix, but here we consider only the case of using the exact Hessian matrix.
4. We assume that obtaining an approximation of $\nabla^2 f(\mathbf{w}^k)$ requires no more than $O(l)$ exp/log operations.

Algorithm 2 Basic framework of GLMNET (Friedman et al., 2010) for L1-regularized logistic regression.

1. Given $\mathbf{w}^1$.
2. For $k = 1, 2, 3, \ldots$
   - Compute $\nabla L(\mathbf{w}^k)$ and $\nabla^2 L(\mathbf{w}^k)$ by (7).
   - Obtain $\mathbf{d}^k$ by solving sub-problem (13) by CD with certain stopping condition.
   - $\mathbf{w}^{k+1} = \mathbf{w}^k + \mathbf{d}^k$.

$\mathbf{w}$ by (14). Although many optimization methods can be applied to solve the sub-problem (13), Friedman et al. (2010) consider a cyclic coordinate descent method similar to CDN in Section 2.1. Indeed, it is simpler than CDN because (13) is only a quadratic problem. We use $\mathbf{d}^p$ to denote the CD iterates (cycles) for solving (13). Each CD cycle is now considered as an inner iteration of GLMNET. Sequentially, $\mathbf{d}^p$'s values are updated by minimizing the following one-variable function.

$$\begin{aligned} & q_k(\mathbf{d}^{p,j} + z\mathbf{e}_j) - q_k(\mathbf{d}^{p,j}) \\ &= |w_j^k + d_j^p + z| - |w_j^k + d_j^p| + \nabla_j \bar{q}_k(\mathbf{d}^{p,j}) z + \frac{1}{2} \nabla^2_{jj} \bar{q}_k(\mathbf{d}^{p,j}) z^2, \end{aligned} \tag{15}$$

where the definition of $\mathbf{d}^{p,j}$ is similar to $\mathbf{w}^{k,j}$ of CDN in (5)

$$\mathbf{d}^{p,j} \equiv [d_1^{p-1}, d_2^{p-1}, \ldots, d_{j-1}^{p-1}, d_j^p, \ldots, d_n^p]^T,$$

and $\mathbf{d}^p = \mathbf{d}^{p,1} = \mathbf{d}^{p-1,n+1}$. Further,

$$\bar{q}_k(\mathbf{d}) \equiv \nabla L(\mathbf{w}^k)^T \mathbf{d} + \frac{1}{2} (\mathbf{d})^T \nabla^2 L(\mathbf{w}^k) \mathbf{d}$$

represents the smooth terms of $q_k(\mathbf{d})$ and plays a similar role to $L(\mathbf{w})$ for (1). We have

$$\begin{aligned} & \nabla_j \bar{q}_k(\mathbf{d}^{p,j}) = \nabla_j L(\mathbf{w}^k) + (\nabla^2 L(\mathbf{w}^k) \mathbf{d}^{p,j})_j \quad \text{and} \\ & \nabla^2_{jj} \bar{q}_k(\mathbf{d}^{p,j}) = \nabla^2_{jj} L(\mathbf{w}^k). \end{aligned} \tag{16}$$

Equation (15) is in the same form as (6), so it can be easily solved by (9). Because the one-variable function is exactly minimized, line search is not required in the CD procedure. The basic structure of GLMNET is in Algorithm 2.

Because an iterative procedure (CD method) is used to solve the sub-problem (13), GLMNET contains two levels of iterations. A suitable stopping condition for the inner level is very important. In Section 5.2, we will discuss GLMNET's stopping conditions and make some improvements.

We analyze GLMNET's complexity to confirm that it less frequently conducts exp/log operations. At each CD step, most operations are spent on calculating $\nabla_j \bar{q}_k(\mathbf{d}^{p,j})$ and $\nabla^2_{jj} \bar{q}_k(\mathbf{d}^{p,j})$ in (16). Note that $\nabla^2_{jj} \bar{q}_k(\mathbf{d}^{p,j}) = \nabla^2_{jj} L(\mathbf{w}^k), \forall j$ can be pre-calculated before the CD procedure. For $\nabla_j \bar{q}_k(\mathbf{d}^{p,j})$, the first term $\nabla_j L(\mathbf{w}^k)$ can also be pre-calculated. With (7), the second term is

$$(\nabla^2 L(\mathbf{w}^k) \mathbf{d}^{p,j})_j = C \sum_{t=1}^{n} \sum_{i=1}^{l} X_{ji}^T D_{ii} X_{it} d_t^{p,j} = C \sum_{i=1}^{l} X_{ji}^T D_{ii} (X \mathbf{d}^{p,j})_i.$$

One cycle of n CD steps		CDN	GLMNET
Total # of operations	dense data	$O(nl)$	$O(nl)$
	sparse data	$O(\text{nnz})$	$O(\text{nnz})$
# exp/log operations	dense data	$O(nl)$	$\leq O(l)$
	sparse data	$O(\text{nnz})$	$\leq O(l)$

Table 2: A comparison between CDN and GLMNET on the number of exp/log operations in one CD cycle. We assume that in (12), the number of line search steps of CDN is small (e.g., one or two). Note that in GLMNET, exp/log operations are needed in the beginning/end of an outer iteration. That is, they are conducted once every several CD cycles. We make each CD cycle to share the cost in this table even though a CD cycle in GLMNET involves no exp/log operations.

If $X\mathbf{d}^{p,j}$ (i.e., $\mathbf{x}_i^T\mathbf{d}^{p,j}, \forall i$) is maintained and updated by

$$(X\mathbf{d}^{p,j+1})_i \leftarrow (X\mathbf{d}^{p,j})_i + X_{ij}z, \ \forall i, \tag{17}$$

then calculating $\nabla_j \bar{q}_k(\mathbf{d})$ costs $O(l)$ operations.[5] Therefore, the CD method for (13) requires

$$O(nl) \text{ operations for one inner iteration (cycle) of } n \text{ CD steps.} \tag{18}$$

The complexity of GLMNET is thus

$$\#\text{outer\_iters} \times (O(nl) + \#\text{inner\_iters} \times O(nl)),$$

where the first $O(nl)$ is for obtaining items such as $\nabla f(\mathbf{w}^k)$ before solving (13). We then compare the number of exp/log operations in CDN and GLMNET. Because they both use CD, we check in Table 2 that relative to the total number of operations of one CD cycle, how many exp/log operations are needed. Clearly, CDN's $O(nl)$ exp/log operations are much more than GLMNET's $O(l)$. The difference becomes smaller for sparse data because CDN's $O(nl)$ is reduced by $O(\text{nnz})$. From this analysis, we expect that CDN suffers from many slow exp/log operations if data are dense and n is not small. This result will be clearly observed in Section 6.

Although our analysis indicates that GLMNET is superior to CDN in terms of the number of exp/log operations, experiments in Yuan et al. (2010) show that overall GLMNET is slower. The final local convergence of GLMNET is fast, but it often spends too much time in early iterations. Therefore, contrary to CDN, GLMNET does not quickly obtain a useful model. Further, GLMNET failed to solve two large problems in the experiment of Yuan et al. (2010) and its theoretical convergence is not guaranteed. In the next two sections, we will propose an improved GLMNET to perform faster than CDN for logistic regression.

4. newGLMNET: An Improved GLMNET

As mentioned in Section 3.2, GLMNET lacks theoretical convergence properties. In this section, we modify GLMNET to be a special case of a class of methods by Tseng and Yun (2009), so the asymptotic convergence immediately follows. We refer to the improved algorithm as newGLMNET.

5. This is like how $e^{\mathbf{w}^T\mathbf{x}_i}, \ \forall i$ are handled in Section 2.1.

The framework by Tseng and Yun (2009) for L1-regularized problems is a generalized coordinate descent method. At each iteration, it selects some variables for update based on certain criteria. An extreme situation is to update one variable at a time, so CDN in Section 2 is a special case. The other extreme is to update all variables at an iteration. In this situation, Tseng and Yun's method considers a quadratic approximation the same as (13). However, for the convergence, they required H^k in (13) to be positive definite. From (7), if X's columns are independent, then $\nabla^2 L(\mathbf{w}^k)$ is positive definite. To handle the situation that $\nabla^2 L(\mathbf{w}^k)$ is only positive semi-definite, we slightly enlarge the diagonal elements by defining

$$H^k \equiv \nabla^2 L(\mathbf{w}^k) + \nu I, \tag{19}$$

where $\nu > 0$ is a small value and $I \in \mathbf{R}^{n\times n}$ is an identity matrix.

In addition, for convergence Tseng and Yun (2009) require that line search is conducted. After obtaining an optimal solution $\mathbf{d}$ of (13), the largest step size $\lambda \in \{\beta^i \mid i = 0, 1, \dots\}$ is found such that $\lambda\mathbf{d}$ satisfies the following sufficient decrease condition.

$$\begin{aligned} & f(\mathbf{w}^k + \lambda\mathbf{d}) - f(\mathbf{w}^k) \\ & \le \sigma\lambda\big(\nabla L(\mathbf{w}^k)^T\mathbf{d} + \gamma\mathbf{d}^T H^k\mathbf{d} + \|\mathbf{w}^k + \mathbf{d}\|_1 - \|\mathbf{w}^k\|_1\big), \end{aligned} \tag{20}$$

where $0 < \beta < 1$, $0 < \sigma < 1$, and $0 \le \gamma < 1$. GLMNET does not conduct line search, so function values of its iterations may not be decreasing. In newGLMNET we use (20) with $\gamma = 0$.

If the sub-problem (13) is exactly solved, we prove that because all conditions needed in Tseng and Yun (2009, Theorems 1(e) and 3) are satisfied, line search is guaranteed to stop after a finite number of step sizes. Further, for asymptotic convergence, we have that any limit point of $\{\mathbf{w}^k\}$ generated by newGLMNET is an optimal solution of (1); see the proof in Appendix A. For local convergence rate, we prove in Appendix B that if the loss function $L(\mathbf{w})$ is strictly convex,[6] then the objective function value converges at least linearly.

If we know that $L(\mathbf{w})$ is strictly convex beforehand, we can directly use $H^k = \nabla^2 L(\mathbf{w}^k)$ without adding νI. The same explanation in Appendix A implies both the finite termination of line search and the asymptotic convergence. In this situation, we can obtain a better local quadratic convergence. See details in the supplementary document, in which we modify the proof in Hsieh et al. (2011) for L1-regularized Gaussian Markov random field.

4.1 Cost of Line Search

GLMNET does not conduct line search because of the concern on its cost. Interestingly, by the following analysis, we show that line search in newGLMNET introduces very little extra cost. For each step size λ tried in line search, we must calculate $f(\mathbf{w}^k + \lambda\mathbf{d})$. A direct implementation requires $O(nl)$ operations to obtain $(\mathbf{w}^k + \lambda\mathbf{d})^T\mathbf{x}_i, \forall i = 1, \dots, l$. If $e^{(\mathbf{w}^k)^T\mathbf{x}_i}, \forall i$ are available, we need only $O(l)$ by using $X\mathbf{d}$ maintained by (17).

$$e^{(\mathbf{w}^k+\lambda\mathbf{d})^T\mathbf{x}_i} = e^{(\mathbf{w}^k)^T\mathbf{x}_i} \cdot e^{\lambda(X\mathbf{d})_i}. \tag{21}$$

Thus, the cost for finding $f(\mathbf{w}^k + \lambda\mathbf{d})$ is reduced to $O(n + l)$, where $O(n)$ comes from calculating $\|\mathbf{w}^k + \lambda\mathbf{d}\|_1$. After the last λ is obtained in line search, we have $e^{(\mathbf{w}^{k+1})^T\mathbf{x}_i} = e^{(\mathbf{w}^k+\lambda\mathbf{d})^T\mathbf{x}_i}, \forall i$ for the next iteration. If only a small number of λ's are tried, then the $O(n + l)$ cost is negligible because the whole iteration costs at least $O(nl)$ from earlier discussion.

6. For situations such as $n > l$, $L(\mathbf{w}^k)$ is not strictly convex.

Discussion in Section 3 indicated that in every $O(nl)$ operations, GLMNET needs a much smaller amount of exp/log operations than CDN. We will show that a similar situation occurs for the line-search operations of newGLMNET and CDN. Note that line search is used in different places of the two methods. For CDN, at each CD step for updating one variable, line search is needed. In contrast, in newGLMNET, we conduct line search only in the end of an outer iteration. Following the discussion in Section 2.1 and this section, for each step size λ tried in line search of CDN and newGLMNET, the cost is $O(l)$ and $O(n+l)$, respectively. If we distribute the line search cost of newGLMNET to its inner CD cycles, we have that, for the same λ in one CD cycle,[7] CDN costs $O(nl)$ and newGLMNET costs no more than $O(n+l)$. This difference is similar to that of exp/log operations discussed in Section 3.

Because of CDN's high cost on line search, Yuan et al. (2010) develop the following trick. By deriving an upper-bound function $\delta(\lambda)$ such that

$$f(\mathbf{w}^{k,j}+\lambda d\mathbf{e}_j)-f(\mathbf{w}^{k,j})\le\delta(\lambda),\quad\forall\lambda\ge 0, \tag{22}$$

they check first if

$$\delta(\lambda)\le\sigma\lambda(\nabla_j L(\mathbf{w}^{k,j})d+|w_j^k+d|-|w_j^k|). \tag{23}$$

This trick, used for each step size λ, is particularly useful if the above inequality holds at $\lambda=1$. If $\delta(\lambda)$ can be calculated in $O(1)$, the $O(l)$ cost for line search at a CD step is significantly reduced to $O(1)$. More details about this upper-bound function can be found in Fan et al. (2008, Appendix G).

In Section 6.2, we will conduct experiments to investigate the line search cost of CDN and newGLMNET.

5. Implementation Issues of newGLMNET

Besides theoretical issues discussed in Section 4, in this section, we discuss some implementation techniques to make newGLMNET an efficient method in practice.

5.1 Random Permutation of One-variable Sub-problems

To solve the sub-problem (13), a conventional CD method sequentially updates variables $d_1, d_2, \ldots, d_n$. Many past works (e.g., Chang et al., 2008; Hsieh et al., 2008; Yuan et al., 2010) have experimentally indicated that using a random order leads to faster convergence. We adapt this strategy in the CD procedure of newGLMNET to solve sub-problem (13).

5.2 An Adaptive Inner Stopping Condition

GLMNET contains two levels of iterations. An "outer iteration" corresponds to the process from $\mathbf{w}^k$ to $\mathbf{w}^{k+1}$, while the "inner" level consists of CD iterations for solving (13). For an algorithm involving two levels of iterations, the stopping condition of the inner iterations must be carefully designed. A strict inner stopping condition may cause the algorithm to take a prohibitive amount of time at the first outer iteration. Alternatively, a loose inner condition leads to an inaccurate solution of (13) and possibly lengthy outer iterations. GLMNET terminates the CD procedure by checking if $\mathbf{d}$ is still significantly changed. That is, in the pth CD cycle to update $d_1^p, \ldots, d_n^p$, the corresponding

7. For simplicity, we assume that this λ is tried in all n CD steps of the cycle.

changes $z_1, \ldots, z_n$ satisfy

$$\max_j (\nabla^2_{jj} L(\mathbf{w}^k) \cdot z_j^2) \le \varepsilon_{\text{in}}, \tag{24}$$

where ε_{in} is the inner stopping tolerance. For the outer stopping condition, similarly, GLMNET checks if $\mathbf{w}$ is still significantly changed. Let $\mathbf{w}^{k+1} = \mathbf{w}^k + \mathbf{d}^k$. GLMNET stops if the following condition holds:

$$\max_j (\nabla^2_{jj} L(\mathbf{w}^{k+1}) \cdot (d_j^k)^2) \le \varepsilon_{\text{out}}, \tag{25}$$

where ε_{out} is the outer stopping tolerance. GLMNET uses the same value for inner and outer tolerances; that is, $\varepsilon_{\text{in}} = \varepsilon_{\text{out}}$. We find that if users specify a small ε_{out}, a huge amount of time may be needed for the first outer iteration. This observation indicates that the inner tolerance must be carefully decided.

For newGLMNET, we propose an adaptive inner stopping condition. The design principle is that in the early stage, newGLMNET should behave like CDN to quickly obtain a reasonable model, while in the final stage, newGLMNET should achieve fast local convergence by using Newton-like directions. In the pth inner iteration p, we assume that $\mathbf{d}^{p,1}, \ldots, \mathbf{d}^{p,n}$ are sequentially generated and from $\mathbf{d}^{p,j}$ to $\mathbf{d}^{p,j+1}$, the jth element is updated. We propose the following inner stopping condition.

$$\sum_{j=1}^{n} |\nabla_j^S q_k(\mathbf{d}^{p,j})| \le \varepsilon_{\text{in}}, \tag{26}$$

where $\nabla^S q(\mathbf{d})$ is the minimum-norm subgradient at $\mathbf{d}$.

$$\nabla_j^S q(\mathbf{d}) \equiv \begin{cases} \nabla_j \bar{q}(\mathbf{d}) + 1 & \text{if } w_j + d_j > 0, \\ \nabla_j \bar{q}(\mathbf{d}) - 1 & \text{if } w_j + d_j < 0, \\ \operatorname{sgn}(\nabla_j \bar{q}(\mathbf{d})) \max(|\nabla_j \bar{q}(\mathbf{d})| - 1, 0) & \text{if } w_j + d_j = 0. \end{cases}$$

From standard convex analysis,

$$\nabla q^S(\mathbf{d}) = \mathbf{0} \quad \text{if and only if } \mathbf{d} \text{ is optimal for (13).} \tag{27}$$

Note that we do not need to calculate the whole $\nabla^S q_k(\mathbf{d}^{p,j})$. Instead, $\nabla_j^S q_k(\mathbf{d}^{p,j})$ is easily available via $\nabla_j \bar{q}_k(\mathbf{d}^{p,j})$ in (16).

If at one outer iteration, the condition (26) holds after only one cycle of n CD steps, then we reduce ε_{in} by

$$\varepsilon_{\text{in}} \leftarrow \varepsilon_{\text{in}}/4. \tag{28}$$

That is, the program automatically adjusts ε_{in} if it finds that too few CD steps are conducted for minimizing $q_k(\mathbf{d})$. Therefore, we can choose a large ε_{in} in the beginning.

We use an outer stopping condition similar to (26).

$$\sum_{j=1}^{n} |\nabla_j^S f(\mathbf{w}^k)| \le \varepsilon_{\text{out}}. \tag{29}$$

Like (27), $\nabla^S f(\mathbf{w}) = \mathbf{0}$ is an optimality condition for (1). In (29), we choose 1-norm instead of ∞-norm because 1-norm is not determined by extreme values in $\nabla_j^S f(\mathbf{w}^k), j = 1, \ldots, n$. In (7), $\nabla_j L(\mathbf{w})$ can be seen as a function of $x_{ij}, \forall i$. It is expected that $|\nabla_j^S f(\mathbf{w})|$ is relatively large if the jth

column of X has a larger norm than other columns. If using ∞-norm, the stopping condition may be dominated by a large $|\nabla_j^S f(\mathbf{w})|$; therefore, under a given ε_{out}, the total number of iterations may be quite different for two feature-wisely norm-varying X's. In contrast, the sum of violations should be less sensitive to the different numeric ranges of X's columns.

5.3 A Two-level Shrinking Scheme

Shrinking is a common technique to heuristically remove some variables during the optimization procedure.[8] That is, some $\mathbf{w}$'s elements are conjectured to be already optimal, so a smaller optimization problem is solved. GLMNET applies this technique on the sub-problem (13) by selecting a working set $J \subset \{1,\ldots,n\}$. Sub-problem (13) becomes

$$\min_{\mathbf{d}} \quad q_k(\mathbf{d}) \quad \text{subject to} \quad d_j = 0, \quad \forall j \notin J. \tag{30}$$

More precisely, at the kth iteration, GLMNET conducts the following loop to sequentially solve some smaller sub-problems.

While (`TRUE`)

- Conduct one cycle of n CD steps. Let J include indices of $\mathbf{d}$'s elements that still need to be changed.
- If (24) holds, then `break`.
- Use CD to solve a sub-problem with the working set J until (24) holds.

The way to choose J in the above procedure is by checking if $z = 0$ is optimal for $\min_z q_k(\mathbf{d} + z\mathbf{e}_j) - q_k(\mathbf{d})$.

For newGLMNET, we propose a heuristic shrinking scheme following its two-level structure: the outer level removes some $\mathbf{w}$'s elements so that a smaller sub-problem (13) similar to (30) is solved; the inner level is applied to remove elements in $\mathbf{d}$ so that (13) becomes an even smaller sub-problem. For each level, our setting is related to the shrinking implementation of CDN; see Yuan et al. (2010, Section 4.1.2).

In the beginning of each outer iteration, we remove w_j if

$$w_j^k = 0 \quad \text{and} \quad -1 + \frac{M^{\text{out}}}{l} < \nabla_j L(\mathbf{w}^k) < 1 - \frac{M^{\text{out}}}{l}, \tag{31}$$

where

$$M^{\text{out}} \equiv \max\left(\left|\nabla_1^S f(\mathbf{w}^{k-1})\right|, \ldots, \left|\nabla_n^S f(\mathbf{w}^{k-1})\right|\right).$$

The conditions in (31) come from the optimality condition that an optimal solution $\mathbf{w}^*$ of (1) satisfies

$$-1 < \nabla_j L(\mathbf{w}^*) < 1 \quad \text{implies} \quad w_j^* = 0.$$

Therefore, we conjecture that variables satisfying (31) are already optimal. M^{out} in (31) is used to adjust the shrinking scheme from a conservative setting in the beginning to an aggressive setting in the end. Our shrinking implementation differs from GLMNET's in several aspects. First, by using $\nabla f(\mathbf{w}^k)$ that is available in the beginning of the kth iteration, we do not conduct a special cycle of n CD steps in GLMNET for selecting variables. Note that $\nabla^S f(\mathbf{w}^k)$ can be easily obtained via $\nabla L(\mathbf{w}^k)$

8. Shrinking is widely used in solving SVM optimization problems; see, for example, Joachims (1998) and Chang and Lin (2011).

Algorithm 3 Overall procedure of newGLMNET

- Given $\mathbf{w}^1$, ε_{in}, and ε_{out}. Choose a small positive number ν. Choose $\beta \in (0,1)$, $\gamma \in [0,1)$, and $\sigma \in (0,1)$.
- Let $M^{\text{out}} \leftarrow \infty$.
- For $k = 1, 2, 3, \ldots$ // outer iterations
  1. Let $J \leftarrow \{1, \ldots, n\}$, $M \leftarrow 0$, and $\bar{M} \leftarrow 0$.
  2. For $j = 1, \ldots, n$
     2.1. Calculate H^k_{jj}, $\nabla_j L(\mathbf{w}^k)$ and $\nabla^S_j f(\mathbf{w}^k)$.
     2.2. If $w^k_j = 0$ and $|\nabla_j L(\mathbf{w}^k)| < 1 - M^{\text{out}}/l$ // outer-level shrinking
          $J \leftarrow J \backslash \{j\}$.
          Else
          $M \leftarrow \max(M, |\nabla^S_j f(\mathbf{w}^k)|)$ and $\bar{M} \leftarrow \bar{M} + |\nabla^S_j f(\mathbf{w}^k)|$.
  3. If $\bar{M} \le \varepsilon_{\text{out}}$
     `return` $\mathbf{w}^k$.
  4. Let $M^{\text{out}} \leftarrow M$.
  5. Get $\mathbf{d}$ and update ε_{in} by solving sub-problem (13) by Algorithm 4.
  6. Compute $\lambda = \max\{1, \beta, \beta^2, \ldots\}$ such that $\lambda \mathbf{d}$ satisfies (20).
  7. $\mathbf{w}^{k+1} = \mathbf{w}^k + \lambda \mathbf{d}$.

and is used for obtaining M^{out} of the next iteration.[9] Second, (31) shrinks only zero elements and uses an interval slightly smaller than $(-1, 1)$. Thus, newGLMNET is less aggressive than GLMNET in removing variables.

For the inner shrinking scheme of newGLMNET, assume the previous CD cycle contains points $\mathbf{d}^{p-1,1}, \ldots, \mathbf{d}^{p-1,|J^p|}$, where elements in the set $J^p = \{j_1, \ldots, j_{|J^p|}\}$ were updated. Because J^p corresponds to the remained variables, at the current cycle, sequentially $j \in J^p$ is checked. An element j is removed if

$$w^k_j + d^{p,t}_j = 0 \quad \text{and} \quad -1 + \frac{M^{\text{in}}}{l} < \nabla_j \bar{q}_k(\mathbf{d}^{p,t}) < 1 - \frac{M^{\text{in}}}{l}, \tag{32}$$

where t is the iteration index of the current cycle and

$$M^{\text{in}} \equiv \max\left(\left|\nabla^S_{j_1} q_k(\mathbf{d}^{p-1,1})\right|, \ldots, \left|\nabla^S_{j_{|J^p|}} q_k(\mathbf{d}^{p-1,|J^p|})\right|\right).$$

If (32) does not hold, element j remains.[10] After the set J^p has been processed, a smaller subset J^{p+1} is obtained and we move to the next CD cycle.[11]

The overall procedure of newGLMNET with two-level shrinking is shown in Algorithms 3 and 4. For theoretical properties, if the subproblem (13) is exactly solved, for any given outer stopping tolerance, newGLMNET terminates in finite iterations. Further, any limit point of $\{\mathbf{w}^k\}$ is an optimal solution. More details are in the supplementary document.

5.4 The Overall Procedure of newGLMNET

We use Algorithms 3 to illustrate the overall procedure of newGLMNET. Steps 1–4 are for outer-level shrinking and the stopping condition. In particular, in Step 2.2, M is used to calculate M^{out}

9. If $k = 1$, $\nabla^S f(\mathbf{w}^{k-1})$ is not available. We set $M^{\text{out}} = \infty$, so no variables are shrunk at the first outer iteration.
10. Note that in (32), $t = 1, \ldots, |J^{p+1}|$ instead of $1, \ldots, |J^p|$.
11. Similar to the way to initialize M^{out}, for the first CD cycle, we set $M^{\text{in}} = \infty$.

Algorithm 4 Inner iterations of newGLMNET with shrinking

- Given working set J, initial solution $\mathbf{d}$, inner stopping condition ε_{in}, and a small positive number ν from the outer problem.
- Let $M^{\text{in}} \leftarrow \infty$, $T \leftarrow J$, and $\mathbf{d} \leftarrow \mathbf{0}$.
- For $p = 1, 2, 3, \ldots, 1000$ // inner iterations
  1. Let $m \leftarrow 0$ and $\bar{m} \leftarrow 0$.
  2. For $j \in T$
     - Let $\nabla^2_{jj} \bar{q}_k(\mathbf{d}) = H^k_{jj}$. Calculate $\nabla_j \bar{q}_k(\mathbf{d})$ and $\nabla^S_j q_k(\mathbf{d})$.
     - If $w^k_j + d_j = 0$ and $|\nabla_j \bar{q}_k(\mathbf{d})| < 1 - M^{\text{in}}/l$ // inner-level shrinking
       $T \leftarrow T \backslash \{j\}$.

       Else
       $m \leftarrow \max(m, |\nabla^S_j q_k(\mathbf{d})|)$ and $\bar{m} \leftarrow \bar{m} + |\nabla^S_j q_k(\mathbf{d})|$.
       $d_j \leftarrow d_j + \arg\min_z q_k(\mathbf{d} + z\mathbf{e}_j) - q_k(\mathbf{d})$.
  3. If $\bar{m} \le \varepsilon_{\text{in}}$
     - If $T = J$ // inner stopping
       `break`.

       Else // active set reactivation
       $T \leftarrow J$ and $M^{\text{in}} \leftarrow \infty$.

     Else
     - $M^{\text{in}} \leftarrow m$.
- If $p = 1$, then $\varepsilon_{\text{in}} \leftarrow \varepsilon_{\text{in}}/4$.

in (31) for the outer-level shrinking, while $\bar{M}$ is for calculating $\sum_{j=1}^n |\nabla^S_j f(\mathbf{w})|$ in the outer stopping condition (29). Step 5 obtains the Newton direction by a CD method, where details are shown in Algorithm 4. Step 6 then conducts line search.

In Algorithm 4, besides the stopping condition (26), we set 1,000 as the maximal number of CD cycles. Some ill-conditioned sub-problem (13) may take lengthy inner CD iterations to satisfy (26), so a maximal number must be set. In the beginning of Algorithm 4, the working set J is obtained from outer-level shrinking. Subsequently, in the inner CD iterations, J is further shrunk; see the set T in Algorithm 4. Because each CD cycle goes through only elements in T, the inner stopping condition (26) is also calculated using T. To ensure that sub-problem (13) with the working set J has been accurately solved, if (26) holds on T, we reset T to J; see Step 3 of Algorithm 4. That is, the inner iterations terminate only if the condition (26) holds on J or the maximal number of iterations is reached. This way of resetting T to J has been used in LIBSVM (Chang and Lin, 2011, Section 5.1).

In (28), we reduce the inner stopping tolerance ε_{in} if the stopping condition (26) holds after one CD cycle. This is implemented in the last step of Algorithm 4.

6. Experiments on L1-regularized Logistic Regression

We investigate the performance of CDN, GLMNET, and newGLMNET on L1-regularized logistic regression. All these methods can be easily extended to solve logistic regression with a bias term b:

$$\min_{\mathbf{w},b} \quad \|\mathbf{w}\|_1 + C \sum_{i=1}^{l} \log(1 + e^{-y_i(\mathbf{w}^T \mathbf{x}_i + b)}). \tag{33}$$

Data set	#data		#features	#nnz in	#nnz per	Sparsity	
	train	test		training	instance	C	(%)
KDD2010-b	19,264,097	748,401	29,890,095	566,345,888	29	0.5	97.4
rcv1	541,920	135,479	47,236	39,625,144	73	4	76.9
yahoo-japan	140,963	35,240	832,026	18,738,315	133	4	99.0
yahoo-korea	368,444	92,110	3,052,939	125,190,807	340	4	99.1
news20	15,997	3,999	1,355,191	7,281,110	455	64	99.1
epsilon	400,000	100,000	2,000	800,000,000	2,000	0.5	44.9
webspam	280,000	70,000	16,609,143	1,043,724,776	3,727	64	99.9
gisette	6,000	1,000	5,000	29,729,997	4,955	0.25	72.9

Table 3: Data statistics, the parameter C selected after cross validation, and the model sparsity (%). Data sets are sorted by the number of nonzero elements per instance in the training data. We conduct five-fold cross validation on the training set to select C in $\{2^k \mid k = -4, -3, \ldots, 6\}$. The model sparsity is the percentage of the number of zeros in the final model $\mathbf{w}$. #nnz denotes the number of nonzero elements.

Because the GLMNET implementation solves (33) instead of (1), in our comparison, (33) is used. We do not consider other methods because in Yuan et al. (2010), CDN is shown to be the best for sparse data.

Programs used in this paper are available at `http://www.csie.ntu.edu.tw/~cjlin/liblinear/exp.html`.

6.1 Data Sets and Experimental Settings

We use eight data sets in our experiments. Five of them (news20, rcv1, yahoo-japan, yahoo-korea, and webspam) are document data sets, where news20 is a collection of news documents, rcv1 is an archive of manually categorized news stories from Reuters, yahoo-japan and yahoo-korea are document data from Yahoo!, and webspam includes web pages represented in trigram. The other three data sets come from different learning problems: gisette is a handwriting digit recognition problem from NIPS 2003 feature selection challenge; epsilon is an artificial data set for Pascal large scale learning challenge in 2008; KDD2010-b includes student performance prediction data for a tutoring system and is used for the data mining competition KDD Cup 2010. Each instance in document data sets is normalized to a unit vector. For non-document data, features of gisette are linearly scaled to the $[-1,1]$ interval. Features of epsilon are scaled to $N(0,1)$ and each instance is normalized to a unit vector. Except yahoo-japan and yahoo-korea, all data sets and their detailed information are publicly available at `http://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/`.

We prepare training and testing sets for each problem. For gisette and KDD2010-b, we use their original training and test sets. For others, we randomly split data to one fifth for testing and the remaining for training.

We choose the parameter C in (33) by five-fold cross validation on the training set. All methods then solve (33) with the best C to obtain the model for prediction. Table 3 shows the statistics and the best C of all data sets. We can clearly see that two data sets (epsilon and gisette) are very dense, while others are sparse.

Next, we describe software information and parameter settings in our experiments.

- CDN: this coordinate descent method is described in Section 2.1. In the line search procedure, we use $\sigma = 0.01$ and $\beta = 0.5$. The C/C++ implementation is included in LIBLINEAR (version 1.7), which is available at `http://www.csie.ntu.edu.tw/~cjlin/liblinear/oldfiles`; see the implementation document (Fan et al., 2008) for more details.

- GLMNET: this method is described in Section 3.2. GLMNET is implemented in Fortran with an R interface. The source code (version 1.5.3) is available at `http://cran.r-project.org/web/packages/glmnet/`. GLMNET uses the regularization parameter $\lambda = 1/(Cl)$ instead of C. We ensure that the equivalent settings have been made in our experiments.

- newGLMNET: this improved GLMNET is described in Sections 4 and 5. For the positive definiteness of H^k, we set $\nu = 10^{-12}$ in (19). To check the sufficient decrease condition (20), we use $\beta = 0.5$, $\gamma = 0$, and $\sigma = 0.01$. We choose the initial $\varepsilon_{\text{in}} = \|\nabla^S f(\mathbf{w}^1)\|_1$. The C/C++ implementation is included in LIBLINEAR (version 1.8).

GLMNET offers an option to find a solution path $\{\mathbf{w}^{C_1}, \ldots, \mathbf{w}^{C_*}\}$ of an increasing parameter sequence $\{C_1, \ldots, C_*\}$. It applies a warm start technique so that the optimal solution of the previous C_{i-1} is used as the initial point for the current C_i. The number of outer iterations should be small because of using a more accurate initial point. GLMNET authors suggest that finding a solution path may be faster than solving a single optimization problem under a fixed C (Friedman et al., 2010, Section 2.5). We refer to this approach as GLMNETpath and include it for comparison. By their default setting, we consider a parameter sequence of length 100 starting from the smallest C_1 such that $\mathbf{w}^{C_1} = \mathbf{0}$. Given our desired parameter C_*, a geometric sequence is generated by a fixed ratio between successive C values.

We set the initial $\mathbf{w}^1 = \mathbf{0}$ for all methods. All experiments are conducted on a 64-bit machine with Intel Xeon 2.0GHz CPU (E5504), 4MB cache, and 32GB main memory. We use GNU C/C++/Fortran compilers and the optimization flag is properly set.

6.2 Running Time Comparison

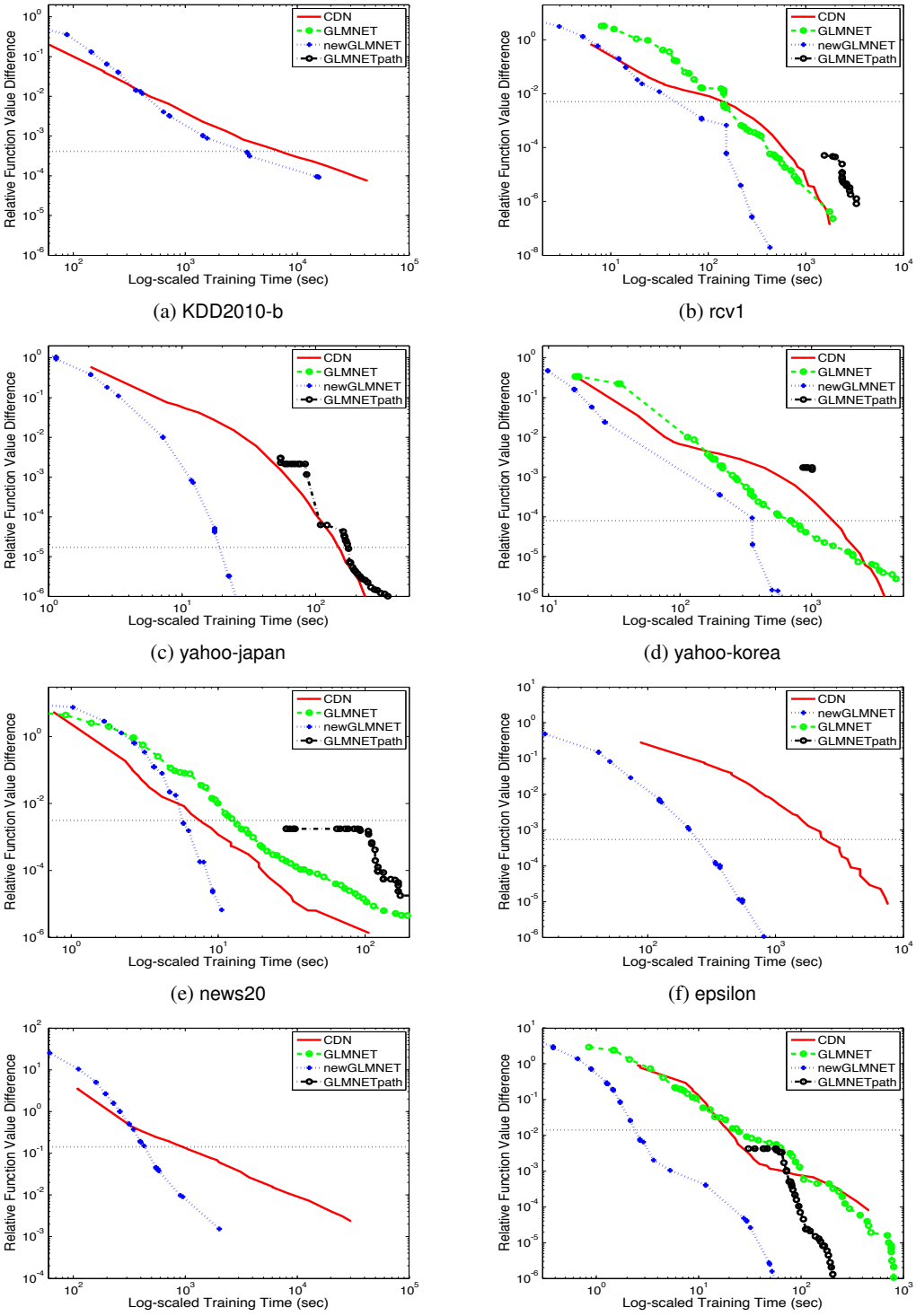
We begin with checking the change of function values along the running time in Figure 1. Given a stopping tolerance for running a solver, we can obtain a pair of (training time, function value). Using a decreasing sequence of the stopping tolerances, we obtain several pairs and then draw a curve.[12] The x-axis in Figure 1 is the log-scaled training time and the y-axis is the relative difference to the optimal function value:

$$\frac{f(\mathbf{w}) - f^*}{f^*},$$

where $\mathbf{w}$ is the solution under the specified tolerance and f^* is the optimal function value. Because f^* is not available, we obtain an approximation by running newGLMNET with a small stopping tolerance

$$\varepsilon_{\text{out}} = \varepsilon \cdot \frac{\min(\#\text{pos}, \#\text{neg})}{l} \cdot \|\nabla^S f(\mathbf{w}^1)\|_1, \tag{34}$$

12. For GLMNET and newGLMNET, the tolerance means the outer tolerance ε_{out} in (25). Ranges of ε_{out} values used for GLMNET and newGLMNET differ because their stopping conditions are not the same. Note that for GLMNET, ε_{out} is not directly specified by users; instead, it is the product between a user-specified value and a constant. For GLMNETpath, under any given ε_{out}, a sequence of problems (1) is solved.



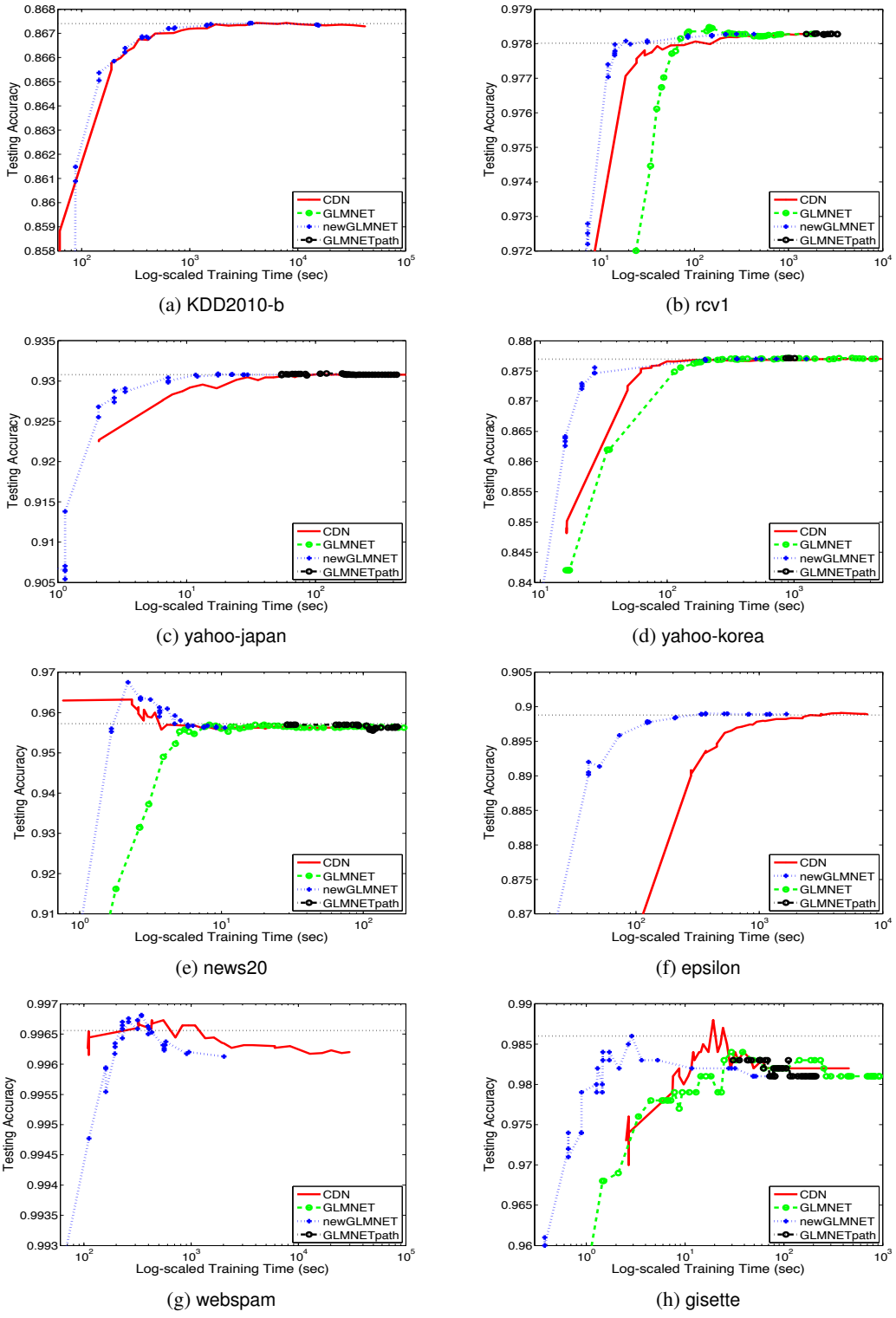
(a) KDD2010-b (b) rcv1

(c) yahoo-japan (d) yahoo-korea

(e) news20 (f) epsilon

(g) webspam (h) gisette

Figure 1: L1-regularized logistic regression: relative difference to the optimal function value versus training time. Both x-axis and y-axis are log-scaled. GLMNET and GLMNETpath failed to generate some results because of either memory problems or lengthy running time.



(a) KDD2010-b (b) rcv1

(c) yahoo-japan (d) yahoo-korea

(e) news20 (f) epsilon

(g) webspam (h) gisette

Figure 2: L1-regularized logistic regression: testing accuracy versus training time (log-scaled).

where $\varepsilon = 10^{-8}$, and #pos and #neg indicate the numbers of positive and negative labels in the training set, respectively. The horizontal dotted line in Figure 1 indicates the relative function difference by running CDN using LIBLINEAR's default stopping tolerance with $\varepsilon = 0.01$ in (34). The point where a method's curve passes this horizontal line roughly indicates the time needed to obtain an accurate enough solution.

From Figure 1, if the optimization problem is loosely solved using a large ε_{out}, CDN is faster than newGLMNET and GLMNET. This result is reasonable because CDN uses a greedy setting to sequentially update variables. In contrast, in each outer iteration, newGLMNET uses only a fixed H^k. If using a smaller ε_{out}, newGLMNET surpasses CDN and achieves fast local convergence. For dense data (epsilon and gisette), newGLMNET is always better than CDN. Take epsilon as an example. In Figure 1(f), to reach the horizontal dotted line, newGLMNET is ten times faster than CDN. This huge difference is expected following the analysis on the number of exp/log operations in Sections 2 and 3.

From results above the horizontal lines in Figure 1, we see that newGLMNET is faster than GLMNET in the early stage. Recall that GLMNET sets $\varepsilon_{\text{in}} = \varepsilon_{\text{out}}$, while newGLMNET uses an adaptive setting to adjust ε_{in}. Because a large ε_{in} is considered in the beginning, newGLMNET can compete with CDN in the early stage by loosely solving (13). We use an example to further illustrate the importance to properly choose ε_{in}. By running GLMNET with the default $\varepsilon_{\text{out}} = 10^{-6}$ on news20 and rcv1, the training time is 20.10 and 758.82 seconds, respectively. The first outer iteration already takes 6.99 seconds on news20 and 296.87 on rcv1. A quick fix is to enlarge the initial ε_{in}, but the local convergence in the later stage may be slow. A better inner stopping condition should be adaptive like ours so that the sub-problem (13) can be solved properly at each outer iteration.

In Figure 1, GLMNET and GLMNETpath failed to generate some results because of either memory problems or lengthy running time. This indicates that a careful implementation is very important for large-scale problems. We also observe that GLMNETpath is not faster than GLMNET. Another drawback of GLMNETpath is that it is hard to quickly obtain an approximate solution. That is, regardless of ε_{out} specified, a sequence of problems (1) is always solved.

We further check the relationship between the testing accuracy and the training time. The comparison result, shown in Figure 2, is similar to that in Figure 1.

In summary, because of the proposed adaptive inner stopping condition, newGLMNET takes both advantages of fast approximation in the early stage like CDN and of fast local convergence in the final stage like GLMNET.

6.3 Analysis on Line Search

Recall in Section 4 we added a line search procedure in newGLMNET. To check if line search costs much in newGLMNET, we report the average number of line search steps per outer iteration in Table 4. Clearly, in all cases, $\lambda = 1$ satisfies the sufficient decrease condition (20), so conducting line search to ensure the convergence introduces very little cost. As a comparison, we also, respectively, show the average numbers of updated variables and line search steps per cycle of CDN in the first and second columns of the same table. Note that because a shrinking technique is applied to CDN, the number of updated variables in one cycle is much smaller than the number of features. We see that the number of line search steps is very close to the number of updated variables in a CD cycle. Therefore, in most cases, (10) holds when $\lambda = 1$. Although line search in both CDN and newGLMNET often succeeds at the first step size $\lambda = 1$, from Section 4.1, their numbers of operations are

Data set	CDN			newGLMNET
	#variables in a CD cycle	#λ tried in line search	# δ(λ) successfully applied in (23)	#λ tried in an outer iteration
KDD2010-b	630,455	630,588	622,267	1
rcv1	11,396	11,398	449	1
yahoo-japan	8,269	8,270	922	1
yahoo-korea	27,103	27,103	1,353	1
news20	6,395	6,396	2,413	1
epsilon	1,130	1,130	0	1
webspam	17,443	17,444	3,389	1
gisette	1,121	1,121	0	1

Table 4: Logistic regression: the average number of line search steps per CD cycle of CDN and per outer iteration of newGLMNET. The data are collected by running CDN and newGLMNET using the best C and the default stopping condition of LIBLINEAR.

very different. The $O(nl)$ cost of CDN can be significantly reduced due to shrinking, but is still more expensive than $O(n+l)$ of newGLMNET.

In Section 4.1, we mentioned an $O(1)$-cost upper-bound function $\delta(\lambda)$ to efficiently check (10) in line search of CDN. In Table 4, we further report the average number of line search steps in a CD cycle where this check is successfully applied. We see that the trick is particularly effective on KDD2010-b; however, it helps in a limited manner on other data sets. For KDD2010-b, in addition to a small nnz$/l$, the faster line search is another possible reason why CDN is comparable to newGLMNET; see Figure 1(a). For gisette and epsilon, the trick is not useful because the assumption $x_{ij} \geq 0,\ \forall i, j$ needed for deriving the upper-bound function does not hold.

6.4 Effect of Exp/log Operations

In Sections 2–3, we pointed out the difference between CDN's $O(\text{nnz})$ and newGLMNET's $O(l)$ exp/log operations per CD cycle. Figures 1–2 confirm this result because newGLMNET is much faster for the two dense data (epsilon and gisette). We further extend Table 1 to compare the running time of the first CD cycle in Table 5. For easy comparison, we deliberately sort data sets in all tables of this section by nnz$/l$, which is the average number of non-zero values per instance. We expect the ratio of time spent on exp/log operations gradually increases along with nnz$/l$, although this is not very clearly observed in Table 5. The reason might be either that the number of data sets used is small or other data characteristics affect the running time.

6.5 Approximate Exponential Operations for CDN

Because CDN suffers from slow exp/log operations, we tried to use the approximate exponentiation proposed by Schraudolph (1999). However, we failed to speed up CDN because of erroneous numerical results. One of the several possible reasons is that when d in (11) is small, the approximate $\exp(dx_{ij})$ is inaccurate. The inaccurate $\exp(dx_{ij})$ makes $|\log(1+e^{-\mathbf{w}^T\mathbf{x}}\cdot e^{-dx_{ij}}) - \log(1+e^{-\mathbf{w}^T\mathbf{x}})|$ have a large relative error because the correct value is near zero with small d. We may encounter this problem when calculating the function value difference needed by the sufficient decrease condition

Data set	CDN		newGLMNET	
	exp/log	Total	exp/log	Total
KDD2010-b	21.72 (30.7%)	70.80	3.88 (6.9%)	56.50
rcv1	4.61 (73.8%)	6.25	0.12 (5.3%)	2.20
yahoo-japan	1.47 (70.9%)	2.08	0.03 (4.2%)	0.71
yahoo-korea	10.65 (66.3%)	16.06	0.08 (1.2%)	6.66
news20	0.21 (27.3%)	0.76	0.003 (0.5%)	0.60
epsilon*	64.25 (73.0%)	88.18	0.08 (0.7%)	11.62
webspam	72.89 (66.6%)	109.39	0.06 (0.1%)	41.10
gisette*	1.66 (66.8%)	2.49	0.002 (0.6%)	0.27

Table 5: Timing analysis of the first cycle of n CD steps. Time is in seconds. (*: dense data)

(10). In our experiment, the function-value difference using approximate exp/log operations tend to be larger than the correct value; therefore, it is hard to find a step size $\bar{\lambda}$ satisfying (10). Consequently, if d is small when the current $\mathbf{w}^k$ is near the optimal solution, line search terminates with a very small step size $\bar{\lambda}$ and results in bad convergence. Furthermore, because we update $e^{\mathbf{w}^T\mathbf{x}}$ by (11), the error is accumulated. Schraudolph (1999, Section 5) has mentioned that the approximation may not be suitable for some numerical methods due to error amplification. We also tried different reformulation of $\log(1+e^{-\mathbf{w}^T\mathbf{x}}\cdot e^{-dx_{ij}})-\log(1+e^{-\mathbf{w}^T\mathbf{x}})$, but still failed to use an approximate exponential function.

6.6 Effect of Shrinking

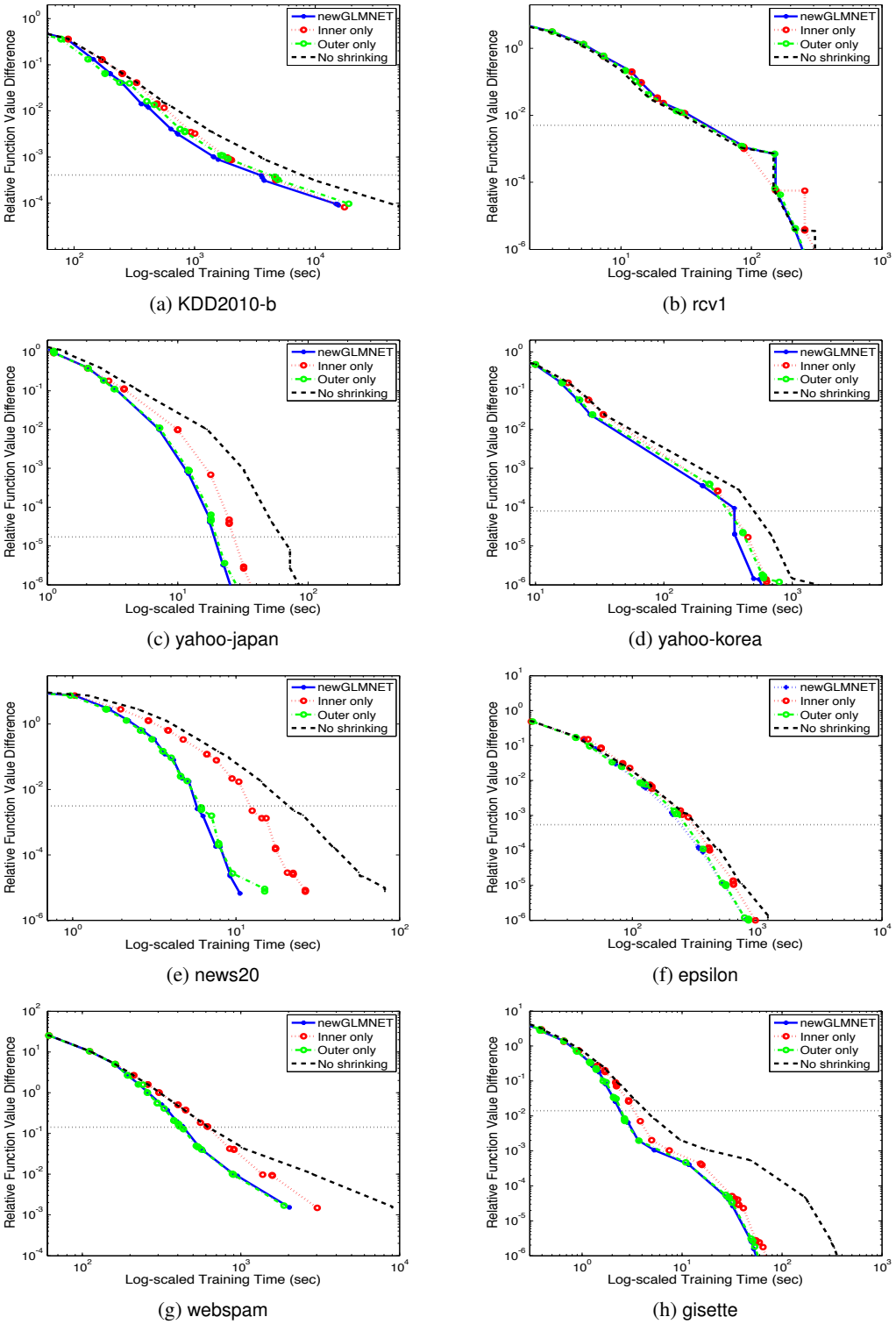
Our investigation contains two parts. First, we investigate the effect of newGLMNET's two levels of shrinking by presenting results of only inner or outer level. Secondly, we compare the shrinking strategies of GLMNET and newGLMNET. Because these two implementations differ in many places, for a fair comparison, we modify newGLMNET to apply GLMNET's shrinking strategy. The comparison results are presented in Figure 3. We can clearly see that all shrinking implementations are better than the one without shrinking.

Results in Figure 3 show that the outer-level shrinking is more useful than the inner-level shrinking. We suspect that the difference is due to that in the CD procedure for sub-problem (13), the (inner-level) shrinking is done in a sequential manner. Thus, not only is M^{in} not calculated based on the gradient at the same point, but also variables are not removed together. In contrast, for the outer-level shrinking, M^{out} is calculated by the gradient at $\mathbf{w}^{k-1}$ and all variables are checked together. Therefore, the outer-level shrinking is a more integrated setting for checking and removing variables. The same explanation may also apply to the result that shrinking is slightly more effective for newGLMNET than CDN; see Yuan et al. (2010, Figure 9) and Figure 3 here.

Regarding GLMNET's shrinking strategy, it performs slightly better than the inner-level shrinking of newGLMNET, but is worse than both the outer-level and the two-level settings.

7. Using newGLMNET to Solve Problems with Cheap Loss Functions

The analysis and experiments in previous sections have shown that newGLMNET is more efficient than CDN for logistic regression. However, it is not clear if newGLMNET is superior when a loss



(a) KDD2010-b (b) rcv1

(c) yahoo-japan (d) yahoo-korea

(e) news20 (f) epsilon

(g) webspam (h) gisette

Figure 3: Effect of two-level shrinking. "Inner only" ("Outer only") indicates that only inner-level (outer-level) shrinking is conducted.

function can be calculated cheaply. In this section, we consider the L2-loss function in Equation (3) and investigate the performance of newGLMNET in comparison with CDN. The CDN algorithm for L2-loss SVM has been developed in Fan et al. (2008, Appendix F) and Yuan et al. (2010, Section 7).

We briefly describe how to apply newGLMNET to solve L2-loss SVM. The objective function can be written as

$$f(\mathbf{w}) \equiv \|\mathbf{w}\|_1 + C \sum_{i \in I(\mathbf{w})} b_i(\mathbf{w})^2,$$

where

$$b_i(\mathbf{w}) \equiv 1 - y_i \mathbf{w}^T \mathbf{x}_i \quad \text{and} \quad I(\mathbf{w}) \equiv \{i \mid b_i(\mathbf{w}) > 0\}.$$

Similar to (2), we define

$$L(\mathbf{w}) \equiv C \sum_{i \in I(\mathbf{w})} b_i(\mathbf{w})^2.$$

The gradient of $L(\mathbf{w})$ is

$$\nabla L(\mathbf{w}) = -2C \sum_{i \in I(\mathbf{w})} b_i(\mathbf{w}) y_i \mathbf{x}_i. \tag{35}$$

Different from logistic loss, $L(\mathbf{w})$ is not twice differentiable. Following Mangasarian (2002) and Yuan et al. (2010), we consider the following generalized Hessian:

$$\nabla^2 L(\mathbf{w}) = 2CX^T DX, \tag{36}$$

where $D \in \mathbf{R}^{l \times l}$ is a diagonal matrix with

$$D_{ii} = \begin{cases} 1 & \text{if } b_i(\mathbf{w}) > 0, \\ 0 & \text{otherwise.} \end{cases}$$

At the kth outer iteration, newGLMNET solves a quadratic sub-problem

$$\min_{\mathbf{d}} \quad q_k(\mathbf{d}), \tag{37}$$

where

$$\begin{aligned} q_k(\mathbf{d}) &\equiv \|\mathbf{w}^k + \mathbf{d}\|_1 - \|\mathbf{w}^k\|_1 + \bar{q}_k(\mathbf{d}), \\ \bar{q}_k(\mathbf{d}) &\equiv \nabla L(\mathbf{w}^k)^T \mathbf{d} + \frac{1}{2} \mathbf{d}^T H^k \mathbf{d} \quad \text{and} \quad H^k \equiv \nabla^2 L(\mathbf{w}^k) + \nu I. \end{aligned}$$

To minimize (37), we also use a CD procedure to sequentially minimize one-variable functions at each inner iteration p.

$$\begin{aligned} & q_k(\mathbf{d}^{p,j} + z\mathbf{e}_j) - q_k(\mathbf{d}^{p,j}) \\ &= |w_j^k + d_j^p + z| - |w_j^k + d_j^p| + \nabla_j \bar{q}_k(\mathbf{d}^{p,j}) z + \frac{1}{2} \nabla_{jj}^2 \bar{q}_k(\mathbf{d}^{p,j}) z^2, \end{aligned} \tag{38}$$

where from (35) and (36),

$$\begin{aligned} \nabla_j \bar{q}_k(\mathbf{d}^{p,j}) &= \nabla_j L(\mathbf{w}^k) + (H^k \mathbf{d}^{p,j})_j \\ &= -2C \sum_{i \in I(\mathbf{w}^k)} b_i(\mathbf{w}^k) y_i x_{ij} + 2C \sum_{i \in I(\mathbf{w}^k)} (X^T)_{ji} D_{ii} (X \mathbf{d}^{p,j})_i + \nu d_j^p \quad \text{and} \\ \nabla_{jj}^2 \bar{q}_k(\mathbf{d}^{p,j}) &= \nabla_{jj}^2 L(\mathbf{w}^k) + \nu = 2C \sum_{i \in I(\mathbf{w}^k)} x_{ij}^2 + \nu. \end{aligned}$$

The CD procedure is almost the same as the one described in Section 3.2 for logistic regression. The function in (38) can be exactly minimized by (9) and line search is not needed. Moreover, $X\mathbf{d}^{p,j}$ is maintained by (17), so the cost per CD cycle is the same as that shown in (18).

7.1 Line Search and Asymptotic Convergence

At every outer iteration, after $\mathbf{d}$ is obtained by solving sub-problem (37), we need a line search procedure to find the maximal $\lambda \in \{\beta^i \mid i = 0, 1, \dots\}$ such that (20) is satisfied. Following the discussion in Section 4.1, the computational bottleneck is on calculating $(\mathbf{w}^k + \lambda\mathbf{d})^T\mathbf{x}_i, \forall i$. Similar to the trick in Equation (21), we maintain $b_i(\mathbf{w}^k),\ \forall i$ to save the cost. In line search, we use

$$\begin{aligned} b_i(\mathbf{w}^k + \beta^t\mathbf{d}) &= 1 - y_i(\mathbf{w}^k + \beta^{t-1}\mathbf{d})^T\mathbf{x}_i + (\beta^{t-1} - \beta^t)y_i(X\mathbf{d})_i \\ &= b_i(\mathbf{w}^k + \beta^{t-1}\mathbf{d}) + (\beta^{t-1} - \beta^t)y_i(X\mathbf{d})_i \end{aligned}$$

for calculating $f(\mathbf{w}^k + \beta^t\mathbf{d})$. The last $b_i(\mathbf{w}^k + \beta^t\mathbf{d})$ is passed to the next outer iteration as $b_i(\mathbf{w}^{k+1})$.

In Appendix C, we prove that newGLMNET for L2-loss SVM is an example of Tseng and Yun's framework, so the finite termination of line search holds and any limit point of $\{\mathbf{w}^k\}$ is an optimal solution.

7.2 Comparison with CDN

The analysis in Section 2 indicates that CDN needs more exp/log operations than newGLMNET. Experiments in Section 6.4 confirm this analysis by showing that CDN is much slower than newGLMNET on dense data. However, the situation for L2-loss SVM may be completely different because exp/log operations are not needed. Without this advantage, whether newGLMNET can still compete with CDN is an interesting question. We will answer this question by experiments in Section 7.3.

Following the analysis in Section 4.1, the cost of line search is still much different between CDN and newGLMNET for L2-loss SVM. For each cycle of n CD steps, the $O(nl)$ cost is required in CDN, while less than $O(n+l)$ is required in newGLMNET. For the high cost of line search in CDN, Yuan et al. (2010) also find out an upper-bound function like (22), which can be obtained in $O(1)$; see Fan et al. (2008, Appendix F) for more details. If this trick succeeds at $\lambda = 1$ in every CD step of a cycle, then the $O(nl)$ cost is reduced to $O(l)$. In Section 7.3, we check if this trick is useful.

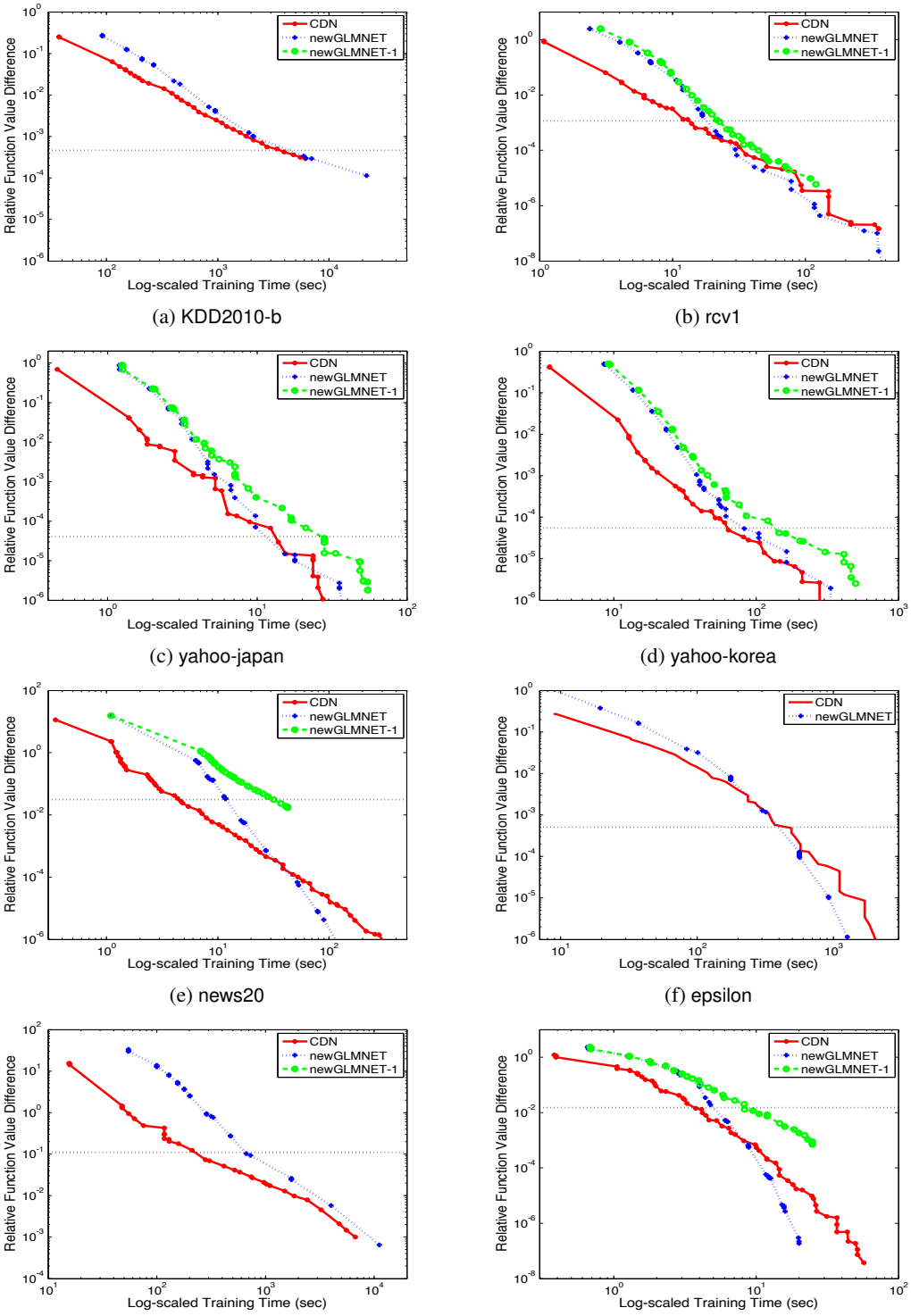
7.3 Experiments

We compare CDN and newGLMNET under a similar experimental setting to that for logistic regression. Different from (33), we solve L2-loss SVM without a bias term b.[13]

We plot the relative difference to the optimal function value in Figure 4. The reference f^* is obtained by running newGLMNET with a strict stopping tolerance $\varepsilon_{\text{out}} = 10^{-8}$. Figure 5 presents the testing accuracy along training time. We can clearly see that CDN is much faster than newGLMNET in the early stage. While newGLMNET still enjoys fast local convergence, it catches up with CDN only in the very end of the training procedure. This result is consistent with our analysis in Section 7.2 showing that newGLMNET loses the advantages of taking fewer exp/log operations.

In Table 6, we analyze the line search procedure by a setting like Table 4. Similar results are observed: the sufficient decrease condition (20) always holds when $\lambda = 1$ for newGLMNET;

13. Earlier we solved problem (33) in order to compare with the GLMNET implementation by Friedman et al..



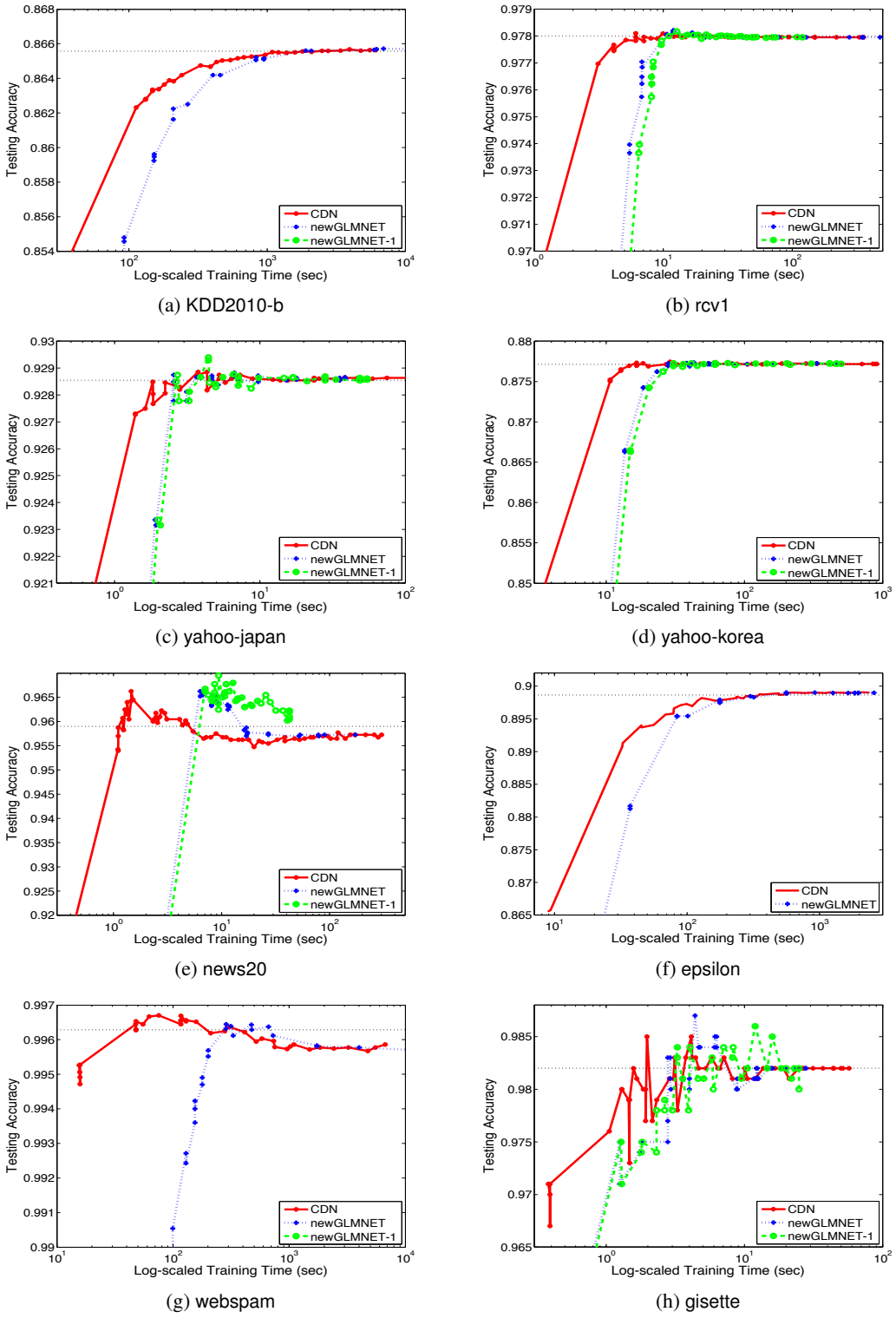
(a) KDD2010-b (b) rcv1

(c) yahoo-japan (d) yahoo-korea

(e) news20 (f) epsilon

(g) webspam (h) gisette

Figure 4: L1-regularized L2-loss SVM: relative difference to the optimal function value versus training time. Both x-axis and y-axis are log-scaled.



(a) KDD2010-b (b) rcv1

(c) yahoo-japan (d) yahoo-korea

(e) news20 (f) epsilon

(g) webspam (h) gisette

Figure 5: L1-regularized L2-loss SVM: testing accuracy versus training time (log-scaled).

Data set	CDN #variables in a CD cycle	CDN #λ tried in line search	CDN # δ(λ) successfully applied in (23)	newGLMNET #λ tried in an outer iteration
KDD2010-b	246,318	248,151	124,175	1
rcv1	13,350	13,384	1,251	1
yahoo-japan	10,286	10,289	4,931	1
yahoo-korea	31,265	31,270	25,711	1
news20	7,688	7,838	1,461	1
epsilon	1,136	1,137	501	1
webspam	8,165	8,312	361	1
gisette	1,145	1,145	76	1

Table 6: L2-loss SVM: the average number of line search steps per CD cycle of CDN and newGLMNET. The data are collected by running CDN and newGLMNET using the best C and the default stopping tolerance.

moreover, for CDN, $\lambda = 1$ is successful almost all the time. One difference is that the trick of using an upper-bound function in CDN is slightly more effective for L2-loss SVM than logistic regression.

8. Discussions and Conclusions

In newGLMNET, a CD method is applied to solve the sub-problem (13). Using the property that CD involves simple and cheap updates, we carefully adjust the stopping condition for sub-problems. Then, newGLMNET is competitive with a CD method like CDN in the early stage, but becomes a Newton method in the end. This design is similar to "truncated Newton" methods in optimization. While CD seems to be a very good choice for solving the sub-problem, whether there are better alternatives is an interesting future issue.

In Section 5, we proposed several implementation techniques for newGLMNET. For shrinking, we consider thresholds M^{out}/l and M^{in}/l in (31) and (32), respectively. These values are heuristically chosen. While it is difficult to find an optimal setting for all data sets, we hope to investigate if the current thresholds are suitable.

Some recent works such as El Ghaoui et al. (2010) and Tibshirani et al. (2011) proposed rules to cheaply eliminate features prior to the L1 training. Preliminary results in the supplementary document show that training is more efficient if we can remove some zero variables beforehand. How to efficiently and correctly identify these variables before training is an interesting future topic.

In our newGLMNET implementation, the sub-problem (13) is approximately solved by CD. However, so far we only establish the convergence results of newGLMNET under the assumption that the sub-problem (13) is exactly solved. In the future, we will strive to address this issue.

In this work, we point out that a state-of-the-art algorithm CDN for L1-regularized logistic regression suffers from frequent exp/log operations. We then demonstrate that Newton-type methods can effectively address this issue. By improving a Newton-type method GLMNET in both theoretical and practical aspects, the proposed newGLMNET is more efficient than CDN for logistic regression. The difference is huge for dense data. However, if a loss function is cheap to compute (e.g., L2 loss),

CDN is still competitive. Based on this research work, we have replaced CDN with newGLMNET as the solver of L1-regularized logistic regression in the software LIBLINEAR.

Acknowledgments

This work was supported in part by the National Science Council of Taiwan via the grant 98-2221-E-002-136-MY3. The authors thank Mark Schmidt, associate editor, and anonymous reviewers for valuable comments.

Appendix A. Convergence of newGLMNET for L1-regularized Logistic Regression

We have explained that newGLMNET is in the framework of Tseng and Yun (2009). Thus, it is sufficient to check conditions needed for their convergence result.

To have the finite termination of the line search procedure, Tseng and Yun (2009, Lemma 5) require that there exists $\Lambda > 0$ such that

$$\|\nabla L(\mathbf{w}_1) - \nabla L(\mathbf{w}_2)\| \le \Lambda \|\mathbf{w}_1 - \mathbf{w}_2\|, \ \forall \mathbf{w}_1, \mathbf{w}_2 \in \mathbf{R}^n \tag{39}$$

and

$$H^k \succ 0. \tag{40}$$

Note that "$A \succ B$" indicates that $A - B$ is positive definite.

Because $L(\mathbf{w})$ is twice differentiable,

$$\|\nabla L(\mathbf{w}_1) - \nabla L(\mathbf{w}_2)\| \le \|\nabla^2 L(\tilde{\mathbf{w}})\| \|\mathbf{w}_1 - \mathbf{w}_2\|,$$

where $\tilde{\mathbf{w}}$ is between $\mathbf{w}_1$ and $\mathbf{w}_2$. Furthermore, $\|\nabla^2 L(\tilde{\mathbf{w}})\|$ is bounded:

$$\|\nabla^2 L(\tilde{\mathbf{w}})\| = C\|X^T D(\tilde{\mathbf{w}}) X\| \le C\|X^T\| \|X\|. \tag{41}$$

Note that $D(\tilde{\mathbf{w}})$ is the diagonal matrix defined in (8) though here we denote it as a function of $\mathbf{w}$. The inequality in (41) follows from that all $D(\tilde{\mathbf{w}})$'s components are smaller than one. Thus, Equation (39) holds with $\Lambda = C\|X^T\|\|X\|$. For (40), $H^k \succeq \nu I \succ 0$ because we add νI to $\nabla^2 L(\mathbf{w}^k)$ and $\nabla^2 L(\mathbf{w}^k)$ is positive semi-definite. With (39) and (40), the line search procedure terminates in finite steps.

For the asymptotic convergence, Tseng and Yun (2009) further assume that there exist positive constants $\lambda_{\min}$ and $\lambda_{\max}$ such that

$$\lambda_{\min} I \preceq H^k \preceq \lambda_{\max} I, \quad \forall k. \tag{42}$$

Since $H^k = \nabla^2 L(\mathbf{w}^k) + \nu I$, clearly we can set $\lambda_{\min} = \nu$. For the upper bound, it is sufficient to prove that the level set is bounded. See the proof in, for example, Yuan et al. (2010, Appendix A).

Following Theorem 1(e) in Tseng and Yun (2009), any limit point of $\{\mathbf{w}^k\}$ is an optimum of (1) with logistic loss.

Appendix B. Linear Convergence of newGLMNET for L1-regularized Logistic Regression

To apply the linear convergence result in Tseng and Yun (2009), we show that L1-regularized logistic regression satisfies the conditions in their Theorem 3 if the loss term $L(\mathbf{w})$ is strictly convex (and therefore $\mathbf{w}^*$ is unique).

From Appendix A, we know L1-regularized logistic regression has the following properties.

1. $\nabla L(\mathbf{w})$ is Lipschitz continuous; see (39).
2. The level set is compact, and hence the optimal solution $\mathbf{w}^*$ exists.
3. $\lambda_{\min} I \preceq H^k \preceq \lambda_{\max} I$, $\forall k$; see (42).

In addition to the above three conditions, Tseng and Yun (2009, Theorem 3) require that for all $\zeta \geq \min_{\mathbf{w}} f(\mathbf{w})$, there exists $T > 0, \varepsilon > 0$, such that

$$T\|\mathbf{d}_I(\mathbf{w})\| \geq \|\mathbf{w} - \mathbf{w}^*\|, \quad \forall \mathbf{w} \in \{\mathbf{w} \mid f(\mathbf{w}) \leq \zeta \text{ and } \|\mathbf{d}_I(\mathbf{w})\| \leq \varepsilon\}, \tag{43}$$

where $\mathbf{d}_I(\mathbf{w})$ is the solution of (13) at $\mathbf{w}$ with $H = I$ (Tseng and Yun, 2009, Assumption 2). We prove (43) by following the approach in Tseng and Yun (2009, Theorem 4).

To simplify the notation, we denote $\mathbf{d}_I \equiv \mathbf{d}_I(\mathbf{w})$. For all $\zeta > 0$, we show that there exists $T > 0$ so that (43) is satisfied for all $\mathbf{w}$ with $f(\mathbf{w}) \leq \zeta$. That is, a stronger result independent of ε is obtained. We assume $\mathbf{w}$ is in the level set $\{\mathbf{w} \mid f(\mathbf{w}) \leq \zeta\}$ in the following proof. Because $\mathbf{d}_I$ is the solution of (13) with $H = I$, by checking the optimality condition,[14] $\mathbf{d}_I$ is also an optimal solution of

$$\min_{\mathbf{d}} \quad (\nabla L(\mathbf{w}) + \mathbf{d}_I)^T \mathbf{d} + \|\mathbf{w} + \mathbf{d}\|_1.$$

Therefore,

$$(\nabla L(\mathbf{w}) + \mathbf{d}_I)^T \mathbf{d}_I + \|\mathbf{w} + \mathbf{d}_I\|_1 \leq (\nabla L(\mathbf{w}) + \mathbf{d}_I)^T (\mathbf{w}^* - \mathbf{w}) + \|\mathbf{w}^*\|_1. \tag{44}$$

Besides, because $\mathbf{w}^*$ minimizes $f(\mathbf{w})$, the following inequality holds for all $\mathbf{w}$ and $\delta \in (0,1)$.

$$\begin{aligned}
&\frac{L(\mathbf{w}^* + \delta(\mathbf{w} - \mathbf{w}^*)) - L(\mathbf{w}^*)}{\delta} + \|\mathbf{w}\|_1 - \|\mathbf{w}^*\|_1 && (45)\\
&\geq \frac{L(\mathbf{w}^* + \delta(\mathbf{w} - \mathbf{w}^*)) - L(\mathbf{w}^*) + \|\mathbf{w}^* + \delta(\mathbf{w} - \mathbf{w}^*)\|_1 - \|\mathbf{w}^*\|_1}{\delta} && (46)\\
&= \frac{f(\mathbf{w}^* + \delta(\mathbf{w} - \mathbf{w}^*)) - f(\mathbf{w}^*)}{\delta}\\
&\geq 0,
\end{aligned}$$

where (46) is from the convexity of $\|\cdot\|_1$. Take $\delta \to 0$ and replace $\mathbf{w}$ with $\mathbf{w} + \mathbf{d}_I$ in (45). We get

$$0 \leq \nabla L(\mathbf{w}^*)^T (\mathbf{w} + \mathbf{d}_I - \mathbf{w}^*) + \|\mathbf{w} + \mathbf{d}_I\|_1 - \|\mathbf{w}^*\|_1. \tag{47}$$

Adding (44) to (47) yields

$$(\nabla L(\mathbf{w}) - \nabla L(\mathbf{w}^*))^T (\mathbf{w} - \mathbf{w}^*) + \|\mathbf{d}_I\|^2 \leq (\nabla L(\mathbf{w}^*) - \nabla L(\mathbf{w}))^T \mathbf{d}_I + \mathbf{d}_I^T (\mathbf{w}^* - \mathbf{w}). \tag{48}$$

14. For example, the minimal-norm subgradients of the two objective functions are the same at $\mathbf{d}_I$.

Because the level set $\{\mathbf{w} \mid f(\mathbf{w}) \leq \zeta\}$ is compact, a strictly convex $L(\mathbf{w})$ is strongly convex in the level set. That is, there exists an $m > 0$ such that

$$(\nabla L(\mathbf{w}) - \nabla L(\mathbf{w}^*))^T(\mathbf{w} - \mathbf{w}^*) \geq m\|\mathbf{w} - \mathbf{w}^*\|^2, \quad \forall \mathbf{w} \in \{\mathbf{w} \mid f(\mathbf{w}) \leq \zeta\}.$$

Then we can relax (48) to

$$m\|\mathbf{w} - \mathbf{w}^*\|^2 \leq m\|\mathbf{w} - \mathbf{w}^*\|^2 + \|\mathbf{d}_I\|^2 \leq \Lambda\|\mathbf{w} - \mathbf{w}^*\|\|\mathbf{d}_I\| + \|\mathbf{d}_I\|\|\mathbf{w} - \mathbf{w}^*\|,$$

where Λ is the Lipschitz constant in (39). Dividing both sides by $m\|\mathbf{w} - \mathbf{w}^*\|$ generates

$$\|\mathbf{w} - \mathbf{w}^*\| \leq \frac{\Lambda + 1}{m}\|\mathbf{d}_I\|.$$

Then $T = (\Lambda + 1)/m$ satisfies condition (43). Therefore, all conditions in Tseng and Yun (2009, Theorem 3) are satisfied, so linear convergence is guaranteed.

Appendix C. Convergence of newGLMNET for L1-regularized L2-loss SVM

Similar to Appendix A, we only check the conditions required by Tseng and Yun (2009). To have the finite termination of line search, we need Equations (39) and (40), while for asymptotic convergence, we need Equation (42). Following the same explanation in Appendix A, we easily have (39) and (42). For (40), which means that $\nabla L(\mathbf{w})$ is globally Lipschitz continuous, a proof is in, for example, Mangasarian (2002, Section 3). Therefore, any limit point of $\{\mathbf{w}^k\}$ is an optimum of (1) with L2 loss by Theorem 1(e) in Tseng and Yun (2009).

Further, if the L2-loss function $L(\mathbf{w})$ is strictly convex, (43) is satisfied with L2 loss following the proof in Appendix B. Hence, $\{\mathbf{w}^k\}$ converges to the unique optimum solution at least linearly.

References

Galen Andrew and Jianfeng Gao. Scalable training of L1-regularized log-linear models. In *Proceedings of the Twenty Fourth International Conference on Machine Learning (ICML)*, 2007.

Amir Beck and Marc Teboulle. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM Journal on Imaging Sciences*, 2(1):183–202, 2009.

Chih-Chung Chang and Chih-Jen Lin. LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2:27:1–27:27, 2011. Software available at `http://www.csie.ntu.edu.tw/~cjlin/libsvm`.

Kai-Wei Chang, Cho-Jui Hsieh, and Chih-Jen Lin. Coordinate descent method for large-scale L2-loss linear SVM. *Journal of Machine Learning Research*, 9:1369–1398, 2008. URL `http://www.csie.ntu.edu.tw/~cjlin/papers/cdl2.pdf`.

Ingrid Daubechies, Michel Defrise, and Christine De Mol. An iterative thresholding algorithm for linear inverse problems with a sparsity constraint. *Communications on Pure and Applied Mathematics*, 57:1413–1457, 2004.

Laurent El Ghaoui, Vivian Viallon, and Tarek Rabbani. Safe feature elimination in sparse supervised learning. Technical report, EECS Department, University of California, Berkeley, 2010.

Rong-En Fan, Kai-Wei Chang, Cho-Jui Hsieh, Xiang-Rui Wang, and Chih-Jen Lin. LIBLINEAR: A library for large linear classification. *Journal of Machine Learning Research*, 9:1871–1874, 2008. URL http://www.csie.ntu.edu.tw/~cjlin/papers/liblinear.pdf.

Jerome H. Friedman, Trevor Hastie, and Robert Tibshirani. Regularization paths for generalized linear models via coordinate descent. *Journal of Statistical Software*, 33(1):1–22, 2010.

Alexandar Genkin, David D. Lewis, and David Madigan. Large-scale Bayesian logistic regression for text categorization. *Technometrics*, 49(3):291–304, 2007.

Cho-Jui Hsieh, Kai-Wei Chang, Chih-Jen Lin, S. Sathiya Keerthi, and Sellamanickam Sundararajan. A dual coordinate descent method for large-scale linear SVM. In *Proceedings of the Twenty Fifth International Conference on Machine Learning (ICML)*, 2008. URL http://www.csie.ntu.edu.tw/~cjlin/papers/cddual.pdf.

Cho-Jui Hsieh, Matyas A. Sustik, Pradeep Ravikumar, and Inderjit S. Dhillon. Sparse inverse covariance matrix estimation using quadratic approximation. In *Advances in Neural Information Processing Systems (NIPS) 24*, 2011.

Fang-Lan Huang, Cho-Jui Hsieh, Kai-Wei Chang, and Chih-Jen Lin. Iterative scaling and coordinate descent methods for maximum entropy. *Journal of Machine Learning Research*, 11:815–848, 2010. URL http://www.csie.ntu.edu.tw/~cjlin/papers/maxent_journal.pdf.

Thorsten Joachims. Making large-scale SVM learning practical. In Bernhard Schölkopf, Christopher J. C. Burges, and Alexander J. Smola, editors, *Advances in Kernel Methods – Support Vector Learning*, pages 169–184, Cambridge, MA, 1998. MIT Press.

Kwangmoo Koh, Seung-Jean Kim, and Stephen Boyd. An interior-point method for large-scale l1-regularized logistic regression. *Journal of Machine Learning Research*, 8:1519–1555, 2007. URL http://www.stanford.edu/~boyd/l1_logistic_reg.html.

Su-In Lee, Honglak Lee, Pieter Abbeel, and Andrew Y. Ng. Efficient l1 regularized logistic regression. In *Proceedings of the Twenty-first National Conference on Artificial Intelligence (AAAI-06)*, pages 1–9, Boston, MA, USA, July 2006.

Chih-Jen Lin and Jorge J. Moré. Newton's method for large-scale bound constrained problems. *SIAM Journal on Optimization*, 9:1100–1127, 1999.

Jun Liu, Jianhui Chen, and Jieping Ye. Large-scale sparse logistic regression. In *Proceedings of The 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 547–556, 2009.

Olvi L. Mangasarian. A finite Newton method for classification. *Optimization Methods and Software*, 17(5):913–929, 2002.

Mark Schmidt, Glenn Fung, and Romer Rosales. Optimization methods for l1-regularization. Technical Report TR-2009-19, University of British Columbia, 2009.

Nicol N. Schraudolph. A fast, compact approximation of the exponential function. *Neural Computation*, 11:853–862, 1999.

Shai Shalev-Shwartz and Ambuj Tewari. Stochastic methods for l1 regularized loss minimization. In *Proceedings of the Twenty Sixth International Conference on Machine Learning (ICML)*, 2009.

Shirish Krishnaj Shevade and S. Sathiya Keerthi. A simple and efficient algorithm for gene selection using sparse logistic regression. *Bioinformatics*, 19(17):2246–2253, 2003.

Jianing Shi, Wotao Yin, Stanley Osher, and Paul Sajda. A fast hybrid algorithm for large scale l1-regularized logistic regression. *Journal of Machine Learning Research*, 11:713–741, 2010.

Choon Hui Teo, S.V.N. Vishwanathan, Alex Smola, and Quoc V. Le. Bundle methods for regularized risk minimization. *Journal of Machine Learning Research*, 11:311–365, 2010.

Robert Tibshirani, Jacob Bien, Jerome Friedman, Trevor Hastie, Noah Simon, Jonatha Taylor, and Ryan J. Tibshirani. Strong rules for discarding predictors in lasso-type problems. *Journal of Royal Statistical Society: Series B*, 2011.

Paul Tseng and Sangwoon Yun. A coordinate gradient descent method for nonsmooth separable minimization. *Mathematical Programming*, 117:387–423, 2009.

Guo-Xun Yuan, Kai-Wei Chang, Cho-Jui Hsieh, and Chih-Jen Lin. A comparison of optimization methods and software for large-scale l1-regularized linear classification. *Journal of Machine Learning Research*, 11:3183–3234, 2010. URL `http://www.csie.ntu.edu.tw/~cjlin/papers/l1.pdf`.

Guo-Xun Yuan, Chia-Hua Ho, and Chih-Jen Lin. An improved GLMNET for l1-regularized logistic regression. In *Proceedings of the Seventeenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 33–41, 2011.

Journal of Machine Learning Research 13 (2012) 2031-2061 Submitted 9/11; Revised 2/12; Published 6/12

EP-GIG Priors and Applications in Bayesian Sparse Learning

Zhihua Zhang ZHZHANG@ZJU.EDU.CN
Shusen Wang WSSATZJU@GMAIL.COM
Dehua Liu DEHUALIU0427@GMAIL.COM
College of Computer Science and Technology
Zhejiang University
Hangzhou, Zhejiang 310027, China

Michael I. Jordan JORDAN@CS.BERKELEY.EDU
Computer Science Division and Department of Statistics
University of California
Berkeley, CA 94720-1776, USA

Editor: Neil Lawrence

Abstract

In this paper we propose a novel framework for the construction of sparsity-inducing priors. In particular, we define such priors as a mixture of exponential power distributions with a generalized inverse Gaussian density (EP-GIG). EP-GIG is a variant of generalized hyperbolic distributions, and the special cases include Gaussian scale mixtures and Laplace scale mixtures. Furthermore, Laplace scale mixtures can subserve a Bayesian framework for sparse learning with nonconvex penalization. The densities of EP-GIG can be explicitly expressed. Moreover, the corresponding posterior distribution also follows a generalized inverse Gaussian distribution. We exploit these properties to develop EM algorithms for sparse empirical Bayesian learning. We also show that these algorithms bear an interesting resemblance to iteratively reweighted ℓ_2 or ℓ_1 methods. Finally, we present two extensions for grouped variable selection and logistic regression.

Keywords: sparsity priors, scale mixtures of exponential power distributions, generalized inverse Gaussian distributions, expectation-maximization algorithms, iteratively reweighted minimization methods

1. Introduction

In this paper we are concerned with sparse supervised learning problems over a training data set $\mathcal{X} = \{(\mathbf{x}_i, y_i)\}_{i=1}^n$. The point of departure for our work is the traditional formulation of supervised learning as a regularized optimization problem:

$$\min_{\mathbf{b}} \Big\{ L(\mathbf{b}; \mathcal{X}) + P_\lambda(\mathbf{b}) \Big\},$$

where $\mathbf{b}$ denotes the model parameter vector, $L(\cdot)$ a loss function that penalizes data misfit, $P_\lambda(\cdot)$ a regularization term penalizing model complexity, and $\lambda > 0$ a tuning parameter balancing the relative significance of the loss function and the penalty.

Variable selection is a fundamental problem in the high-dimensional learning setting, and is closely tied to the notion that the data-generating mechanism can be described using a sparse representation. In supervised learning scenarios, the problem is to obtain sparse estimates for the regression vector $\mathbf{b}$. Given that it is NP-hard to use the ℓ_0 penalty (that is, the number of the

nonzero elements of $\mathbf{b}$) (Weston et al., 2003), attention has focused on use of the ℓ_1 penalty (Tibshirani, 1996). But in addition a number of studies have emphasized the advantages of nonconvex penalties—such as the bridge penalty and the log-penalty—for achieving sparsity (Fu, 1998; Fan and Li, 2001; Mazumder et al., 2011).

The regularized optimization problem can be cast into a maximum *a posteriori* (MAP) framework. This is done by taking a Bayesian decision-theoretic approach in which the loss function $L(\mathbf{b};\mathcal{X})$ is based on the conditional likelihood of the response y_i and the penalty $P_\lambda(\mathbf{b})$ is associated with a prior distribution for $\mathbf{b}$. For example, the least-squares loss function is associated with a Gaussian likelihood, while there exists duality between the ℓ_1 penalty and the Laplace prior.

The MAP framework provides us with Bayesian underpinnings for the sparse estimation problem. This has led to Bayesian versions of the lasso, which are based on expressing the Laplace prior as a scale-mixture of a Gaussian distribution and an exponential density (Andrews and Mallows, 1974; West, 1987). Figueiredo (2003) and Kiiveri (2008) presented a Bayesian lasso based on the Expectation-Maximization (EM) algorithm. Caron and Doucet (2008) considered EM-based estimation with normal-gamma or normal-inverse-gaussian priors. In recent work, Polson and Scott (2011) proposed using generalized hyperbolic distributions, variance-mean mixtures of Gaussians with generalized inverse Gaussian densities, devising EM algorithms via data augmentation methodology. Lee et al. (2010) referred to such methods as "quasi-Bayesian." Related empirical-Bayesian sparse learning methods have been developed by Tipping (2001).

Recently, Park and Casella (2008) and Hans (2009) proposed full Bayesian lasso models based on Gibbs sampling. Further work by Griffin and Brown (2010a) involved the use of a family of normal-gamma priors as a generalization of the Bayesian lasso. This prior has been also used by Archambeau and Bach (2009) to develop sparse probabilistic projections. In the work of Carvalho et al. (2010), the authors proposed horseshoe priors which are a mixture of normal distributions and a half-Cauchy density on the positive reals with a scale parameter. Kyung et al. (2010) conducted in-depth performance analysis of Bayesian lassos.

There has also been work on nonconvex penalties within a Bayesian framework. Zou and Li (2008) derived their local linear approximation (LLA) algorithm by combining the EM algorithm with an inverse Laplace transformation. In particular, they showed that the bridge penalty can be obtained by mixing the Laplace distribution with a stable distribution. Other authors have shown that the prior induced from the log-penalty has an interpretation as a scale mixture of Laplace distributions with an inverse gamma density (Cevher, 2009; Garrigues and Olshausen, 2010; Lee et al., 2010; Armagan et al., 2011). Additionally, Griffin and Brown (2010b) devised a family of normal-exponential-gamma priors for a Bayesian adaptive lasso (Zou, 2006). Polson and Scott (2010, 2012) provided a unifying framework for the construction of sparsity priors using Lévy processes.

In this paper we develop a novel framework for constructing sparsity-inducing priors. Generalized inverse Gaussian (GIG) distributions (Jørgensen, 1982) are conjugate with respect to an exponential power (EP) distribution (Box and Tiao, 1992)—an extension of Gaussian and Laplace distributions. Accordingly, we propose a family of distributions that we refer to as *EP-GIG*. In particular, we define EP-GIG distributions as scale mixtures of EP distributions with a GIG density, and derive their explicit densities. EP-GIG distributions can be regarded as a variant of generalized hyperbolic distributions, and include Gaussian scale mixtures and Laplacian scale mixtures as special cases. The Gaussian scale mixture is a class of generalized hyperbolic distributions (Polson and Scott, 2011) and its special cases include normal-gamma distributions (Griffin and Brown, 2010a)

as well as the Laplacian distribution. The generalized double Pareto distribution (Cevher, 2009; Armagan et al., 2011; Lee et al., 2010) and the bridge distribution inducing the $\ell_{1/2}$ bridge penalty (Zou and Li, 2008) are special cases of Laplacian scale mixtures. In Appendix B, we devise a set of new EP-GIG priors.

Since GIG priors are conjugate with respect to EP distributions, it is feasible to apply EP-GIG to Bayesian sparse learning. Although it has been illustrated that fully Bayesian sparse learning methods based on Markov chain Monte Carlo sampling work well, our main focus is on a quasi-Bayesian approach. Our goal is to explore the relationship between MAP estimators and classical regularized estimators. In particular, using the fact that EP-GIG distributions are scale mixtures of exponential power distributions, we devise EM algorithms for finding a sparse MAP estimate of $\mathbf{b}$.

When we set the exponential power distribution to be the Gaussian distribution, the resulting EM algorithm is closely related to the iteratively reweighted ℓ_2 minimization methods in Daubechies et al. (2010); Chartrand and Yin (2008) and Wipf and Nagarajan (2010). When we employ the Laplace distribution as a special exponential power distribution, we obtain an EM algorithm which is identical to the iteratively reweighted ℓ_1 minimization method in Candès et al. (2008).

We also develop hierarchical Bayesian approaches for grouped variable selection (Yuan and Lin, 2007) and penalized logistic regression by using EP-GIG priors. We apply our proposed EP-GIG priors in Appendix B to conduct experimental analysis. The experimental results validate that the proposed EP-GIG priors which induce nonconvex penalties are potentially feasible and effective in sparsity modeling. Finally, we would like to highlight that our work offers several important theoretical insights as follows.

1. Theorem 2 establishes a limiting relationship of EP-GIG distributions with the corresponding EP distributions, an extension of the classical limiting relationship between the t-distribution and Gaussian distribution as the degree of freedom approaches infinity. Theorem 5 proves that an exponential power distribution of order $q/2$ ($q > 0$) can be represented a scale mixture of exponential power distributions of order q with a gamma mixing density.

2. The first part of Theorem 6 shows that GIG is conjugate with respect to EP, while the second part then offers theoretical support for relating EM algorithms with iteratively reweighted minimization methods under our framework.

3. Theorem 7 shows that the negative log EP-GIG can induce a class of sparsity penalties, in particular an interesting class of nonconvex penalties. Theorem 9 gives convergence analysis for the EM algorithm. Finally, Theorem 10 establishes the oracle properties of the sparse estimator based on Laplace scale mixture priors.

The paper is organized as follows. A brief review of exponential power distributions and generalized inverse Gaussian distributions is given in Section 2. Section 3 presents EP-GIG distributions and some of their properties, Section 4 develops our EM algorithm for Bayesian sparse learning, and Section 5 discusses the relationship between the EM and iteratively reweighted minimization methods. In Section 6 we conduct our experimental evaluations. Finally, we conclude our work in Section 7, defer all proofs to Appendix A, and provide several new sparsity priors in Appendix B.

2. Preliminaries

Before presenting EP-GIG priors for sparse modeling of regression vector **b**, we review the exponential power (EP) and generalized inverse Gaussian (GIG) distributions.

2.1 Exponential Power Distributions

A univariate random variable $b \in \mathbb{R}$ is said to follow an EP distribution if the density is specified by

$$p(b) = \frac{\eta^{-1/q}}{2^{\frac{q+1}{q}}\Gamma(\frac{q+1}{q})}\exp(-\frac{1}{2\eta}|b-u|^q) = \frac{q}{2}\frac{(2\eta)^{-\frac{1}{q}}}{\Gamma(\frac{1}{q})}\exp(-\frac{1}{2\eta}|b-u|^q),$$

with $\eta > 0$. In the literature (Box and Tiao, 1992), it is typically assumed that $q \geq 1$. However, it is possible to relax this assumption into $q > 0$, which will be useful for our purposes. Moreover, we will only consider the setting that $u = 0$.

The distribution is denoted by $\mathsf{EP}(b|u,\eta,q)$. There are two classical special cases: the Gaussian distribution arises when $q = 2$ (denoted $N(b|u,\eta)$) and the Laplace distribution arises when $q = 1$ (denoted $L(b|u,\eta)$). As for the case that $q < 1$, the corresponding density induces a bridge penalty for b. We thus refer to it as the bridge distribution.

2.2 Generalized Inverse Gaussian Distributions

We first let $G(\eta|\tau,\theta)$ denote the gamma distribution whose density is

$$p(\eta) = \frac{\theta^\tau}{\Gamma(\tau)}\eta^{\tau-1}\exp(-\theta\eta), \quad \tau,\theta > 0,$$

and $\mathsf{IG}(\eta|\tau,\theta)$ denote the inverse gamma distribution whose density is

$$p(\eta) = \frac{\theta^\tau}{\Gamma(\tau)}\eta^{-(1+\tau)}\exp(-\theta\eta^{-1}), \quad \tau,\theta > 0.$$

We now consider the GIG distribution. The density of the GIG distribution is defined as

$$p(\eta) = \frac{(\alpha/\beta)^{\gamma/2}}{2K_\gamma(\sqrt{\alpha\beta})}\eta^{\gamma-1}\exp(-(\alpha\eta+\beta\eta^{-1})/2),\ \eta > 0,$$

where $K_\gamma(\cdot)$ represents the modified Bessel function of the second kind with the index γ. We denote this distribution by $\mathsf{GIG}(\eta|\gamma,\beta,\alpha)$. It is well known that its special cases include the gamma distribution $G(\eta|\gamma,\alpha/2)$ when $\beta = 0$ and $\gamma > 0$, the inverse gamma distribution $\mathsf{IG}(\eta|-\gamma,\beta/2)$ when $\alpha = 0$ and $\gamma < 0$, the inverse Gaussian distribution when $\gamma = -1/2$, and the hyperbolic distribution when $\gamma = 0$. Please refer to Jørgensen (1982) for details.

Note in particular that the pdf of the inverse Gaussian $\mathsf{GIG}(\eta|-1/2,\beta,\alpha)$ is

$$p(\eta) = \Big(\frac{\beta}{2\pi}\Big)^{1/2}\exp(\sqrt{\alpha\beta})\eta^{-\frac{3}{2}}\exp(-(\alpha\eta+\beta\eta^{-1})/2),\ \beta > 0,$$

and the pdf of $\mathsf{GIG}(\eta|1/2,\beta,\alpha)$ is

$$p(\eta) = \Big(\frac{\alpha}{2\pi}\Big)^{1/2}\exp(\sqrt{\alpha\beta})\eta^{-\frac{1}{2}}\exp(-(\alpha\eta+\beta\eta^{-1})/2),\ \alpha > 0.$$

Note moreover that $\mathsf{GIG}(\eta|-1/2,\beta,0)$ and $\mathsf{GIG}(\eta|1/2,0,\alpha)$ degenerate to $\mathsf{IG}(\eta|1/2,\beta/2)$ and $G(\eta|1/2,\alpha/2)$, respectively.

We now present an alternative expression for the GIG density that is interesting. Let $\psi=\sqrt{\alpha\beta}$ and $\phi=\sqrt{\alpha/\beta}$. We can rewrite the density of $\mathsf{GIG}(\eta|\gamma,\beta,\alpha)$ as

$$p(\eta)=\frac{\phi^{\gamma}}{2K_{\gamma}(\psi)}\eta^{\gamma-1}\exp(-\psi(\phi\eta+(\phi\eta)^{-1})/2),\ \eta>0. \tag{1}$$

Let us consider that the case $\gamma=0$. Furthermore, letting $\psi\to 0$, we can see that $p(\eta)\propto 1/\eta$, an improper prior. Note that this improper prior can regarded as the Jeffreys prior because the Fisher information of $\mathsf{EP}(b|0,\eta)$ with respect to η is η^{-2}/q. Finally, we present some useful limiting properties of GIG distributions in Appendix A.2.

3. EP-GIG Distributions

We now develop a family of distributions by mixing the exponential power $\mathsf{EP}(b|0,\eta,q)$ with the generalized inverse Gaussian $\mathsf{GIG}(\eta|\gamma,\beta,\alpha)$. The marginal density of b is thus defined by

$$p(b)=\int_0^{+\infty}\mathsf{EP}(b|0,\eta,q)\mathsf{GIG}(\eta|\gamma,\beta,\alpha)d\eta.$$

We refer to this distribution as the EP-GIG and denote it by $\mathsf{EGIG}(b|\alpha,\beta,\gamma,q)$. The density can be obtained via direct calculations. We have:

Theorem 1 *Let $b\sim\mathsf{EGIG}(b|\alpha,\beta,\gamma,q)$. Then its density is*

$$p(b)=\frac{K_{\frac{\gamma q-1}{q}}(\sqrt{\alpha(\beta+|b|^q)})}{2^{\frac{q+1}{q}}\Gamma(\frac{q+1}{q})K_{\gamma}(\sqrt{\alpha\beta})}\frac{\alpha^{1/(2q)}}{\beta^{\gamma/2}}[\beta+|b|^q]^{(\gamma q-1)/(2q)}. \tag{2}$$

The following theorem establishes an important relationship between an EP-GIG distribution and the underlying EP distribution. It is an extension of the classical relationship of a t-distribution with the Gaussian distribution. The proof can be directly obtained from Proposition 19 in Appendix A.2.

Theorem 2 *We have the following asymptotic relationships:*

(1) $\lim_{\gamma\to+\infty}\mathsf{EGIG}(b|\gamma\alpha,\beta,\gamma,q)=\mathsf{EP}(b|0,2/\alpha,q)$;

(2) $\lim_{\gamma\to-\infty}\mathsf{EGIG}(b|\alpha,-\gamma\beta,\gamma,q)=\mathsf{EP}(b|0,\beta/2,q)$.

(3) $\lim_{\psi\to+\infty}\mathsf{EGIG}(b|\alpha,\beta,\gamma,q)=\mathsf{EP}(b|0,\phi,q)$ *where* $\psi=\sqrt{\alpha\beta}$ *and* $\phi=\sqrt{\alpha/\beta}\in(0,\infty)$.

EP-GIG distributions can be regarded as variants of generalized hyperbolic distributions (Jørgensen, 1982), because when $q=2$ EP-GIG distributions are generalized hyperbolic distributions—a class of Gaussian scale mixtures. However, EP-GIG becomes a class of Laplace scale mixtures when $q=1$. Note that when $0<q<2$ an EP distribution is a class of Gaussian scale mixtures (West, 1987; Lange and Sinsheimer, 1993), which implies that EP-GIG can also be represented as

a class of Gaussian scale mixtures. However, the difficulty with such a representation is that the corresponding mixing prior is usually not analytically available.

In Appendix B we present several new concrete EP-GIG distributions, obtained from particular settings of γ and q. We now consider the two special cases in which the mixing density is either a gamma distribution or an inverse gamma distribution. This yields two special EP-GIG distributions: exponential power-gamma distributions and exponential power-inverse gamma distributions.

3.1 Generalized t Distributions

We first consider an important family of EP-GIG distributions which are scale mixtures of exponential power $\mathsf{EP}(b|u,\eta,q)$ with inverse gamma $\mathsf{IG}(\eta|\tau/2,\tau/(2\lambda))$. Following the terminology of Lee et al. (2010), we refer them as *generalized t distributions* and denote them by $\mathsf{GT}(b|u,\tau/\lambda,\tau/2,q)$. Specifically, the density of the generalized t is

$$p(b)=\int \mathsf{EP}(b|u,\eta,q)\mathsf{IG}(\eta|\tau/2,\tau/(2\lambda))d\eta=\frac{q}{2}\frac{\Gamma(\frac{\tau}{2}+\frac{1}{q})}{\Gamma(\frac{\tau}{2})\Gamma(\frac{1}{q})}\Big(\frac{\lambda}{\tau}\Big)^{\frac{1}{q}}\Big(1+\frac{\lambda}{\tau}|b-u|^q\Big)^{-(\frac{\tau}{2}+\frac{1}{q})}, \quad (3)$$

where $\tau>0$, $\lambda>0$ and $q>0$. Clearly, when $q=2$ the generalized t distribution becomes to a t-distribution. Moreover, when $\tau=1$, it is the Cauchy distribution.

On the other hand, when $q=1$, Cevher (2009) and Armagan et al. (2011) called the resulting distributions *generalized double Pareto distributions* (GDP). The densities are given as follows:

$$p(b)=\int_0^\infty L(b|0,\eta)\mathsf{IG}(\eta|\tau/2,\tau/(2\lambda))d\eta=\frac{\lambda}{4}\Big(1+\frac{\lambda|b|}{\tau}\Big)^{-(\tau/2+1)}, \quad \lambda>0,\tau>0.$$

Furthermore, consider $\tau=1$, such that $\eta\sim\mathsf{IG}(\eta|1/2,1/(2\lambda))$. We obtain

$$p(b)=\frac{\lambda}{4}(1+\lambda|b|)^{-3/2}.$$

It is well known that the limit of the t-distribution as $\tau\to\infty$ is the normal distribution. We find that we are able to extend this property to the generalized t distribution. In particular, we have the following theorem, which is a corollary of the first part of Theorem 2.

Corollary 3 *Let the generalized t distribution be defined in (3). Then, for $\lambda>0$ and $q>0$,*

$$\lim_{\tau\to\infty}\mathsf{GT}(b|u,\tau/\lambda,\tau/2,q)=\mathsf{EP}(b|u,1/\lambda,q).$$

Thus, as a special case of Corollary 3 for $q=1$, we have

$$\lim_{\tau\to\infty}\mathsf{GT}(b|u,\tau/\lambda,\tau/2,1)=L(b|u,1/\lambda).$$

3.2 Exponential Power-Gamma Distributions

The density of the exponential power-gamma distribution is defined by

$$p(b|\gamma,\alpha)=\int_0^\infty \mathsf{EP}(b|0,\eta,q)G(\eta|\gamma,\alpha/2)d\eta=\frac{\alpha^{\frac{q\gamma+1}{2q}}|b|^{\frac{q\gamma-1}{2}}}{2^{\frac{q\gamma+1}{q}}\Gamma(\frac{q+1}{q})\Gamma(\gamma)}K_{\gamma-\frac{1}{q}}(\sqrt{\alpha|b|^q}),\gamma,\alpha>0.$$

We denote the distribution by $\mathsf{EG}(b|\alpha,\gamma,q)$. The density of the normal-gamma distribution (Griffin and Brown, 2010a) is

$$p(b|\gamma,\alpha)=\int_0^\infty N(b|0,\eta)G(\eta|\gamma,\alpha/2)d\eta=\frac{\alpha^{\frac{2\gamma+1}{4}}|b|^{\gamma-\frac{1}{2}}}{2^{\gamma-\frac{1}{2}}\sqrt{\pi}\Gamma(\gamma)}K_{\gamma-\frac{1}{2}}(\sqrt{\alpha}|b|),\quad \gamma,\alpha>0.$$

As an application of the second part of Theorem 2 in this case, we can obtain the following theorem.

Corollary 4 *Let* $\mathsf{EG}(b|\lambda\gamma,\gamma/2,q)=\int_0^\infty \mathsf{EP}(b|0,\eta,q)G(\eta|\gamma/2,\lambda\gamma/2)d\eta$ *with* $\lambda>0$. *Then*

$$\lim_{\gamma\to\infty}\mathsf{EG}(b|\lambda\gamma,\gamma/2,q)=\mathsf{EP}(b|0,1/\lambda,q).$$

It is easily seen that when we let $\gamma=1$, the normal-gamma distribution degenerates to the Laplace distribution $L(b|0,\alpha^{-1/2}/2)$. In addition, when $q=1$ and $\gamma=3/2$, which implies that $[b|\eta]\sim L(b|0,\eta)$ and $\eta\sim G(\eta|3/2,\alpha/2)$, we have

$$p(b|\alpha)=\frac{\alpha}{4}\exp(-\sqrt{\alpha|b|})=\int_0^{+\infty}L(b|0,\eta)G(\eta|3/2,\alpha/2)d\eta. \tag{4}$$

Obviously, the current exponential power-gamma distribution is identical to exponential power distribution $\mathsf{EP}(b|0,\alpha^{-1/2}/2,1/2)$, a bridge distribution with $q=1/2$. Interestingly, we can extend this relationship between the Gaussian and Laplace as well as between the Laplace and $1/2$-bridge to the general case. That is,

Theorem 5 *Let* $\gamma=\frac{1}{2}+\frac{1}{q}$. *Then,*

$$\mathsf{EP}(b|0,\alpha^{-1/2}/2,q/2)=\frac{q\alpha^{1/q}}{4\Gamma(2/q)}\exp(-\sqrt{\alpha|b|^q})=\int_0^{+\infty}\mathsf{EP}(b|0,\eta,q)G(\eta|\gamma,\alpha/2)d\eta.$$

This theorem implies that a $q/2-$bridge distribution can be represented as a scale mixture of $q-$bridge distributions. A class of important settings are $q=2^{1-m}$ and $\gamma=\frac{1}{2}+\frac{1}{q}=\frac{1+2^m}{2}$ where m is any nonnegative integer.

3.3 Conditional Priors, Marginal Priors and Posteriors

We now study the posterior distribution of η conditioning on b. It is immediate that the posterior distribution follows $\mathsf{GIG}(\eta|(\gamma q-1)/q,(\beta+|b|^q),\alpha)$. This implies that GIG distributions are conjugate with respect to the EP distribution. We note that in the cases $\gamma=1/2$ and $q=1$, as well as $\gamma=0$ and $q=2$, the posterior distribution is $\mathsf{GIG}(\eta|-1/2,(\beta+|b|^q),\alpha)$. In the cases $\gamma=3/2$ and $q=1$, as well as $\gamma=1$ and $q=2$, the posterior distribution is $\mathsf{GIG}(\eta|1/2,(\beta+|b|^q),\alpha)$. When $\gamma=-1/2$ and $q=1$, or $\gamma=-1$ and $q=2$, the posterior distribution is $\mathsf{GIG}(\eta|-3/2,(\beta+|b|^q),\alpha)$.

Additionally, we have the following theorem.

Theorem 6 *Suppose that* $b|\eta\sim\mathsf{EP}(b|0,\eta,q)$ *and* $\eta\sim\mathsf{GIG}(\eta|\gamma,\beta,\alpha)$. *Then*

(i) $b\sim\mathsf{EGIG}(b|\alpha,\beta,\gamma,q)$ *and* $\eta|b\sim\mathsf{GIG}(\eta|(\gamma q-1)/q,(\beta+|b|^q),\alpha)$.

(ii) $\frac{\partial-\log p(b)}{\partial|b|^q}=\frac{1}{2}E(\eta^{-1}|b)=\frac{1}{2}\int\eta^{-1}p(\eta|b)d\eta$.

When $-\log p(b)$ is used as a penalty in supervised sparse learning, iteratively reweighted ℓ_1 or ℓ_2 methods are generally used for solving the resulting optimization problem. We will see that Theorem 6 implies a relationship between an iteratively reweighted method and an EM algorithm, which is presented in Section 4.

3.4 Duality between Priors and Penalties

Since there is duality between a prior and a penalty, we are able to construct a penalty from $p(b)$; in particular, $-\log p(b)$ corresponds to a penalty. For example, let $p(b)$ be defined as in (7) or (8) (see Appendix B). It is then easily checked that $-\log p(b)$ is concave in $|b|$. Moreover, if $p(b)$ is given in (4), then $-\log p(b)$ induces the $\ell_{1/2}$ penalty $|b|^{1/2}$. In fact, we have the following theorem.

Theorem 7 *Let $p(b)$ be the EP-GIG density given in (2). If $-\log p(b)$ is regarded as a function of $|b^q|$, then $-\frac{d\log(p(b))}{d|b|^q}$ is completely monotone on $(0,\infty)$. Furthermore, when $0<q\leq 1$, $-\log(p(b))$ is concave in $|b|$ on $(0,\infty)$; namely, $-\log(p(b))$ defines a class of nonconvex penalties for b.*

Here a function $\phi(z)$ on $(0,\infty)$ is said to be completely monotone (Feller, 1971) if it possesses derivatives $\phi^{(n)}$ of all orders and

$$(-1)^n\phi^{(n)}(z)\geq 0,\ z>0.$$

Theorem 7 implies that the first-order and second-order derivatives of $-\log(p(b))$ with respect to $|b|^q$ are nonnegative and nonpositive, respectively. Thus, $-\log(p(b))$ is concave and nondecreasing in $|b|^q$ on $(0,\infty)$. Additionally, $|b|^q$ for $0<q\leq 1$ is concave in $|b|$ on $(0,\infty)$. Consequently, when $0<q\leq 1$, $-\log(p(b))$ is concave in $|b|$ on $(0,\infty)$. In other words, $-\log(p(b))$ with $0<q\leq 1$ induces a nonconvex penalty for b.

Figure 1 depicts several penalties graphically; these are obtained from the special priors in Appendix B. It is readily seen that the fist three penalty functions are concave in $|b|$ on $(0,\infty)$. In Figure 2, we also illustrate the penalties induced from the $1/2$-bridge scale mixture priors (see Examples 7 and 8 in in Appendix B), generalized t priors and EP-Gamma priors. Again, we see that the two penalties induced from the $1/2$-bridge mixture priors are concave in $|b|$ on $(0,\infty)$. This agrees with Theorem 7.

4. Quasi-Bayesian Sparse Learning Methods

In this section we apply EP-GIG priors to quasi-Bayesian sparse learning. Suppose we are given a set of training data $\{(\mathbf{x}_i,y_i):i=1,\ldots,n\}$, where the $\mathbf{x}_i\in\mathbb{R}^p$ are the input vectors and the y_i are the corresponding responses. Moreover, we assume that $\sum_{i=1}^n\mathbf{x}_i=\mathbf{0}$ and $\sum_{i=1}^n y_i=0$. We now consider the following linear regression model:

$$\mathbf{y}=\mathbf{X}\mathbf{b}+\varepsilon,$$

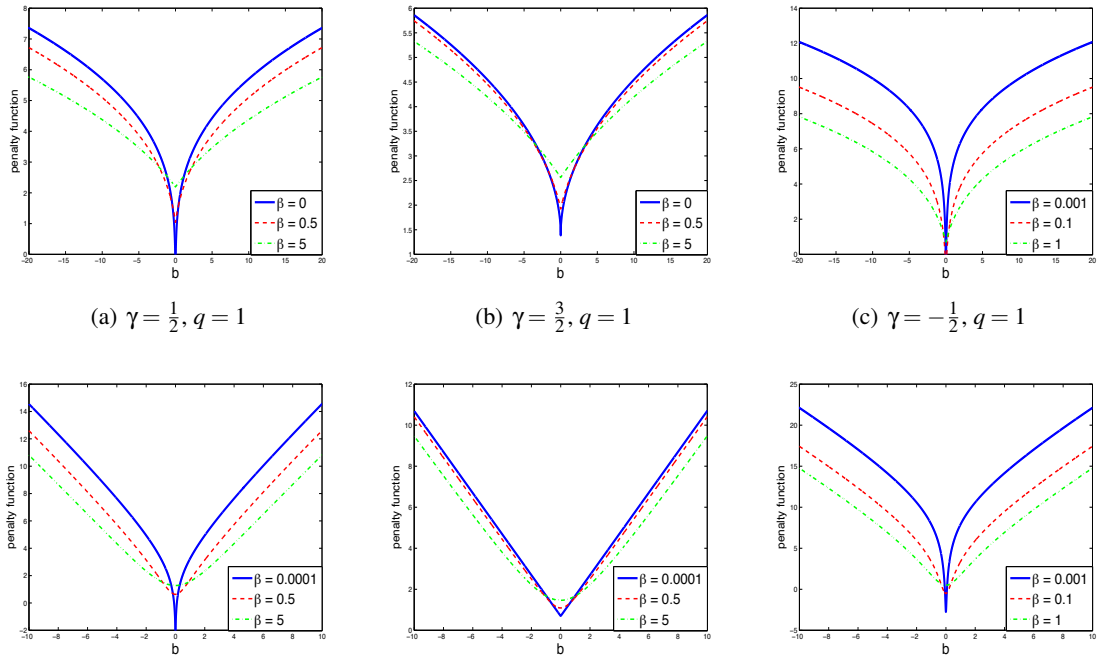
where $\mathbf{y}=(y_1,\ldots,y_n)^T$ is the $n\times 1$ response vector, $\mathbf{X}=[\mathbf{x}_1,\ldots,\mathbf{x}_n]^T$ is the $n\times p$ input matrix, and ε is a Gaussian error vector $N(\varepsilon|\mathbf{0},\sigma\mathbf{I}_n)$. We aim to estimate the vector of regression coefficients $\mathbf{b}=(b_1,\ldots,b_p)^T$ under a MAP framework.

4.1 Bayesian Sparse Regression

We place an EP-GIG prior on each of the elements of $\mathbf{b}$. That is,

$$p(\mathbf{b}|\sigma)=\prod_{j=1}^p \mathsf{EGIG}(b_j|\sigma^{-1}\alpha,\sigma\beta,\gamma,q).$$

Using the property that the EP-GIG distribution is a scale mixture of exponential power distributions, we devise an EM algorithm for the MAP estimation of $\mathbf{b}$. For this purpose, we define a



(a) $\gamma = \frac{1}{2}, q = 1$ (b) $\gamma = \frac{3}{2}, q = 1$ (c) $\gamma = -\frac{1}{2}, q = 1$

(d) $\gamma = 0, q = 2$ (e) $\gamma = 1, q = 2$ (f) $\gamma = -1, q = 2$

Figure 1: Penalty functions induced from exponential power-generalized inverse gamma (EP-GIG) priors in which $\alpha = 1$.

hierarchical model:

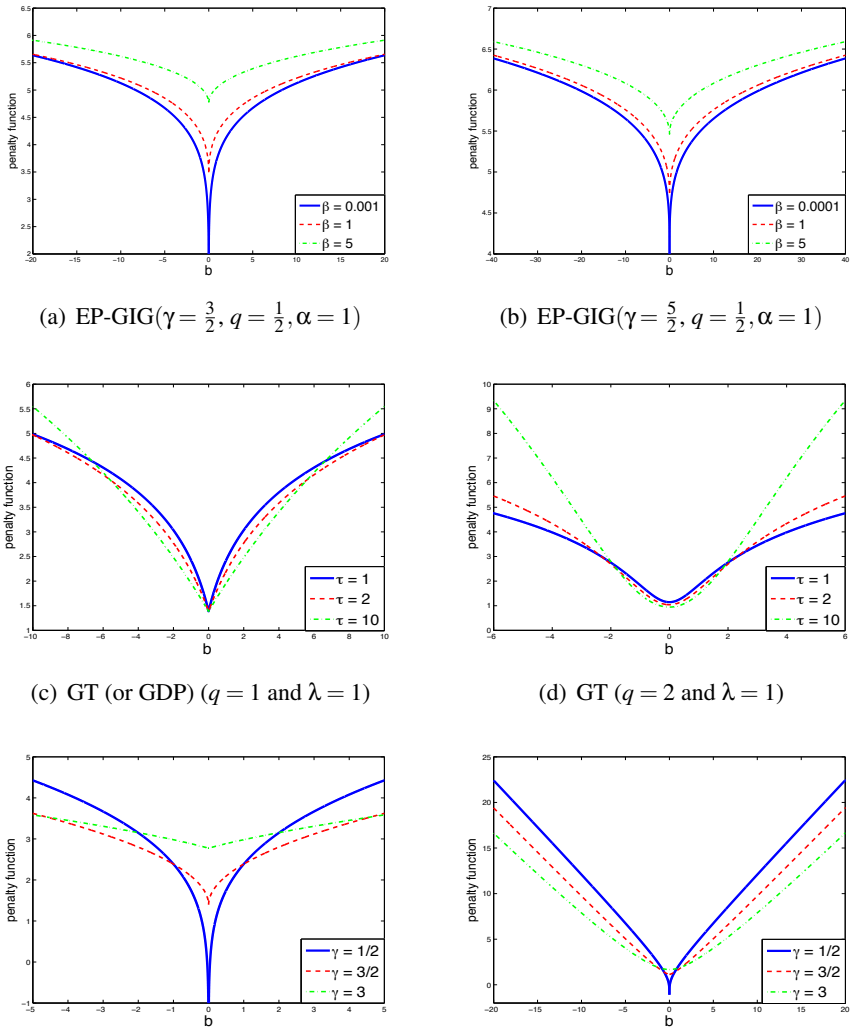
$$
\begin{aligned}
[\mathbf{y}|\mathbf{b},\sigma] &\sim N(\mathbf{y}|\mathbf{X}\mathbf{b},\sigma\mathbf{I}_n),\\
[b_j|\eta_j,\sigma] &\overset{ind}{\sim} \mathsf{EP}(b_j|0,\sigma\eta_j,q),\\
[\eta_j|\gamma,\beta,\alpha] &\overset{iid}{\sim} \mathsf{GIG}(\eta_j|\gamma,\beta,\alpha),\\
p(\sigma) &= \text{Constant}.
\end{aligned}
$$

According to Section 3.3, we have

$$[\eta_j|b_j,\sigma,\alpha,\beta,\gamma] \sim \mathsf{GIG}\big(\eta_j\big|(\gamma q-1)/q,\ \beta+\sigma^{-1}|b_j|^q,\ \alpha\big).$$

Given the tth estimates $(\mathbf{b}^{(t)},\sigma^{(t)})$ of $(\mathbf{b},\sigma)$, the E-step of EM calculates

$$
\begin{aligned}
Q(\mathbf{b},\sigma|\mathbf{b}^{(t)},\sigma^{(t)}) &\triangleq \log p(\mathbf{y}|\mathbf{b},\sigma) + \sum_{j=1}^{p}\int \log p[b_j|\eta_j,\sigma]p(\eta_j|b_j^{(t)},\sigma^{(t)},\alpha,\beta,\gamma)d\eta_j\\
&\propto -\frac{n}{2}\log\sigma - \frac{1}{2\sigma}(\mathbf{y}-\mathbf{X}\mathbf{b})^T(\mathbf{y}-\mathbf{X}\mathbf{b}) - \frac{p}{q}\log\sigma\\
&\quad - \frac{1}{2\sigma}\sum_{j=1}^{p}|b_j|^q\int \eta_j^{-1}p(\eta_j|b_j^{(t)},\sigma^{(t)},\alpha,\beta,\gamma)d\eta_j.
\end{aligned}
$$



(a) EP-GIG($\gamma=\frac{3}{2}, q=\frac{1}{2}, \alpha=1$) (b) EP-GIG($\gamma=\frac{5}{2}, q=\frac{1}{2}, \alpha=1$)

(c) GT (or GDP) ($q=1$ and $\lambda=1$) (d) GT ($q=2$ and $\lambda=1$)

(e) EG ($q=1$ and $\lambda=1$) (f) EG ($q=2$ and $\lambda=1$)

Figure 2: Penalty functions induced from $1/2$-bridge scale mixture priors, exponential power-inverse gamma (or generalized t, GT) priors and exponential power-gamma (EG) priors.

Here we omit some terms that are independent of parameters σ and $\mathbf{b}$. In fact, we only need to calculate $E(\eta_j^{-1}|b_j^{(t)},\sigma^{(t)})$ in the E-step. It follows from Proposition 16 (see Appendix A) that

$$w_j^{(t+1)} \triangleq E(\eta_j^{-1}|b_j^{(t)},\sigma^{(t)}) = \frac{\alpha^{1/2}}{\left[\beta+|b_j^{(t)}|^q/\sigma^{(t)}\right]^{1/2}} \frac{K_{(\gamma q-q-1)/q}\left(\sqrt{\alpha[\beta+|b_j^{(t)}|^q/\sigma^{(t)}]}\right)}{K_{(\gamma q-1)/q}\left(\sqrt{\alpha[\beta+|b_j^{(t)}|^q/\sigma^{(t)}]}\right)}. \quad (5)$$

There do not exist analytic computational formulae for arbitrary modified Bessel functions K_ν. Thus, in general we need to resort to a numerical approximation of the Bessel function. Fortunately, however, when γ and q take the special values in Appendix B, we have closed-form expressions for

(γ, q)	$\gamma=\frac{1}{2}, q=1$	$\gamma=\frac{3}{2}, q=1$	$\gamma=0, q=2$	$\gamma=1, q=2$
$w_j=$	$\frac{1+\sqrt{\alpha(\beta+\sigma^{-1}\lvert b_j\rvert)}}{\beta+\sigma^{-1}\lvert b_j\rvert}$	$\sqrt{\frac{\alpha}{\beta+\sigma^{-1}\lvert b_j\rvert}}$	$\frac{1+\sqrt{\alpha(\beta+\sigma^{-1}b_j^2)}}{\beta+\sigma^{-1}b_j^2}$	$\sqrt{\frac{\alpha}{\beta+\sigma^{-1}b_j^2}}$

Table 1: E-steps of EM for different settings of γ and q. Here we omit superscripts "(t)".

the corresponding Bessel functions and thus for the w_j. In particular, we have from Proposition 17 (see Appendix A) that

$$w_j^{(t+1)} = \begin{cases} \left[\frac{\sigma^{(t)}\alpha}{\sigma^{(t)}\beta+|b_j^{(t)}|^q}\right]^{1/2} & (\gamma q-1)/q = 1/2, \\ \frac{\sigma^{(t)}+[\sigma^{(t)}\alpha(\sigma^{(t)}\beta+|b_j^{(t)}|^q)]^{1/2}}{\sigma^{(t)}\beta+|b_j^{(t)}|^q} & (\gamma q-1)/q = -1/2, \\ \frac{3\sigma^{(t)}}{\sigma^{(t)}\beta+|b_j^{(t)}|^q} + \frac{\sigma^{(t)}\alpha}{\sigma^{(t)}+[\sigma^{(t)}\alpha(\sigma^{(t)}\beta+|b_j^{(t)}|^q)]^{1/2}} & (\gamma q-1)/q = -3/2. \end{cases}$$

In Table 1 we list these cases with different settings of γ and q.

The M-step maximizes $Q(\mathbf{b},\sigma|\mathbf{b}^{(t)},\sigma^{(t)})$ with respect to $(\mathbf{b},\sigma)$. In particular, we have:

$$\mathbf{b}^{(t+1)} = \underset{\mathbf{b}}{\operatorname{argmin}}\ (\mathbf{y}-\mathbf{X}\mathbf{b})^T(\mathbf{y}-\mathbf{X}\mathbf{b}) + \sum_{j=1}^{p} w_j^{(t+1)}|b_j|^q,$$

$$\sigma^{(t+1)} = \frac{q}{qn+2p}\Big\{(\mathbf{y}-\mathbf{X}\mathbf{b}^{(t+1)})^T(\mathbf{y}-\mathbf{X}\mathbf{b}^{(t+1)}) + \sum_{j=1}^{p} w_j^{(t+1)}|b_j^{(t+1)}|^q\Big\}.$$

4.2 A Hierarchy for Grouped Variable Selection

In the hierarchy specified previously each b_j is assumed to have distinct scale η_j. We can also let several b_j share a common scale parameter, thereby obtaining a Bayesian approach to group sparsity (Yuan and Lin, 2007). We next briefly describe this approach.

Let I_l for $l=1,\ldots,g$ be a partition of $I=\{1,2,\ldots,p\}$; that is, $\cup_{j=1}^{g} I_j = I$ and $I_j \cap I_l = \emptyset$ for $j \neq l$. Let p_l be the cardinality of I_l, and $\mathbf{b}_l = \{b_j : j \in I_l\}$ denote the subvectors of $\mathbf{b}$, for $l=1,\ldots,g$. The hierarchy is then specified as

$$\begin{aligned} [\mathbf{y}|\mathbf{b},\sigma] &\sim N(\mathbf{y}|\mathbf{X}\mathbf{b},\sigma\mathbf{I}_n), \\ [b_j|\eta_l,\sigma] &\overset{iid}{\sim} \mathsf{EP}(b_j|0,\sigma\eta_l,q), \quad j \in I_l \\ [\eta_l|\gamma_l,\beta,\alpha] &\overset{ind}{\sim} \mathsf{GIG}(\eta_l|\gamma_l,\beta,\alpha), \quad l=1,\ldots,g. \end{aligned}$$

Moreover, given σ, the $\mathbf{b}_l$ are conditionally independent. By integrating out η_l, the marginal density of $\mathbf{b}_l$ conditional on σ is then

$$p(\mathbf{b}_l|\sigma) = \frac{K_{\frac{\gamma_l q-p_l}{q}}(\sqrt{\alpha(\beta+\sigma^{-1}\|\mathbf{b}_l\|_q^q)})}{\left[2^{\frac{q+1}{q}}\sigma^{\frac{1}{q}}\Gamma(\frac{q+1}{q})\right]^{p_l}K_{\gamma_l}(\sqrt{\alpha\beta})}\frac{\alpha^{p_l/(2q)}}{\beta^{\gamma_l/2}}\left[\beta+\sigma^{-1}\|\mathbf{b}_l\|_q^q\right]^{(\gamma_l q-p_l)/(2q)},$$

which implies $\mathbf{b}_l$ is non-factorial. The posterior distribution of η_l on $\mathbf{b}_l$ is then $\mathsf{GIG}(\eta_l|\frac{\gamma_l q-p_l}{q},\ \beta+\sigma^{-1}\|\mathbf{b}_l\|_q^q,\ \alpha)$.

In this case, the iterative procedure for $(\mathbf{b}, \sigma)$ is given by

$$\mathbf{b}^{(t+1)} = \underset{\mathbf{b}}{\operatorname{argmin}} \, (\mathbf{y}-\mathbf{X}\mathbf{b})^T(\mathbf{y}-\mathbf{X}\mathbf{b}) + \sum_{l=1}^{g} w_l^{(t+1)} \|\mathbf{b}_l\|_q^q,$$

$$\sigma^{(t+1)} = \frac{q}{qn+2p}\Big\{(\mathbf{y}-\mathbf{X}\mathbf{b}^{(t+1)})^T(\mathbf{y}-\mathbf{X}\mathbf{b}^{(t+1)}) + \sum_{l=1}^{g} w_l^{(t+1)} \|\mathbf{b}_l^{(t+1)}\|_q^q\Big\},$$

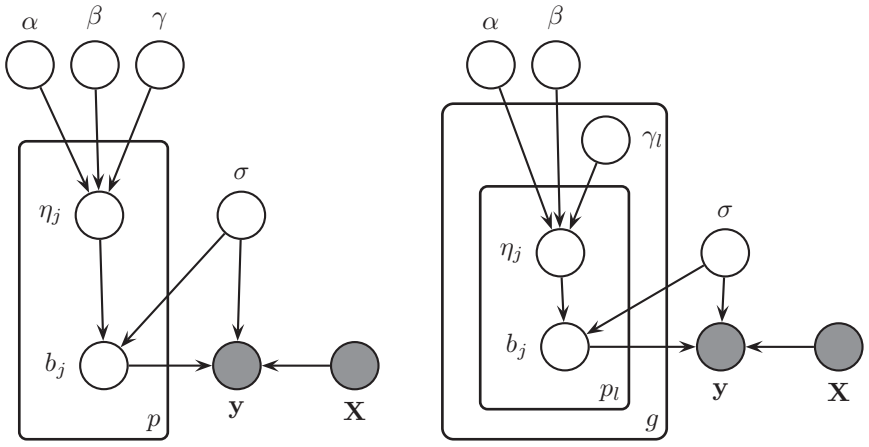
where for $l = 1, \ldots, g$,

$$w_l^{(t+1)} = \frac{\alpha^{1/2}}{\left[\beta + \|\mathbf{b}_l^{(t)}\|_q^q/\sigma^{(t)}\right]^{1/2}} \frac{K_{\frac{\gamma_l q - q - p_l}{q}}\left(\sqrt{\alpha[\beta + \|\mathbf{b}_l^{(t)}\|_q^q/\sigma^{(t)}]}\right)}{K_{\frac{\gamma_l q - p_l}{q}}\left(\sqrt{\alpha[\beta + \|\mathbf{b}_l^{(t)}\|_q^q/\sigma^{(t)}]}\right)}.$$

Recall that there is usually no analytic computation for $w_l^{(t+1)}$. However, setting γ_l such that $\frac{\gamma_l q - p_l}{q} = \frac{1}{2}$ or $\frac{\gamma_l q - p_l}{q} = -\frac{1}{2}$ yields an analytic computation. As a result, we have

$$w_j^{(t+1)} = \begin{cases} \left[\frac{\sigma^{(t)}\alpha}{\sigma^{(t)}\beta + \|\mathbf{b}_l^{(t)}\|_q^q}\right]^{1/2} & (\gamma_l q - p_l)/q = 1/2, \\ \frac{\sigma^{(t)} + \left[\sigma^{(t)}\alpha(\sigma^{(t)}\beta + \|\mathbf{b}_l^{(t)}\|_q^q)\right]^{1/2}}{\sigma^{(t)}\beta + \|\mathbf{b}_l^{(t)}\|_q^q} & (\gamma_l q - p_l)/q = -1/2. \end{cases}$$

Figure 3 depicts the hierarchical models in Section 4.1 and 4.2. It is clear that when $g = p$ and $p_1 = \cdots = p_g = 1$, the models are identical.



(a) independent (b) grouped

Figure 3: Graphical representations.

4.3 Extensions to Logistic Regression

Another extension is the application to penalized logistic regression for classification. We consider a binary classification problem in which $y \in \{0, 1\}$ now represents the label of the corresponding

input vector $\mathbf{x}$. In the logistic regression model the expected value of y_i is given by

$$P(y_i = 1|\mathbf{x}_i) = \frac{1}{1+\exp(-\mathbf{x}_i^T\mathbf{b})} \triangleq \pi_i.$$

In this case $\sigma = 1$ and the log-likelihood function becomes

$$\log p(\mathbf{y}|\mathbf{b}) = \sum_{i=1}^{n}[y_i \log \pi_i + (1-y_i)\log(1-\pi_i)].$$

Given the tth estimate $\mathbf{b}^{(t)}$ of $\mathbf{b}$, the E-step of EM calculates

$$\begin{aligned} Q(\mathbf{b}|\mathbf{b}^{(t)}) &\triangleq \log p(\mathbf{y}|\mathbf{b}) + \sum_{j=1}^{p}\int \log p[b_j|\eta_j]p(\eta_j|b_j^{(t)},\alpha,\beta,\gamma)d\eta_j \\ &\propto \sum_{i=1}^{n}[y_i\log\pi_i + (1-y_i)\log(1-\pi_i)] - \frac{1}{2}\sum_{j=1}^{p} w_j^{(t+1)}|b_j|^q. \end{aligned}$$

As for the M-step, a feasible approach is to first obtain a quadratic approximation to the log-likelihood function based on its second-order Taylor series expansion at the current estimate $\mathbf{b}^{(t)}$ of the regression vector $\mathbf{b}$. We accordingly formulate a penalized linear regression model. In particular, the M-step solves the following optimization problem

$$\mathbf{b}^{(t+1)} = \underset{\mathbf{b}\in\mathbb{R}^p}{\operatorname{argmin}} (\tilde{\mathbf{y}}-\mathbf{X}\mathbf{b})^T\mathbf{W}(\tilde{\mathbf{y}}-\mathbf{X}\mathbf{b}) + \sum_{j=1}^{p} w_j^{(t+1)}|b_j|^q,$$

where $\tilde{\mathbf{y}}$, the working response, is defined by $\tilde{\mathbf{y}} = \mathbf{X}\mathbf{b}^{(t)} + \mathbf{W}^{-1}(\mathbf{y}-\pi)$, $\mathbf{W}$ is a diagonal matrix with diagonal elements $\pi_i(1-\pi_i)$, and $\pi = (\pi_1,\ldots,\pi_n)^T$. Note that here the π are evaluated at $\mathbf{b}^{(t)}$.

5. Iteratively Re-weighted ℓ_q Methods

We employ a penalty induced from the EP-GIG prior $\mathsf{EGIG}(b|\alpha_0,\beta_0,\gamma,q)$. Let

$$R(|b|^q) \triangleq \frac{\gamma q-1}{2q}\log(\beta_0+|b|^q) - \log K_{\frac{\gamma q-1}{q}}(\sqrt{\alpha_0(\beta_0+|b|^q)}) \propto -\log \mathsf{EGIG}(b|\alpha_0,\beta_0,\gamma,q).$$

Then the penalized regression problem is

$$\min_{\mathbf{b}} \Big\{F(b) \triangleq \frac{1}{2}\|\mathbf{y}-\mathbf{X}\mathbf{b}\|_2^2 + \lambda\sum_{j=1}^{p} R(|b_j|^q)\Big\},$$

which can be solved via an iteratively reweighted ℓ_q method. Given the tth estimate $\mathbf{b}^{(t)}$ of $\mathbf{b}$, the method considers the first-order Taylor approximation of $R(|b_j|^q)$ w.r.t. $|b_j|^q$ at $|b_j^{(t)}|^q$ and solves the following problem

$$\min_{\mathbf{b}} \Big\{Q(\mathbf{b}|\mathbf{b}^{(t)}) \triangleq \frac{1}{2}\|\mathbf{y}-\mathbf{X}\mathbf{b}\|_2^2 + \lambda\sum_{j=1}^{p}\big[R(|b_j^{(t)}|^q) + \omega_j^{(t+1)}(|b_j|^q - |b_j^{(t)}|^q)\big]\Big\},$$

which is equivalent to

$$\min_{\mathbf{b}} \frac{1}{2}\|\mathbf{y}-\mathbf{X}\mathbf{b}\|_2^2+\lambda\sum_{j=1}^{p}\omega_j^{(t+1)}|b_j|^q.$$

Here $\omega_j^{(t+1)}=\frac{\partial R(|b_j|^q)}{\partial|b_j|^q}\Big|_{b_j=b_j^{(t)}}$. It follows from Theorem 6-(ii) that

$$\omega_j=\frac{1}{2}\frac{\sqrt{\alpha_0}}{\sqrt{\beta_0+|b_j|^q}}\frac{K_{\frac{\gamma q-1}{q}-1}(\sqrt{\alpha_0(\beta_0+|b_j|^q)})}{K_{\frac{\gamma q-1}{q}}(\sqrt{\alpha_0(\beta_0+|b_j|^q)})}. \tag{6}$$

5.1 Relationship between EM and Iteratively Re-weighted Methods

Under certain conditions, Zou and Li (2008) established a relationship between their LLA algorithm and an EM algorithm by using an inverse Laplace transformation. In particular, calculating weights in the former is equivalent to calculating the E-step in the latter. In our case, furthermore, Theorem 6 shows the weights are equal to the expectations involved in the corresponding EM algorithm up to the constant $1/2$.

We pursue this relationship here, focusing on the relationship of the EM algorithm in Section 4.1 with the iteratively reweighted ℓ_q method proposed above. Letting $\alpha_0=\alpha/\sigma$, $\beta_0=\beta\sigma$ and $\lambda=\sigma$, we immediately see that $2\omega_j$'s in (6) are equal to w_j's in (5). This implies the iteratively reweighted minimization method is identical to the EM algorithm given in Section 4.1. When $q=2$, the EM algorithm is identical to the reweighted ℓ_2 method and corresponds to a local quadratic approximation (Fan and Li, 2001; Hunter and Li, 2005). When $q=1$, the EM algorithm is reweighted ℓ_1 minimization and corresponds to an LLA.

In particular, when we set $\gamma=1$ and $q=2$, the EM algorithm is the same as one studied by Daubechies et al. (2010). This implies that the reweighted ℓ_2 method of Daubechies et al. (2010) can be equivalently viewed as an EM algorithm based on our proposed EP-GIG in Example 5 of Appendix B. When the EM algorithm is based on our proposed EP-GIG prior in Example 4 of Appendix B (i.e., $\gamma=1$ and $q=2$), we obtain the combination of the reweighted ℓ_2 method of Daubechies et al. (2010) and the reweighted ℓ_2 method of Chartrand and Yin (2008).

When $\gamma=\frac{3}{2}$ and $q=1$, the EM algorithm (see Table 1) is equivalent to a reweighted ℓ_1 method, which in turn has a close connection with the reweighted ℓ_2 method of Daubechies et al. (2010). Additionally, the EM algorithm based on $\gamma=\frac{1}{2}$ and $q=1$ (see Table 1) can be regarded as the combination of the above reweighted ℓ_1 method and the reweighted ℓ_1 of Candès et al. (2008). Interestingly, the EM algorithm based on the EP-GIG priors given in Examples 7 and 8 of Appendix B (i.e., $\gamma=\frac{3}{2}$ and $q=\frac{1}{2}$ or $\gamma=\frac{5}{2}$ and $q=\frac{1}{2}$) corresponds a reweighted $\ell_{1/2}$ method.

In is also worth mentioning that in Appendix C we present EP-Jeffreys priors. Using this prior, we can establish the close relationship of the adaptive lasso of Zou (2006) with an EM algorithm. In particular, when $q=1$, the EM algorithm based on the Jeffreys prior is equivalent to the adaptive lasso.

5.2 Convergence Analysis

Owing to the equivalence between the iteratively reweighted ℓ_q method and the EM algorithm, we investigate convergence analysis based on the iteratively reweighted ℓ_q method. Using the previous notation, we have the following theorem.

Lemma 8 *Let* $\{\mathbf{b}^{(t)} : 0,1,2,\ldots\}$ *be a sequence defined by the the iteratively reweighted* ℓ_q *method. Then*

$$F(\mathbf{b}) \leq Q(\mathbf{b}|\mathbf{b}^{(t)}) \quad \textit{and only} \quad F(\mathbf{b}^{(t)}) = Q(\mathbf{b}^{(t)}|\mathbf{b}^{(t)}).$$

Furthermore,

$$F(\mathbf{b}^{(t+1)}) \leq F(\mathbf{b}^{(t)})$$

with equality if and only if $\mathbf{b}^{(t+1)} = \mathbf{b}^{(t)}$.

It follows from Theorem 7 that $\frac{\partial R(|b|^q)}{\partial |b|^q} < 0$. Thus, $R(|b|^q)$ is strictly concave in $|b|^q$. Accordingly, the lemma is proven. Since $F(\mathbf{b}^{(t)}) \geq 0$, this lemma shows that $F(\mathbf{b}^{(t)})$ converges monotonically to some $F^* \geq 0$. In fact, the iteratively reweighted ℓ_q method enjoys the same convergence as the standard EM algorithm (Dempster et al., 1977; Wu, 1983). Let $\mathcal{A}(\mathbf{b}^{(t)})$ be the set of values of $\mathbf{b}$ that minimize $\mathbf{Q}(\mathbf{b}|\mathbf{b}^{(t)})$ over $\Omega \subset \mathbb{R}^p$ and $\mathcal{S}$ be the set of stationary points of F in the interior of Ω. From the Zangwill global convergence theorem (Wu, 1983; Sriperumbudur and Lanckriet, 2009) we have that

Theorem 9 *Let* $\{\mathbf{b}^{(t)}\}$ *be an iterative sequence generated by* $\mathbf{b}^{(t+1)} \in \mathcal{A}(\mathbf{b}^{(t)})$. *Suppose that* (i) $\mathcal{A}(\mathbf{b}^{(t)})$ *is closed over the complement of* $\mathcal{S}$ *and that* (ii)

$$F(\mathbf{b}^{(t+1)}) < F(\mathbf{b}^{(t)}) \quad \textit{for all } \mathbf{b}^{(t)} \notin \mathcal{S}.$$

Then all the limit points of $\{\mathbf{b}^{(t)}\}$ *are stationary points of* $F(\mathbf{b})$ *and* $F(\mathbf{b}^{(t)})$ *converges monotonically to* $F(\mathbf{b}^*)$ *for some stationary point* $\mathbf{b}^*$.

5.3 Oracle Properties

We now study the oracle property of our sparse estimator based on Laplace scale mixture priors. For this purpose, following the setup of Zou and Li (2008), we assume two conditions: (1) $y_i = \mathbf{x}_i^T\mathbf{b}^* + \varepsilon_i$ where $\varepsilon_1,\ldots,\varepsilon_n$ are i.i.d errors with mean 0 and variance σ^2; (2) $\mathbf{X}^T\mathbf{X}/n \to \mathbf{C}$ where $\mathbf{C}$ is a positive definite matrix. Let $\mathcal{A} = \{j : b_j^* \neq 0\}$. Without loss of generality, we assume that $\mathcal{A} = \{1,2,\ldots,p_0\}$ with $p_0 < p$. Thus, partition $\mathbf{C}$ as

$$\begin{bmatrix} \mathbf{C}_{11} & \mathbf{C}_{12} \\ \mathbf{C}_{21} & \mathbf{C}_{22} \end{bmatrix},$$

where $\mathbf{C}_{11}$ is $p_0 \times p_0$. Additionally, let $\mathbf{b}_1^* = \{b_j^* : j \in \mathcal{A}\}$ and $\mathbf{b}_2^* = \{u_{nj} : j \notin \mathcal{A}\}$.

We in particular consider the following one-step sparse estimator:

$$\mathbf{b}_n^{(1)} = \underset{\mathbf{b}}{\operatorname{argmin}} \, (\mathbf{y}-\mathbf{X}\mathbf{b})^T(\mathbf{y}-\mathbf{X}\mathbf{b}) + \lambda_n \sum_{j=1}^{p} |b_j| \frac{Q_{\gamma-1}(\alpha_n(\beta_n + |b_j^{(0)}|))}{Q_{\gamma-1}(\alpha_n(\beta_n+1))},$$

where $Q_\nu(z) = K_{\nu-1}(\sqrt{z})/(\sqrt{z}K_\nu(\sqrt{z}))$ and $\mathbf{b}^{(0)} = (b_1^{(0)},\ldots,b_p^{(0)})^T$ is a root-n-consistent estimator of $\mathbf{b}^*$. The following theorem shows that this estimator has the oracle property. That is,

Theorem 10 *Let* $\mathbf{b}_{n1}^{(1)} = \{b_{nj}^{(1)} : j \in \mathcal{A}\}$ *and* $\mathcal{A}_n = \{j : b_{nj}^{(1)} \neq 0\}$. *Suppose that* $\lambda_n \to \infty$, $\lambda_n/\sqrt{n} \to 0$, $\alpha_n/n \to c_1$ *and* $\alpha_n\beta_n \to c_2$, *or that* $\lambda_n/n^{1/4} \to \infty$, $\lambda_n/\sqrt{n} \to 0$, $\alpha_n/\sqrt{n} \to c_1$ *and* $\alpha_n\beta_n \to c_2$. *Here* $c_1, c_2 \in (0,\infty)$. *Then* $\mathbf{b}_n^{(1)}$ *satisfies the following properties:*

(1) *Consistency in variable selection:* $\lim_{n\to\infty} P(\mathcal{A}_n = \mathcal{A}) = 1$.

(2) *Asymptotic normality:* $\sqrt{n}(\mathbf{b}_{n1}^{(1)} - \mathbf{b}_1^*) \to_d N(0, \sigma^2\mathbf{C}_{11}^{-1})$.

6. Experimental Studies

In this paper our principal purpose has been to provide a new hierarchical framework within which we can construct sparsity-inducing priors and EM algorithms. In this section we conduct an experimental investigation of particular instances of these EM algorithms. In particular, we study the cases in Table 1. We also studied two EM algorithms based on the generalized t priors, that is, the exponential power-inverse gamma priors (see Section 3.1). For simplicity of presentation, we denote them by "Method 1," "Method 2," "Method 3," "Method 4," "Method 5," "Method 6," and "Method 7," respectively. Table 2 lists their EP-GIG prior specifications (the notation is the same as in Section 3). As we see, using the EP-GIG priors given in Examples 7 and 8 (see Appendix B) yields EM algorithms with closed-form E-steps. However, the corresponding M-steps are a weighted $\ell_{1/2}$ minimization problem, which is not efficiently solved. Thus, we did not implement such EM algorithms.

For Method 1, Method 2, Method 3, Method 5 and Method 6, we fix $\alpha = 1$ and $\sigma^{(0)} = 1$, and use the cross validation method to select β. In Method 4 and Method 7, the parameter λ was selected by using cross validation. In addition, we implemented the lasso, the adaptive lasso (adLasso) and the SCAD-based method for comparison. For the lasso, the adLasso and the reweighted ℓ_1 problems in the M-step, we solved the optimization problems by a coordinate descent algorithm (Mazumder et al., 2011).

Method 1	Method 2	Method 3	Method 4
$\mathsf{EGIG}(b\|\sigma^{-1},\sigma\beta,\frac{1}{2},1)$ $(q=1,\gamma=\frac{1}{2})$	$\mathsf{EGIG}(b\|\sigma^{-1},\sigma\beta,\frac{3}{2},1)$ $(q=1,\gamma=\frac{3}{2})$	$\mathsf{EGIG}(b\|\sigma^{-1},\sigma\beta,-\frac{1}{2},1)$ $(q=1,\gamma=-\frac{1}{2})$	$\mathsf{GT}(b\|0,\frac{\sigma}{\lambda},\frac{1}{2},1)$ $(q=1,\tau=1)$
Method 5	**Method 6**	**Method 7**	**AdLasso**
$\mathsf{EGIG}(b\|\sigma^{-1},\sigma\beta,0,2)$ $(q=2,\gamma=0)$	$\mathsf{EGIG}(b\|\sigma^{-1},\sigma\beta,1,2)$ $(q=2,\gamma=1)$	$\mathsf{GT}(b\|0,\frac{\sigma}{\lambda},\frac{1}{2},2)$ $(q=2,\tau=1)$	$\propto \exp(-\|b\|^{1/2})$ $(q=\frac{1}{2})$

Table 2: The EP-GIG specifications of the algorithms.

Recall that Method 1, Method 2, Method 3, Method 4 and AdLasso in fact work with the nonconvex penalties. In particular, Method 1, Method 2 and Method 3 are based on the Laplace scale mixture priors proposed in Appendix B. Method 4 is based on the GDP prior by Armagan et al. (2011) and Lee et al. (2010), and we employed the $\ell_{1/2}$ penalty in the adLasso. Thus, this adLasso is equivalent to the EM algorithm which given in Appendix D. Additionally, Method 5 and Method 6 are based on the Gaussian scale mixture priors given in Appendix B, and Method 7 is based on the Cauchy prior. In Appendix C we present an EM algorithm based on the EP-Jeffreys prior. This algorithm can be also regarded as an adaptive lasso with weights $1/|b_j^{(t)}|$. Since the performance of the algorithms is same to that of Method 4, we did not include the results with this prior. We also did not report the results with the Gaussian scale mixture given in Example 6 of Appendix B, because they are almost identical to those with Method 5 or Method 6.

6.1 Reconstruction on Simulation Data

We first evaluate the performance of each method on the simulated data which were used in Fan and Li (2001) and Zou (2006). Let $\mathbf{b} = (3, 1.5, 0, 0, 2, 0, 0, 0)^T$, $\mathbf{x}_i \overset{iid}{\sim} N(\mathbf{0}, \Sigma)$ with $\Sigma_{ij} = 0.5^{|i-j|}$, and

$\mathbf{y}_0 = \mathbf{X}\mathbf{b}$. Then Gaussian noise $\varepsilon \sim N(\mathbf{0}, \delta^2\mathbf{I}_n)$ is added to $\mathbf{y}_0$ to form the response vector $\mathbf{y} = \mathbf{y}_0 + \varepsilon$. Let $\hat{\mathbf{b}}$ denote the sparse solution obtained from each method which takes $\mathbf{X}$ and $\mathbf{y}$ as inputs and responses. Mean square error (MSE) $\|\mathbf{y}_0 - \mathbf{X}\hat{\mathbf{b}}\|_2^2/n$ is used to measure reconstruction accuracy, and the number of zeros in $\hat{\mathbf{b}}$ is employed to evaluate variable selection accuracy. If a method is accurate, the number of "correct" (C) zeros should be 5 and "incorrect" (IC) should be 0.

For each pair (n, δ), we generate 10,000 data sets. In Table 3 we report the numbers of correct and incorrect zeros as well as the average and standard deviation of MSE on the 10,000 data sets. From Table 3 we see that the nonconvex penalization methods (Methods 1, 2, 3 and 4) yield the best results in terms of reconstruction accuracy and sparsity recovery. It should be pointed out that since the weights are defined as $1/|b_j^{(t)}|^{1/2}$ in the adLasso method, the method suffers from numerical instability. In addition, Methods 5, 6 and 7 are based on reweighted ℓ_2 minimization, so they do not naturally produce sparse estimates. To achieve sparseness, they have to delete small coefficients.

	MSE(±STD)	C	IC	MSE (±STD)	C	IC	MSE (±STD)	C	IC
	$n = 60$, $\delta = 3$			$n = 120$, $\delta = 3$			$n = 120$, $\delta = 1$		
METHOD 1	**0.699(±0.63)**	4.66	0.08	**0.279(±0.26)**	4.87	0.01	**0.0253(± 0.02)**	**5.00**	0.00
METHOD 2	0.700(±0.63)	4.55	0.07	0.287(±0.30)	4.83	0.02	0.0256(±0.03)	4.99	0.00
METHOD 3	0.728(±0.60)	4.57	0.08	0.284(±0.28)	**4.93**	0.00	**0.0253(±0.02)**	**5.00**	0.00
METHOD 4	0.713(±0.68)	**4.78**	0.12	0.281(±0.26)	4.89	0.01	0.0255(±0.03)	**5.00**	0.00
METHOD 5	1.039(±0.56)	0.30	0.00	0.539(±0.28)	0.26	0.00	0.0599(±0.03)	0.77	0.00
METHOD 6	0.745(±0.66)	1.36	0.00	0.320(±0.26)	1.11	0.00	0.0262(±0.02)	4.96	0.00
METHOD 7	0.791(±0.57)	0.20	0.00	0.321(±0.28)	0.42	0.00	0.0265(±0.02)	2.43	0.00
SCAD	0.804(±0.59)	3.24	0.02	0.364(±0.30)	3.94	0.00	0.0264(±0.03)	4.95	0.00
ADLASSO	0.784(±0.57)	3.60	0.04	0.335(±0.27)	4.83	0.01	0.0283(±0.02)	4.82	0.00
LASSO	0.816(±0.53)	2.48	0.00	0.406(±0.26)	2.40	0.00	0.0450(±0.03)	2.87	0.00
RIDGE	1.012(±0.50)	0.00	0.00	0.549(±0.27)	0.00	0.00	0.0658(±0.03)	0.00	0.00

Table 3: Results on the simulated data sets.

6.2 Regression on Real Data

We apply the methods to linear regression problems and evaluate their performance on three data sets: Pyrim and Triazines (both obtained from UCI Machine Learning Repository) and the biscuit data set (the near-infrared (NIR) spectroscopy of biscuit doughs) (Breiman and Friedman, 1997). For Pyrim and Triazines data sets, we randomly held out 70% of the data for training and used the remainder for test. We repeat this process 10 times, and report the mean and standard deviation of the relative errors defined as

$$\frac{1}{n_{test}} \sum_{i=1}^{n_{test}} \left| \frac{y(\mathbf{x}_i) - \tilde{y}(\mathbf{x}_i)}{y(\mathbf{x}_i)} \right|,$$

where $y(\mathbf{x}_i)$ is the target response for the test input $\mathbf{x}_i$, and $\tilde{y}(\mathbf{x}_i)$ is the prediction value computed from a regression method. For the NIR data set, we use the supplied training and test sets: 39 instances for training and the remaining 31 for test (Breiman and Friedman, 1997). Since each response of the NIR data includes 4 attributes ("fat," "sucrose," "flour" and "water"), we treat the data as four regression data sets; namely, the input instances and each-attribute responses constitute one data set.

The results are listed in Table 4. We see that the four new methods outperform the adaptive lasso and lasso in most cases. In particular, Methods 1, 2, 3 and 4 (the nonconvex penalization) yield the best performance over the first two data sets, and Methods 5, 6 and 7 are the best on the NIR data sets. This implies that nonconvex penalization outperforms convex penalization in sparsity, but not always in classification accuracy. The reason is that sparsity is not always in concert with classification accuracy.

	PYRIM	TRIAZINES	NIR(FAT)	NIR(SUCROSE)	NIR(FLOUR)	NIR(WATER)
METHOD 1	**0.1342(±0.065)**	0.2786(±0.083)	0.0530	0.0711	0.0448	0.0305
METHOD 2	0.1363(±0.066)	**0.2704(±0.075)**	0.0556	0.0697	0.0431	0.0312
METHOD 3	0.1423(±0.072)	0.2792(±0.081)	0.0537	0.0803	0.0440	0.0319
METHOD 4	0.1414(±0.065)	0.2772(±0.081)	0.0530	0.0799	0.0448	0.0315
METHOD 5	0.1381(±0.065)	0.2917(±0.089)	0.0290	0.0326	0.0341	0.0210
METHOD 6	0.2352(±0.261)	0.3364(±0.079)	0.0299	**0.0325**	0.0341	**0.0208**
METHOD 7	0.1410(±0.065)	0.3109(±0.110)	**0.0271**	0.0423	**0.0277**	0.0279
SCAD	0.1419(±0.064)	0.2807(±0.079)	0.0556	0.0715	0.0467	0.0352
ADLASSO	0.1430(±0.064)	0.2883(±0.080)	0.0533	0.0803	0.0486	0.0319
LASSO	0.1424(±0.064)	0.2804(±0.079)	0.0608	0.0799	0.0527	0.0340

Table 4: Relative error of each method on the three data sets. The numbers of instances (n) and numbers of features (p) of each data set are: $n = 74$ and $p = 27$ in Pyrim, $n = 186$ and $p = 60$ in Triazines, and $n = 70$ and $p = 700$ in NIR.

6.3 Experiments on Group Variable Selection

Here we use $p = 32$ with 8 groups, each of size 4. Let $\beta_{1:4} = (3, 1.5, 2, 0.5)^T$, $\beta_{9:12} = \beta_{17:20} = (6, 3, 4, 1)^T$, $\beta_{25:28} = (1.5, 0.75, 1, 0.25)^T$ with all other entries set to zero, while $\mathbf{X}$, $\mathbf{y}_0$, and $\mathbf{y}$ are defined in the same way as in Section 6.1. If a method is accurate, the number of "correct" (C) zeros should be 16 and "incorrect" (IC) should be 0. Results are reported in Table 5.

6.4 Experiments on Classification

In this subsection we apply our hierarchical penalized logistic regression models in Section 4.3 to binary classification problems on five real-world data sets: Ionosphere, Spambase, Sonar, Australian, and Heart from UCI Machine Learning Repository and Statlog. Table 6 gives a brief description of these five data sets.

In the experiments, the input matrix $\mathbf{X} \in \mathbb{R}^{n \times p}$ is normalized such that $\sum_{i=1}^n x_{ij} = 0$ and $\sum_{i=1}^n x_{ij}^2 = n$ for all $j = 1, \cdots, p$. For each data set, we randomly choose 70% for training and the rest for test. We repeat this process 10 times and report the mean and the standard deviation of classification error rate. The results in Table 7 are interesting; in most cases Methods 1, 2, 3 and 4 based on the nonconvex penalties outperform the other methods in both accuracy and sparsity.

	MSE(±STD)	C	IC	MSE (±STD)	C	IC	MSE (±STD)	C	IC
	$n = 60$, $\delta = 3$			$n = 120$, $\delta = 3$			$n = 120$, $\delta = 1$		
METHOD 1′	2.531(±1.01)	15.85	0.31	1.201(±0.45)	**16.00**	0.14	0.1335(±0.048)	15.72	0.01
METHOD 2′	2.516(±1.06)	15.87	0.28	1.200(±0.43)	15.97	0.10	0.1333(±0.047)	15.87	**0.00**
METHOD 3′	2.445(±0.96)	**15.88**	0.54	1.202(±0.43)	15.98	0.25	**0.1301**(±0.047)	**16.00**	0.01
METHOD 4′	2.674(±1.12)	15.40	0.30	1.220(±0.45)	15.79	0.49	0.1308(±0.047)	**16.00**	**0.00**
METHOD 5′	**2.314(±0.90)**	5.77	0.04	1.163(±0.41)	7.16	0.03	0.1324(±0.047)	**16.00**	0.01
METHOD 6′	2.375(±0.92)	10.18	0.04	**1.152(±0.41)**	15.56	0.03	0.1322(±0.047)	**16.00**	**0.00**
METHOD 7′	2.478(±0.97)	9.28	0.05	1.166(±0.41)	14.17	0.03	0.1325(±0.047)	15.96	**0.00**
GLASSO	2.755(±0.92)	5.52	**0.00**	1.478(±0.48)	3.45	**0.00**	0.1815(±0.058)	3.05	**0.00**
ADLASSO	3.589(±1.10)	11.36	2.66	1.757(±0.56)	11.85	1.42	0.1712(±0.058)	14.09	0.32
LASSO	3.234(±0.99)	9.17	1.29	1.702(±0.52)	8.53	0.61	0.1969(±0.060)	8.03	0.05

Table 5: Results on the simulated data sets.

	Ionosphere	Spambase	Sonar	Australian	Heart
n	351	4601	208	690	270
p	33	57	60	14	13

Table 6: The description of data sets. Here n: the numbers of instances; p: the numbers of features.

	IONOSPHERE	SPAMBASE	SONAR	AUSTRALIAN	HEART
METHOD 1	**9.91(±2.19)**	7.54(±0.84)	**18.71(±5.05)**	12.46(±2.08)	13.83(±3.33)
METHOD 2	10.19(±2.03)	**7.47(±0.85)**	19.19(±5.18)	12.56(±2.06)	14.20(±3.50)
METHOD 3	10.00(±1.95)	7.58(±0.83)	19.03(±4.35)	12.61(±2.15)	14.32(±3.60)
METHOD 4	10.66(±1.94)	7.61(±0.83)	21.65(±5.11)	12.65(±2.14)	13.95(±3.49)
METHOD 5	11.51(±3.77)	8.78(±0.41)	21.61(±5.70)	**12.03(±1.74)**	**13.21(±3.14)**
METHOD 6	11.51(±3.72)	8.86(±0.41)	21.94(±5.85)	13.24(±2.22)	14.57(±3.38)
METHOD 7	11.70(±4.06)	9.49(±0.33)	22.58(±5.84)	14.11(±2.48)	13.46(±3.10)
SCAD	10.47(±2.06)	7.58(±0.83	21.94(±5.60)	12.66(±2.08)	13.83(±3.43)
$\ell_{1/2}$	10.09(±1.67)	7.51(±0.86)	20.00(±5.95)	12.56(±2.15)	14.20(±3.78)
ℓ_1	10.47(±1.96)	7.57(±0.83)	21.61(±5.11)	12.66(±2.15)	13.95(±3.49)

Table 7: Misclassification rate (%) of each method on the five data sets.

7. Conclusions

In this paper we have proposed a family of sparsity-inducing priors that we call *exponential power-generalized inverse Gaussian* (EP-GIG) distributions. We have defined the EP-GIG family as a mixture of exponential power distributions with a generalized inverse Gaussian (GIG) density. EP-GIG are extensions of Gaussian scale mixtures and Laplace scale mixtures. As a special example

of the EP-GIG framework, the mixture of Laplace with GIG can induce a family of nonconvex penalties. In Appendix B, we have presented five new EP-GIG priors which can induce nonconvex penalties.

Since GIG distributions are conjugate with respect to the exponential power distribution, EP-GIG are natural for Bayesian sparse learning. In particular, we have developed hierarchical Bayesian models and devised EM algorithms for finding sparse solutions. We have also shown how this framework can be applied to grouped variable selection and logistic regression problems. Our experiments have shown that the proposed EP-GIG priors giving rise to nonconvex penalties are potentially feasible and effective in sparsity modeling.

Acknowledgments

The authors would like to thank the Action Editor and three anonymous referees for their constructive comments on the original version of this paper. This work has been supported in part by the Natural Science Foundations of China (No. 61070239), the Google visiting faculty program, and the US ARL and the US ARO under contract/grant number W911NF-11-1-0391.

Appendix A. Proofs

We first present some mathematical preliminaries that will be needed.

A.1 Mathematical Preliminaries

The first three of the following lemmas are well known, so we omit their proofs.

Lemma 11 *Let* $\lim_{\nu\to\infty} a(\nu) = a$. *Then* $\lim_{\nu\to\infty} \left(1+\frac{a(\nu)}{\nu}\right)^{\nu} = \exp(a)$.

Lemma 12 (Stirling Formula) $\lim_{\nu\to\infty} \frac{\Gamma(\nu)}{(2\pi)^{1/2}\nu^{\nu-1/2}\exp(-\nu)} = 1$.

Lemma 13 *Assume* $z > 0$ *and* $\nu > 0$. *Then*

$$\lim_{\nu\to\infty} \frac{K_\nu(\nu^{1/2}z)}{\pi^{1/2}2^{\nu-1/2}\nu^{(\nu-1)/2}z^{-\nu}\exp(-\nu)\exp(-z^2/4)} = 1.$$

Proof Consider the integral representation of $K_\nu(\nu^{1/2}z)$ as

$$\begin{aligned} K_\nu(\nu^{1/2}z) &= \pi^{-1/2}2^\nu\nu^{\nu/2}z^\nu\Gamma\big(\nu+\frac{1}{2}\big)\int_0^\infty (t^2+\nu z^2)^{-\nu-\frac{1}{2}}\cos(t)dt \\ &= \pi^{-1/2}2^\nu\nu^{-(\nu+1)/2}z^{-(\nu+1)}\Gamma\big(\nu+\frac{1}{2}\big)\int_0^\infty \frac{\cos(t)}{(1+t^2/(\nu z^2))^{\nu+\frac{1}{2}}}\cos(t)dt. \end{aligned}$$

Thus, we have

$$\begin{aligned} \lim_{\nu\to\infty} \frac{K_\nu(\nu^{1/2}z)}{\pi^{-1/2}2^\nu\nu^{-(\nu+1)/2}z^{-(\nu+1)}\Gamma\big(\nu+\frac{1}{2}\big)} &= \lim_{\nu\to\infty}\int_0^\infty \frac{\cos(t)}{(1+t^2/(\nu z^2))^{\nu+\frac{1}{2}}}\cos(t)dt \\ &= \int_0^\infty \cos(t)\exp(-t^2/z^2)dt. \end{aligned}$$

We now calculate the integral $\int_0^\infty \cos(t)\exp(-t^2/z^2)dt$ for $z>0$. We denote this integral by $\phi(z)$ and let $u=t/z$. Hence,

$$\phi(z) = z\int_0^\infty \exp(-u^2)\cos(uz)du = zf(z),$$

where $f(z)=\int_0^\infty \exp(-u^2)\cos(uz)du$. Note that

$$\begin{aligned} f'(z) &= -\int_0^\infty \exp(-u^2)\sin(uz)udu = \frac{1}{2}\int_0^\infty \sin(uz)d\exp(-u^2) \\ &= -\frac{z}{2}\int_0^\infty \exp(-u^2)\cos(uz)du = -\frac{z}{2}f(z), \end{aligned}$$

which implies that $f(z)=C\exp(-z^2/4)$ where C is a constant independent of z. We calculate $f(1)$ to obtain C. Since

$$C = \lim_{z\to+0} f(z) = \lim_{z\to+0}\int_0^\infty e^{-u^2}\cos(uz)du = \int_0^\infty e^{-u^2}du = \frac{\sqrt{\pi}}{2},$$

we have $\phi(z)=\frac{\sqrt{\pi}}{2}z\exp(-z^2/4)$. Subsequently,

$$\lim_{\nu\to\infty}\frac{K_\nu(\nu^{1/2}z)}{\pi^{-1/2}2^\nu\nu^{-(\nu+1)/2}z^{-(\nu+1)}\Gamma\left(\nu+\frac{1}{2}\right)} = \frac{\sqrt{\pi}}{2}z\exp(-z^2/4).$$

On the other hand, it follows from Lemmas 11 and 12 that

$$\lim_{\nu\to\infty}\frac{\Gamma(\nu+1/2)}{(2\pi)^{1/2}\nu^\nu\exp(-\nu)} = \lim_{\nu\to\infty}\frac{\Gamma(\nu+1/2)}{\sqrt{2\pi}\nu^\nu[1+1/(2\nu)]^\nu\exp(-\nu)\exp(-1/2)} = 1.$$

Thus,

$$\lim_{\nu\to\infty}\frac{K_\nu(\nu^{1/2}z)}{\pi^{\frac{1}{2}}2^{\nu-\frac{1}{2}}\nu^{\frac{\nu-1}{2}}z^{-\nu}\exp(-\nu)\exp(-\frac{z^2}{4})} = 1.$$

■

Lemma 14 *The modified Bessel function of the second kind $K_\gamma(u)$ satisfies the following properties:*

(1) $K_\gamma(u) = K_{-\gamma}(u)$;

(2) $K_{\gamma+1}(u) = 2\frac{\gamma}{u}K_\gamma(u) + K_{\gamma-1}(u)$;

(3) $K_{1/2}(u) = K_{-1/2}(u) = \sqrt{\frac{\pi}{2u}}\exp(-u)$;

(4) $\frac{\partial K_\gamma(u)}{\partial u} = -\frac{1}{2}(K_{\gamma-1}(u)+K_{\gamma+1}(u)) = -K_{\gamma-1}(u) - \frac{\gamma}{u}K_\gamma(u) = \frac{\gamma}{u}K_\gamma(u) - K_{\gamma+1}(u)$.

(5) *For* $\gamma\in(-\infty,+\infty)$, $K_\gamma(u)\sim\sqrt{\frac{\pi}{2u}}\exp(-u)$ *as* $u\to+\infty$.

Lemma 15 *Let $Q_\nu(z) = K_{\nu-1}(\sqrt{z})/(\sqrt{z}K_\nu(\sqrt{z}))$ where $\nu\in\mathbb{R}$ and $z>0$. Then, Q_ν is completely monotone.*

Proof When $\nu \geq 0$, the result was proved by Grosswald (1976). Thus, we only need to consider the case in which $\nu < 0$. In this case, we let $\nu = -\tau$ where $\tau > 0$. Thus,

$$Q_\nu = \frac{K_{-\tau-1}(\sqrt{z})}{\sqrt{z}K_{-\tau}(\sqrt{z})} = \frac{K_{\tau+1}(\sqrt{z})}{\sqrt{z}K_{\tau}(\sqrt{z})} = \frac{2\tau}{z} + \frac{K_{\tau-1}(\sqrt{z})}{\sqrt{z}K_{\tau}(\sqrt{z})},$$

which is obviously completely monotone. ■

The following proposition of the GIG distribution can be found in Jørgensen (1982).

Proposition 16 *Let* η *be distributed according to* $\mathsf{GIG}(\eta|\gamma,\beta,\alpha)$ *with* $\alpha > 0$ *and* $\beta > 0$*. Then*

$$E(\eta^\nu) = \Big(\frac{\beta}{\alpha}\Big)^{\nu/2} \frac{K_{\gamma+\nu}(\sqrt{\alpha\beta})}{K_\gamma(\sqrt{\alpha\beta})}.$$

We are especially interested in the cases that $\gamma = 1/2$, $\gamma = -1/2$, $\gamma = 3/2$ and $\gamma = -3/2$. For these cases, we have the following results.

Proposition 17 *Let* $\alpha > 0$ *and* $\beta > 0$*.*

(1) *If* η *is distributed according to* $\mathsf{GIG}(\eta|1/2,\beta,\alpha)$*, then*

$$E(\eta) = \frac{1+\sqrt{\alpha\beta}}{\alpha}, \quad E(\eta^{-1}) = \sqrt{\frac{\alpha}{\beta}}.$$

(2) *If* η *is distributed according to* $\mathsf{GIG}(\eta|-1/2,\beta,\alpha)$*, then*

$$E(\eta) = \sqrt{\frac{\beta}{\alpha}}, \quad E(\eta^{-1}) = \frac{1+\sqrt{\alpha\beta}}{\beta}.$$

(3) *If* η *is distributed according to* $\mathsf{GIG}(\eta|3/2,\beta,\alpha)$*, then*

$$E(\eta) = \frac{3}{\alpha} + \frac{\beta}{1+\sqrt{\alpha\beta}}, \quad E(\eta^{-1}) = \frac{\alpha}{1+\sqrt{\alpha\beta}}.$$

(4) *If* η *is distributed according to* $\mathsf{GIG}(\eta|-3/2,\beta,\alpha)$*, then*

$$E(\eta) = \frac{\beta}{1+\sqrt{\alpha\beta}}, \quad E(\eta^{-1}) = \frac{3}{\beta} + \frac{\alpha}{1+\sqrt{\alpha\beta}}.$$

Proof It follows from Lemma 14 that $K_{3/2}(u) = \frac{1+u}{u}K_{1/2}(u) = \frac{1+u}{u}K_{-1/2}(u)$.

We first consider the case that $\eta \sim \mathsf{GIG}(\eta|1/2,\beta,\alpha)$. Consequently, $E(\eta^{-1}) = \alpha/\beta$ and

$$E(\eta) = \Big(\frac{\beta}{\alpha}\Big)^{1/2} \frac{K_{\frac{3}{2}}(\sqrt{\alpha\beta})}{K_{\frac{1}{2}}(\sqrt{\alpha\beta})} = \Big(\frac{\beta}{\alpha}\Big)^{1/2} \frac{1+\sqrt{\alpha\beta}}{\sqrt{\alpha\beta}} = \frac{1+\sqrt{\alpha\beta}}{\alpha}.$$

As for the case that $\eta \sim \mathsf{GIG}(\eta|-3/2,\beta,\alpha)$, it follows from Proposition 16 that

$$E(\eta) = \Big(\frac{\beta}{\alpha}\Big)^{1/2} \frac{K_{-1/2}(\sqrt{\alpha\beta})}{K_{-3/2}(\sqrt{\alpha\beta})} = \frac{\beta}{1+\sqrt{\alpha\beta}}$$

and

$$E(\eta^{-1}) = \Big(\frac{\beta}{\alpha}\Big)^{-1/2} \frac{K_{-5/2}(\sqrt{\alpha\beta})}{K_{-3/2}(\sqrt{\alpha\beta})} = \frac{3}{\beta} + \frac{\alpha}{1+\sqrt{\alpha\beta}}.$$

Likewise, we have the second and third parts. ■

A.2 Some Limiting Properties of GIG Distributions

An interesting property of the gamma and inverse gamma distributions is given as follows.

Proposition 18 *Let* $\lambda > 0$. *Then*

(1) $\lim_{\tau\to\infty} G(\eta|\tau,\tau\lambda) = \delta(\eta|1/\lambda)$.

(2) $\lim_{\tau\to\infty} \mathsf{IG}(\eta|\tau,\tau/\lambda) = \delta(\eta|1/\lambda)$.

Here $\delta(\eta|a)$ *is the Dirac delta function; namely,*

$$\delta(\eta|a) = \begin{cases} \infty & \text{if } \eta = a, \\ 0 & \text{otherwise}. \end{cases}$$

Proof Note that

$$\begin{aligned} \lim_{\tau\to\infty} G(\eta|\tau,\tau\lambda) &= \lim_{\tau\to\infty} \frac{(\tau\lambda)^\tau}{\Gamma(\tau)} \eta^{\tau-1} \exp(-\tau\lambda\eta) \\ &= \lim_{\tau\to\infty} \frac{(\tau\lambda)^\tau}{(2\pi)^{\frac{1}{2}} \tau^{\tau-\frac{1}{2}} \exp(-\tau)} \eta^{\tau-1} \exp(-\tau\lambda\eta) \quad \text{(Use the Stirling Formula)} \\ &= \lim_{\tau\to\infty} \frac{\tau^{\frac{1}{2}}}{(2\pi)^{\frac{1}{2}}\eta} \frac{(\lambda\eta)^\tau}{\exp((\lambda\eta-1)\tau)}. \end{aligned}$$

Since $\ln u \le u-1$ for $u > 0$, with equality if and only if $u = 1$, we can obtain the proof.

The second part follows similarly. ■

As an extension of Proposition 18, we have the limiting property of GIG as follows.

Proposition 19 *Let* $\gamma \in \mathbb{R}$, $\alpha > 0$ *and* $\beta > 0$. *Then*

(1) $\lim_{\gamma\to+\infty} \mathsf{GIG}(\eta|\gamma,\beta,\gamma\alpha) = \delta(\eta|2/\alpha)$.

(2) $\lim_{\gamma\to-\infty} \mathsf{GIG}(\eta|\gamma,-\gamma\beta,\alpha) = \delta(\eta|\beta/2)$.

(3) $\lim_{\psi\to+\infty} \mathsf{GIG}(\eta|\gamma,\beta,\alpha) = \delta(\eta|\phi)$ *where* $\psi = \sqrt{\alpha\beta}$ *and* $\phi = \sqrt{\alpha/\beta} \in (0,\infty)$.

Proof Using Lemma 13,

$$\begin{aligned}\lim_{\gamma\to+\infty}\mathsf{GIG}(\eta|\gamma,\beta,\gamma\alpha) &= \lim_{\gamma\to+\infty}\frac{\gamma^{\gamma/2}(\alpha/\beta)^{\gamma/2}}{2K_\gamma(\sqrt{\gamma\alpha\beta})}\eta^{\gamma-1}\exp(-(\gamma\alpha\eta+\beta\eta^{-1})/2)\\ &= \lim_{\nu\to+\infty}\frac{\alpha^\gamma\exp(\frac{\alpha\beta}{4})\exp(-\beta\eta^{-1}/2)}{\pi^{\frac{1}{2}}2^{\gamma+\frac{1}{2}}\gamma^{-\frac{1}{2}}}\eta^{\gamma-1}\exp(-\gamma(\alpha\eta/2-1))\\ &= \lim_{\gamma\to+\infty}\frac{\eta^{-1}\gamma^{\frac{1}{2}}\exp(\frac{\alpha\beta}{4})}{(2\pi)^{\frac{1}{2}}\exp(\beta\eta^{-1}/2)}(\alpha\eta/2)^\gamma\exp(-\gamma(\alpha\eta/2-1))\\ &= \delta(\eta|2/\alpha).\end{aligned}$$

Again since $\ln u \le u-1$ for $u>0$, with equality if and only if $u=1$, we can obtain the proof of Part (1).

Let $\tau=-\gamma$. We have

$$\begin{aligned}\lim_{\gamma\to-\infty}\mathsf{GIG}(\eta|\gamma,-\gamma\beta,\alpha) &= \lim_{\tau\to+\infty}\mathsf{GIG}(\eta|-\tau,\tau\beta,\alpha)\\ &= \lim_{\tau\to+\infty}\frac{(\alpha/(\tau\beta))^{-\tau/2}}{2K_\tau(\sqrt{\tau\alpha\beta})}\eta^{-\tau-1}\exp(-(\alpha\eta+\tau\beta\eta^{-1})/2),\end{aligned}$$

due to the fact that $K_{-\tau}(\sqrt{\tau\alpha\beta})=K_\tau(\sqrt{\tau\alpha\beta})$. Accordingly, we also have the second part.

Finally, based on (1) and Lemma 14, we have that

$$\lim_{\psi\to+\infty}p(\eta)=\lim_{\psi\to+\infty}\frac{\psi^{1/2}}{\sqrt{2\pi}}\frac{1}{\exp(\frac{\psi}{2\phi\eta}(\phi\eta-1)^2)}=\delta(\eta|\phi).$$

■

A.3 The Proof of Theorem 5

With the setting that $\gamma=\frac{1}{2}+\frac{1}{q}$, we have

$$\begin{aligned}\int_0^\infty \mathsf{EP}(b|0,\eta,q)G(\eta|\gamma,\alpha/2)d\eta &= \frac{\alpha^{\frac{1}{q}+\frac{1}{4}}|b|^{\frac{q}{4}}}{2^{\frac{2}{q}+\frac{1}{2}}\Gamma(\frac{q+1}{q})\Gamma(\frac{1}{2}+\frac{1}{q})}K_{1/2}(\sqrt{\alpha|b|^q})\\ &= \frac{\alpha^{\frac{1}{q}+\frac{1}{4}}|b|^{\frac{q}{4}}}{2^{\frac{2}{q}+\frac{1}{2}}2^{-\frac{2}{q}}\sqrt{\pi}\frac{2}{q}\Gamma(\frac{2}{q})}\frac{2^{-1/2}\sqrt{\pi}}{(\alpha|b|^q)^{1/4}}\exp(-\sqrt{\alpha|b|^q})\\ &= \frac{q\alpha^{1/q}}{4\Gamma(\frac{2}{q})}\exp(-\sqrt{\alpha|b|^q})=\mathsf{EP}(b|0,\alpha^{-1/2}/2,q/2).\end{aligned}$$

Here we use the fact that $\Gamma(\frac{q+1}{q})\Gamma(\frac{1}{2}+\frac{1}{q})=2^{1-2(\frac{1}{2}+\frac{1}{q})}\sqrt{\pi}\Gamma(1+\frac{2}{q})=2^{-\frac{2}{q}}\sqrt{\pi}\frac{2}{q}\Gamma(\frac{2}{q})$.

A.4 The Proof of Theorem 6

The first part is immediate. We consider the proof of the second part. It follows from Lemma 14 that

$$\begin{aligned}\frac{\partial -\log p(b)}{\partial |b|^q} &= \frac{K_{\frac{\gamma q-1}{q}-1}(\sqrt{\alpha(\beta+|b|^q)}) + \frac{(\gamma q-1)/q}{\sqrt{\alpha(\beta+|b|^q)}} K_{\frac{\gamma q-1}{q}}(\sqrt{\alpha(\beta+|b|^q)})}{K_{\frac{\gamma q-1}{q}}(\sqrt{\alpha(\beta+|b|^q)})} \frac{1}{2}\frac{\alpha}{\sqrt{\alpha(\beta+|b|^q)}} \\ &\quad - \frac{\gamma q-1}{2q}\frac{1}{\beta+|b|^q} \\ &= \frac{1}{2}\frac{\sqrt{\alpha}}{\sqrt{\beta+|b|^q}} \frac{K_{\frac{\gamma q-1}{q}-1}(\sqrt{\alpha(\beta+|b|^q)})}{K_{\frac{\gamma q-1}{q}}(\sqrt{\alpha(\beta+|b|^q)})} = \frac{1}{2}E(\eta^{-1}|b).\end{aligned}$$

due to that $\eta|b \sim \mathsf{GIG}(\eta|(\gamma q-1)/q, \sqrt{\beta+|b|^q}, \alpha)$.

A.5 The Proof of Theorem 7

For notational simplicity, we let $z = |b^q|$, $\nu = \frac{\gamma q-1}{q}$ and $\phi(z) = \frac{\partial -\log p(b)}{\partial |b|^q}$. According to the above proof, we have

$$\phi(z) = \frac{\alpha}{2}\frac{1}{\sqrt{\alpha(\beta+z)}}\frac{K_{\nu-1}(\sqrt{\alpha(\beta+z)})}{K_{\nu}(\sqrt{\alpha(\beta+z)})}.$$

It then follows from Lemma 15 that $\phi(z)$ is completely monotone.

A.6 The Proof of Theorem 10

Let $\mathbf{b}_n^{(1)} = \mathbf{b}^* + \frac{\mathbf{u}}{\sqrt{n}}$ and

$$\hat{\mathbf{u}} = \underset{\mathbf{u}}{\operatorname{argmin}} \left\{ \Psi(\mathbf{u}) := \left\| \mathbf{y} - \mathbf{X}(\mathbf{b}^* + \frac{\mathbf{u}}{\sqrt{n}}) \right\|^2 + \lambda_n \sum_{j=1}^{p} \omega_j^{(0)} |b_j^* + \frac{u_j}{\sqrt{n}}| \right\},$$

where

$$\omega_j^{(0)} = \frac{\sqrt{\alpha_n\beta_n+\alpha_n}}{\sqrt{\alpha_n(\beta_n+|b_j^{(0)}|)}} \frac{K_{\gamma-2}(\sqrt{\alpha_n(\beta_n+|b_j^{(0)}|)})}{K_{\gamma-1}(\sqrt{\alpha_n(\beta_n+|b_j^{(0)}|)})} \frac{K_{\gamma-1}(\sqrt{\alpha_n(\beta_n+1)})}{K_{\gamma-2}(\sqrt{\alpha_n(\beta_n+1)})}.$$

Consider that

$$\Psi(\mathbf{u}) - \Psi(0) = \mathbf{u}^T(\frac{1}{n}\mathbf{X}^T\mathbf{X})\mathbf{u} - 2\frac{\varepsilon^T\mathbf{X}}{\sqrt{n}}\mathbf{u} + \lambda_n \sum_{j=1}^{p} \omega_j^{(0)} \left\{ |b_j^* + \frac{u_j}{\sqrt{n}}| - |b_j^*| \right\}.$$

We know that $\mathbf{X}^T\mathbf{X}/n \to \mathbf{C}$ and $\frac{\mathbf{X}^T\varepsilon}{\sqrt{n}} \to_d N(\mathbf{0}, \sigma^2\mathbf{C})$. We thus only consider the third term of the right-hand side of the above equation. Since $\alpha_n\beta_n \to c_1$ and $\alpha_n \to \infty$ (note that $\alpha_n/n \to c_2 > 0$ implies $\alpha_n \to +\infty$), we have

$$\frac{K_{\gamma-1}(\sqrt{\alpha_n(\beta_n+1)})}{K_{\gamma-2}(\sqrt{\alpha_n(\beta_n+1)})} \to 1.$$

If $b_j^* = 0$, then $\sqrt{n}(|b_j^* + \frac{u_j}{\sqrt{n}}| - |b_j^*|) = |u_j|$. And since $\sqrt{n}b_j^{(0)} = O_p(1)$, we have $\alpha_n|b_j^{(0)}| = (\alpha_n/\sqrt{n})\sqrt{n}|b_j^{(0)}| = O_p(1)$. Hence, $Q_{\gamma-1}(\alpha_n(\beta_n + |b_j^{(0)}|))$ converges to a positive constant in probability. As a result, we obtain

$$\frac{\lambda_n \omega_j^{(0)}}{\sqrt{n}} \to_p \to \infty.$$

due to

$$\frac{\sqrt{\alpha_n\beta_n + \alpha_n}}{\sqrt{n}} \frac{K_{\gamma-1}(\sqrt{\alpha_n\beta_n + \alpha_n})}{K_{\gamma-2}(\sqrt{\alpha_n\beta_n + \alpha_n})} \to \sqrt{c_2}.$$

If $b_j^* \neq 0$, then $\omega_j^{(0)} \to_p \frac{1}{\sqrt{|b_j^{(0)}|}} > 0$ and $\sqrt{n}(|b_j^* + \frac{u_j}{\sqrt{n}}| - |b_j^*|) \to u_j \mathrm{sgn}(b_j^*)$. Thus $\lambda_n \frac{\omega_j^{(0)}}{\sqrt{n}} \sqrt{n}(|b_j^* + \frac{u_j}{\sqrt{n}}| - |b_j^*|) \to_p 0$. The remaining parts of the proof can be immediately obtained via some slight modifications to that in Zou (2006) or Zou and Li (2008).

Appendix B. Several Special EP-GIG Distributions

We now present eight other important concrete EP-GIG distributions, obtained from particular settings of γ and q.

B.1 Example 1

We first discuss the case that $q = 1$ and $\gamma = 1/2$. That is, we employ the mixing distribution of $L(b|0,\eta)$ with $\mathsf{GIG}(\eta|1/2,\beta,\alpha)$. In this case, since

$$K_{\frac{1}{2}-1}(\sqrt{\alpha(\beta+|b|)}) = K_{-1/2}(\sqrt{\alpha(\beta+|b|)}) = \frac{(\pi/2)^{1/2}}{(\alpha(\beta+|b|))^{1/4}} \exp(-\sqrt{\alpha(\beta+|b|)})$$

and

$$K_{1/2}(\sqrt{\alpha\beta}) = \frac{(\pi/2)^{1/2}}{(\alpha\beta)^{1/4}} \exp(-\sqrt{\alpha\beta}),$$

we obtain the following pdf for $\mathsf{EGIG}(b|\alpha,\beta,1/2,1)$:

$$p(b) = \frac{\alpha^{1/2}}{4} \exp(\sqrt{\alpha\beta})(\beta+|b|)^{-1/2} \exp(-\sqrt{\alpha(\beta+|b|)}). \tag{7}$$

B.2 Example 2

The second special EP-GIG distribution is based on the setting of $q = 1$ and $\gamma = 3/2$. Since

$$K_{3/2}(u) = \frac{u+1}{u} K_{1/2}(u) = \frac{u+1}{u} \frac{(\pi/2)^{1/2}}{u^{1/2}} \exp(-u),$$

we obtain that the pdf of $\mathsf{GIG}(\eta|3/2,\beta,\alpha)$ is

$$p(\eta|\alpha,\beta,3/2) = \frac{\alpha^{3/2}}{\sqrt{2\pi}} \frac{\exp(\sqrt{\alpha\beta})}{\sqrt{\alpha\beta}+1} \eta^{\frac{1}{2}} \exp(-(\alpha\eta + \beta\eta^{-1})/2)$$

and that the pdf of $\mathsf{EGIG}(b|\alpha,\beta,3/2,1)$ is

$$p(b) = \frac{\alpha \exp(\sqrt{\alpha\beta})}{4(\sqrt{\alpha\beta}+1)} \exp(-\sqrt{\alpha(\beta+|b|)}). \tag{8}$$

B.3 Example 3

We now consider the case that $q = 1$ and $\gamma = -1/2$. In this case, we have $\mathsf{EGIG}(b|\alpha,\beta,-1/2,1)$ which is a mixture of $L(b|0,\eta)$ with density $\mathsf{GIG}(\eta|-1/2,\beta,\alpha)$. The density of $\mathsf{EGIG}(b|\alpha,\beta,-1/2,1)$ is

$$p(b) = \frac{\beta^{1/2}\exp(\sqrt{\alpha\beta})}{4(\beta+|b|)^{3/2}}(1+\sqrt{\alpha(\beta+|b|)})\exp(-\sqrt{\alpha(\beta+|b|)}).$$

B.4 Example 4

The fourth special EP-GIG distribution is $\mathsf{EGIG}(b|\alpha,\beta,0,2)$; that is, we let $q = 2$ and $\gamma = 0$. In other words, we consider the mixture of the Gaussian distribution $N(b|0,\eta)$ with the hyperbolic distribution $\mathsf{GIG}(\eta|\beta,\alpha,0)$. We now have

$$p(b) = \frac{1}{2K_0(\sqrt{\alpha\beta})\sqrt{\beta+b^2}}\exp(-\sqrt{\alpha(\beta+b^2)}).$$

B.5 Example 5

In the fifth special case we set $q = 2$ and $\gamma = 1$; that is, we consider the mixture of the Gaussian distribution $N(b|0,\eta)$ with the generalized inverse Gaussian $\mathsf{GIG}(\eta|1,\beta,\alpha)$. The density of the corresponding EP-GIG distribution $\mathsf{EGIG}(b|\alpha,\beta,1,2)$ is

$$p(b) = \frac{1}{2K_1(\sqrt{\alpha\beta})\beta^{1/2}}\exp(-\sqrt{\alpha(\beta+b^2)}).$$

B.6 Example 6

The final special case is based on the settings $q = 2$ and $\gamma = -1$. In this case, we have

$$p(b) = \int_0^\infty N(b|0,\eta)\mathsf{GIG}(\eta|-1,\beta,\alpha)d\eta = \frac{(\beta/\alpha)^{1/2}}{2K_1(\sqrt{\alpha\beta})}\frac{1+\sqrt{\alpha(\beta+b^2)}}{\exp(\sqrt{\alpha(\beta+b^2)})}(\beta+b^2)^{-\frac{3}{2}}.$$

B.7 Example 7

We are also interested EP-GIG with $q = 1/2$, that is, a class of bridge scale mixtures. In this and next examples, we present two special cases. First, we set $q = 1/2$ and $\gamma = 3/2$. That is,

$$p(b) = \int_0^\infty \mathsf{EP}\big(b|0,\eta,1/2\big)\mathsf{GIG}\big(\eta|3/2,\beta,\alpha\big)d\eta = \frac{\alpha^{\frac{3}{2}}\exp(\sqrt{\alpha\beta})}{2^4(1+\sqrt{\alpha\beta})}\frac{\exp(-\sqrt{\alpha(\beta+|b|^{1/2})})}{(\beta+|b|^{\frac{1}{2}})^{\frac{1}{2}}}.$$

B.8 Example 8

In this case we set $q = 1/2$ and $\gamma = 5/2$. We now have

$$p(b) = \int_0^\infty \mathsf{EP}(b|0,\eta,1/2)\mathsf{GIG}(\eta|5/2,\beta,\alpha)d\eta = \frac{\alpha^2\exp(\sqrt{\alpha\beta})}{2^4(3+3\sqrt{\alpha\beta}+\alpha\beta)}\exp\Big(-\sqrt{\alpha(\beta+|b|^{1/2})}\Big).$$

Appendix C. EP-Jeffreys Priors

We first consider the definition of EP-Jeffreys prior, which the mixture of $\mathsf{EP}(b|0,\eta,q)$ with the Jeffreys prior $1/\eta$. It is easily verified that

$$p(b) \propto \int \mathsf{EP}(b|0,\eta,q)\eta^{-1}d\eta = \frac{q}{2}|b|^{-1}$$

and that $[\eta|b] \sim \mathsf{IG}(\eta|1/q, |b|^q/2)$. In this case, we obtain

$$E(\eta^{-1}|b) = \frac{1}{2q}|b|^{-q}.$$

On the other hand, the EP-Jeffreys prior induces penalty $\log|b|$ for b. Moreover, it is immediately calculated that

$$\frac{d\log|b|}{|b|^q} \triangleq \frac{1}{q}|b|^{-q} = 2E(\eta^{-1}|b).$$

As we can see, our discussions here present an alternative derivation for the adaptive lasso (Zou, 2006). Moreover, we also obtain the relationship of the adaptive lasso with an EM algorithm.

Using the EP-Jeffreys prior, we in particular define a hierarchical model:

$$\begin{aligned}
[\mathbf{y}|\mathbf{b},\sigma] &\sim N(\mathbf{y}|\mathbf{X}\mathbf{b},\sigma\mathbf{I}_n),\\
[b_j|\eta_j,\sigma] &\overset{ind}{\sim} \mathsf{EP}(b_j|0,\sigma\eta_j,q),\\
[\eta_j] &\overset{ind}{\propto} \eta_j^{-1},\\
p(\sigma) &= \text{“}Constant\text{”}.
\end{aligned}$$

It is easy to obtain that

$$[\eta_j|b_j,\sigma] \sim \mathsf{IG}\big(\eta_j\big|1/q,\, \sigma^{-1}|b_j|^q/2\big).$$

Given the tth estimates $(\mathbf{b}^{(t)},\sigma^{(t)})$ of $(\mathbf{b},\sigma)$, the E-step of EM calculates

$$w_j^{(t+1)} \triangleq E(\eta_j^{-1}|b_j^{(t)},\sigma^{(t)}) = \frac{2\sigma^{(t)}}{q|b_j^{(t)}|^q}.$$

The M-step maximizes $Q(\mathbf{b},\sigma|\mathbf{b}^{(t)},\sigma^{(t)})$ with respect to $(\mathbf{b},\sigma)$. In particular, it is obtained as follows:

$$\begin{aligned}
\mathbf{b}^{(t+1)} &= \operatorname*{argmin}_{\mathbf{b}} (\mathbf{y}-\mathbf{X}\mathbf{b})^T(\mathbf{y}-\mathbf{X}\mathbf{b}) + \sum_{j=1}^p w_j^{(t+1)}|b_j|^q,\\
\sigma^{(t+1)} &= \frac{q}{qn+2p}\Big\{(\mathbf{y}-\mathbf{X}\mathbf{b}^{(t+1)})^T(\mathbf{y}-\mathbf{X}\mathbf{b}^{(t+1)}) + \sum_{j=1}^p w_j^{(t+1)}|b_j^{(t+1)}|^q\Big\}.
\end{aligned}$$

Appendix D. The Hierarchy with the Bridge Prior Given in (4)

Using the bridge prior in (4) yields the following hierarchical model:

$$\begin{aligned}
[\mathbf{y}|\mathbf{b},\sigma] &\sim N(\mathbf{y}|\mathbf{X}\mathbf{b},\sigma\mathbf{I}_n),\\
[b_j|\eta_j,\sigma] &\overset{ind}{\sim} L(b_j|0,\sigma\eta_j),\\
[\eta_j] &\overset{ind}{\propto} G(\eta_j|3/2,\alpha/2),\\
p(\sigma) &= \text{“}Constant\text{”}.
\end{aligned}$$

It is easy to obtain that

$$[\eta_j|b_j,\sigma] \sim \mathsf{GIG}\big(\eta_j\big|1/2,\, \sigma^{-1}|b_j|,\alpha\big).$$

Given the tth estimates $(\mathbf{b}^{(t)},\sigma^{(t)})$ of $(\mathbf{b},\sigma)$, the E-step of EM calculates

$$w_j^{(t+1)} \triangleq E(\eta_j^{-1}|b_j^{(t)},\sigma^{(t)}) = \sqrt{\frac{\alpha\sigma^{(t)}}{|b_j^{(t)}|}}.$$

The M-step maximizes $Q(\mathbf{b},\sigma|\mathbf{b}^{(t)},\sigma^{(t)})$ with respect to $(\mathbf{b},\sigma)$. That is,

$$\begin{aligned}
\mathbf{b}^{(t+1)} &= \underset{\mathbf{b}}{\operatorname{argmin}}\ (\mathbf{y}-\mathbf{X}\mathbf{b})^T(\mathbf{y}-\mathbf{X}\mathbf{b}) + \sum_{j=1}^{p} w_j^{(t+1)}|b_j|^q,\\
\sigma^{(t+1)} &= \frac{q}{qn+2p}\Big\{(\mathbf{y}-\mathbf{X}\mathbf{b}^{(t+1)})^T(\mathbf{y}-\mathbf{X}\mathbf{b}^{(t+1)}) + \sum_{j=1}^{p} w_j^{(t+1)}|b_j^{(t+1)}|^q\Big\}.
\end{aligned}$$

References

D. F. Andrews and C. L. Mallows. Scale mixtures of normal distributions. *Journal of the Royal Statistical Society B*, 36:99–102, 1974.

C. Archambeau and F. R. Bach. Sparse probabilistic projections. In *Advances in Neural Information Processing Systems 21*, 2009.

A. Armagan, D. Dunson, and J. Lee. Generalized double Pareto shrinkage. Technical report, Duke University Department of Statistical Science, February 2011.

G. E. P. Box and G. C. Tiao. *Bayesian Inference in Statistical Analysis*. John Willey and Sons, New York, 1992.

L. Breiman and J. Friedman. Predicting multivariate responses in multiple linear regression (with discussion). *Journal of the Royal Statistical Society, B*, 59(1):3–54, 1997.

E. J. Candès, M. B. Wakin, and S. P. Boyd. Enhancing sparsity by reweighted ℓ_1 minimization. *The Journal of Fourier Analysis and Applications*, 14(5):877–905, 2008.

F. Caron and A. Doucet. Sparse Bayesian nonparametric regression. In *Proceedings of the 25th international conference on Machine learning*, page 8895, 2008.

C. M. Carvalho, N. G. Polson, and J. G. Scott. The horseshoe estimator for sparse signals. *Biometrika*, 97:465–480, 2010.

V. Cevher. Learning with compressible priors. In *Advances in Neural Information Processing Systems 22*, pages 261–269, 2009.

R. Chartrand and W. Yin. Iteratively reweighted algorithms for compressive sensing. In *The 33rd IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, 2008.

I. Daubechies, R. Devore, M. Fornasier, and C. S. Güntürk. Iteratively reweighted least squares minimization for sparse recovery. *Communications on Pure and Applied Mathematics*, 63(1): 1–38, 2010.

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.

J. Fan and R. Li. Variable selection via nonconcave penalized likelihood and its Oracle properties. *Journal of the American Statistical Association*, 96:1348–1361, 2001.

W. Feller. *An Introduction to Probability Theory and Its Applications*, volume II. John Wiley & Sons, second edition, 1971.

M. A. T. Figueiredo. Adaptive sparseness for supervised learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(9):1150–1159, 2003.

W. Fu. Penalized regressions: the bridge vs. the lasso. *Journal of Computational and Graphical Statistics*, 7:397–416, 1998.

P. J. Garrigues and B. A. Olshausen. Group sparse coding with a Laplacian scale mixture prior. In *Advances in Neural Information Processing Systems 22*, 2010.

J. E. Griffin and P. J. Brown. Inference with normal-gamma prior distributions in regression problems. *Bayesian Analysis*, 5(1):171–183, 2010a.

J. E. Griffin and P. J. Brown. Bayesian adaptive Lassos with non-convex penalization. Technical report, University of Kent, 2010b.

E. Grosswald. The student t-distribution of any degree of freedom is infinitely divisible. *Zeitschrift für Wahrscheinlichkeitstheorie und verwandte Gebiete*, 36:103–109, 1976.

C. Hans. Bayesian lasso regression. *Biometrika*, 96:835–845, 2009.

D. Hunter and R. Li. Variable selection using MM algorithms. *The Annals of Statistics*, 33(4): 1617–1642, 2005.

B. Jørgensen. *Statistical Properties of the Generalized Inverse Gaussian Distribution*. Lecture Notes in Statistics. Springer, New York, 1982.

H. Kiiveri. A general approach to simultaneous model fitting and variable elimination in response models for biological data with many more variables than observations. *BMC Bioinformatics*, 9: 195, 2008.

M. Kyung, J. Gill, M. Ghosh, and G. Casella. Penalized regression, standard errors, and Bayesian lassos. *Bayesian Analysis*, 5(2):369412, 2010.

K. Lange and J. S. Sinsheimer. Normal/independent distributions and their applications in robust regression. *Journal of Computational and Graphical Statistics*, 2(2):175–198, 1993.

A. Lee, F. Caron, A. Doucet, and C. Holmes. A hierarchical Bayesian framework for constructing sparsity-inducing priors. Technical report, University of Oxford, UK, 2010.

R. Mazumder, J. Friedman, and T. Hastie. SparseNet: Coordinate descent with nonconvex penalties. *Journal of the American Statistical Association*, 106(495):1125–1138, 2011.

T. Park and G. Casella. The Bayesian Lasso. *Journal of the American Statistical Association*, 103 (482):681–686, 2008.

N. G. Polson and J. G. Scott. Shrink globally, act locally: Sparse Bayesian regularization and prediction. In J. M. Bernardo, M. J. Bayarri, J. O. Berger, A. P. Dawid, D. Heckerman, A. F. M. Smith, and M. West, editors, *Bayesian Statistics 9*. Oxford University Press, 2010.

N. G. Polson and J. G. Scott. Sparse Bayes estimation in non-gaussian models via data augmentation. Technical report, University of Texas at Austin, July 2011.

N. G. Polson and J. G. Scott. Local shrinkage rules, Lévy processes, and regularized regression. *Journal of the Royal Statistical Society (Series B)*, 74(2):287–311, 2012.

B. K. Sriperumbudur and G. R. G. Lanckriet. On the convergence of the concave-convex procedure. In *Advances in Neural Information Processing Systems 22*, 2009.

R. Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society, Series B*, 58:267–288, 1996.

M. E. Tipping. Sparse Bayesian learning and the relevance vector machine. *Journal of Machine Learning Research*, 1:211–244, 2001.

M. West. On scale mixtures of normal distributions. *Biometrika*, 74:646–648, 1987.

J. Weston, A. Elisseeff, B. Schö lkopf, and M. Tipping. Use of the zero-norm with linear models and kernel methods. *Journal of Machine Learning Research*, 3:1439–1461, 2003.

D. Wipf and S. Nagarajan. Iterative reweighted ℓ_1 and ℓ_2 methods for finding sparse solutions. *IEEE Journal of Selected Topics in Signal Processing*, 4(2):317–329, 2010.

C. F. J. Wu. On the convergence properties of the EM algorithm. *The Annals of Statistics*, 11: 95–103, 1983.

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, 2007.

H. Zou. The adaptive lasso and its Oracle properties. *Journal of the American Statistical Association*, 101(476):1418–1429, 2006.

H. Zou and R. Li. One-step sparse estimates in nonconcave penalized likelihood models. *The Annals of Statistics*, 36(4):1509–1533, 2008.

Journal of Machine Learning Research 13 (2012) 2063-2067 Submitted 8/11; Revised 3/12; Published 6/12

Pattern for Python

Tom De Smedt TOM.DESMEDT@UA.AC.BE
Walter Daelemans WALTER.DAELEMANS@UA.AC.BE
CLiPS Computational Linguistics Group
University of Antwerp
2000 Antwerp, Belgium

Editor: Cheng Soon Ong

Abstract

Pattern is a package for Python 2.4+ with functionality for web mining (Google + Twitter + Wikipedia, web spider, HTML DOM parser), natural language processing (tagger/chunker, n-gram search, sentiment analysis, WordNet), machine learning (vector space model, *k*-means clustering, Naive Bayes + *k*-NN + SVM classifiers) and network analysis (graph centrality and visualization). It is well documented and bundled with 30+ examples and 350+ unit tests. The source code is licensed under BSD and available from `http://www.clips.ua.ac.be/pages/pattern`.

Keywords: Python, data mining, natural language processing, machine learning, graph networks

1. Introduction

The World Wide Web is an immense collection of linguistic information that has in the last decade gathered attention as a valuable resource for tasks such as machine translation, opinion mining and trend detection, that is, "Web as Corpus" (Kilgarriff and Grefenstette, 2003). This use of the WWW poses a challenge since the Web is interspersed with code (HTML markup) and lacks metadata (language identification, part-of-speech tags, semantic labels).

"Pattern" (BSD license) is a Python package for web mining, natural language processing, machine learning and network analysis, with a focus on ease-of-use. It offers a mash-up of tools often used when harnessing the Web as a corpus, which usually requires several independent toolkits chained together in a practical application. Several such toolkits with a user interface exist in the scientific community, for example ORANGE (Demšar et al., 2004) for machine learning and GEPHI (Bastian et al., 2009) for graph visualization. By contrast, PATTERN is more related to toolkits such as NLTK (Bird et al., 2009), PYBRAIN (Schaul et al., 2010) and NETWORKX (Hagberg et al., 2008), in that it is geared towards integration in the user's own programs. Also, it does not specialize in one domain but provides general cross-domain functionality.

The package aims to be useful to both a scientific and a non-scientific audience. The syntax is straightforward. Function names and parameters were so chosen as to make the commands self-explanatory. The documentation assumes no prior knowledge. We believe that PATTERN is valuable as a learning environment for students, as a rapid development framework for web developers, and in research projects with a short development cycle.

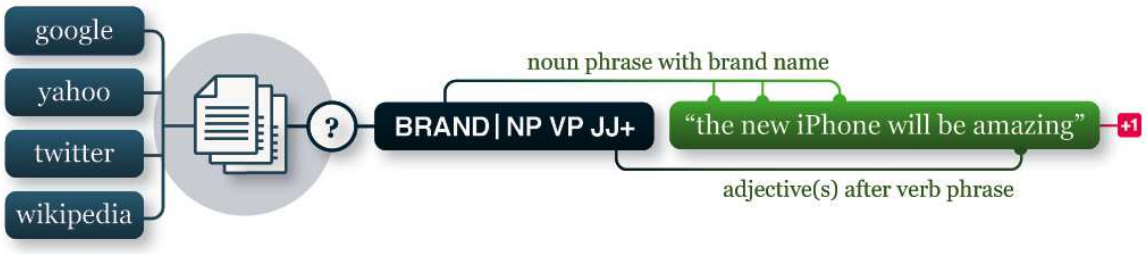


Figure 1: Example workflow. Text is mined from the web and searched by syntax and semantics. Sentiment analysis (positive/negative) is performed on matching phrases.

2. Package Overview

PATTERN is organized in separate modules that can be chained together, as shown in Figure 1. For example, text from Wikipedia (pattern.web) can be parsed for part-of-speech tags (pattern.en), queried by syntax and semantics (pattern.search), and used to train a classifier (pattern.vector).

pattern.web Tools for web data mining, using a download mechanism that supports caching, proxies, asynchronous requests and redirection. A `SearchEngine` class provides a uniform API to multiple web services: Google, Bing, Yahoo!, Twitter, Wikipedia, Flickr and news feeds using FEED PARSER (packages.python.org/feedparser). The module includes an HTML parser based on BEAUTIFUL SOUP (crummy.com/software/beautifulsoup), a PDF parser based on PDFMINER (unixuser.org/ euske/python/pdfminer), a web crawler, and a webmail interface.

pattern.en Fast, regular expressions-based shallow parser for English (identifies sentence constituents, e.g., nouns, verbs), using a finite state part-of-speech tagger (Brill, 1992) extended with a tokenizer, lemmatizer and chunker. Accuracy for Brill's tagger is 95% and up. A parser with higher accuracy (MBSP) can be plugged in. The module has a `Sentence` class for parse tree traversal, functions for singularization/pluralization (Conway, 1998), conjugation, modality and sentiment analysis. It comes bundled with WORDNET3 (Fellbaum, 1998) and PYWORDNET.

pattern.nl Lightweight implementation of pattern.en for Dutch, using the BRILL-NL language model (Geertzen, 2010). Contributors are encouraged to read the developer documentation on how to add support for other languages.

pattern.search N-gram pattern matching algorithm for `Sentence` objects. The algorithm uses an approach similar to regular expressions. Search queries can include a mixture of words, phrases, part-of-speech-tags, taxonomy terms (e.g., `pet` = dog, cat or goldfish) and control characters (e.g., `+` = multiple, `*` = any, `()` = optional) to extract relevant information.

pattern.vector Vector space model using a `Document` and a `Corpus` class. Documents are lemmatized bag-of-words that can be grouped in a sparse corpus to compute TF-IDF, distance metrics (cosine, Euclidean, Manhattan, Hamming) and dimension reduction (Latent Semantic Analysis). The module includes a hierarchical and a k-means clustering algorithm, optimized with the k-means++ initialization algorithm (Arthur and Vassilvitskii, 2007) and triangle inequality (Elkan, 2003). A Naive Bayes, a k-NN, and a SVM classifier using LIBSVM (Chang and Li, 2011) are included, with tools for feature selection (information gain) and K-fold cross validation.

pattern.graph Graph data structure using `Node`, `Edge` and `Graph` classes, useful (for example) for modeling semantic networks. The module has algorithms for shortest path finding, subgraph partitioning, eigenvector centrality and betweenness centrality (Brandes, 2001). Centrality algorithms were ported from NETWORKX. The module has a force-based layout algorithm that positions nodes in 2D space. Visualizations can be exported to HTML and manipulated in a browser (using our `canvas.js` helper module for the HTML5 Canvas2D element).

pattern.metrics Descriptive statistics functions. Evaluation metrics including a code profiler, functions for accuracy, precision and recall, confusion matrix, inter-rater agreement (Fleiss' kappa), string similarity (Levenshtein, Dice) and readability (Flesch).

pattern.db Wrappers for CSV files and SQLITE and MYSQL databases.

3. Example Script

As an example, we chain together four PATTERN modules to train a *k*-NN classifier on adjectives mined from Twitter. First, we mine 1,500 tweets with the hashtag #win or #fail (our classes), for example: "$20 tip off a *sweet little old* lady today #win". We parse the part-of-speech tags for each tweet, keeping adjectives. We group the adjective vectors in a corpus and use it to train the classifier. It predicts "sweet" as `WIN` and "stupid" as `FAIL`. The results may vary depending on what is currently buzzing on Twitter.

The source code is shown in Figure 2. Its size is representative for many real-world scenarios, although a real-world classifier may need more training data and more rigorous feature selection.

```
from pattern.web    import Twitter
from pattern.en     import Sentence, parse
from pattern.search import search
from pattern.vector import Document, Corpus, KNN

corpus = Corpus()
for i in range(1,15):
    for tweet in Twitter().search('#win OR #fail', start=i, count=100):
        p = '#win' in tweet.description.lower() and 'WIN' or 'FAIL'
        s = tweet.description.lower()
        s = Sentence(parse(s))
        s = search('JJ', s) # JJ = adjective
        s = [match[0].string for match in s]
        s = ' '.join(s)
        if len(s) > 0:
        corpus.append(Document(s, type=p))

classifier = KNN()
for document in corpus:
    classifier.train(document)
print classifier.classify('sweet') # yields 'WIN'
print classifier.classify('stupid') # yields 'FAIL'd
```

Figure 2: Example source code for a *k*-NN classifier trained on Twitter messages.

4. Case Study

As a case study, we used PATTERN to create a Dutch sentiment lexicon (De Smedt and Daelemans, 2012). We mined online Dutch book reviews and extracted the 1,000 most frequent adjectives. These were manually annotated with positivity, negativity, and subjectivity scores. We then enlarged the lexicon using distributional expansion. From the TWNC corpus (Ordelman et al., 2007) we extracted the most frequent nouns and the adjectives preceding those nouns. This results in a vector space with approximately 5,750 adjective vectors with nouns as features. For each annotated adjective we then computed *k*-NN and inherited its scores to neighbor adjectives. The lexicon is bundled into PATTERN 2.3.

5. Documentation

PATTERN comes bundled with examples and unit tests. The documentation contains a quick overview, installation instructions, and for each module a detailed page with the API reference, examples of use and a discussion of the scientific principles. The documentation assumes no prior knowledge, except for a background in Python programming. The unit test suite includes a set of corpora for testing accuracy, for example POLARITY DATA SET V2.0 (Pang and Lee, 2004).

6. Source Code

PATTERN is written in pure Python, meaning that we sacrifice performance for development speed and readability (i.e., slow clustering algorithms). The package runs on all platforms and has no dependencies, with the exception of NumPy when LSA is used. The source code is annotated with developer comments. It is hosted online on GitHub (github.com) using the Git revision control system. Contributions are welcomed.

The source code is released under a BSD license, so it can be incorporated into proprietary products or used in combination with other open source packages such as SCRAPY (web mining), NLTK (natural language processing), PYBRAIN and PYML (machine learning) and NETWORKX (network analysis). We provide an interface to MBSP FOR PYTHON (De Smedt et al., 2010), a robust, memory-based shallow parser built on the TIMBL machine learning software. The API's for the PATTERN parser and MBSP are identical.

Acknowledgments

Development was funded by the Industrial Research Fund (IOF) of the University of Antwerp.

References

David Arthur and Sergei Vassilvitskii. k-means++: the advantages of careful seeding. *Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1027–1035, 2007.

Mathieu Bastian, Sebastien Heymann, and Mathieu Jacomy. Gephi: An open source software for exploring and manipulating networks. *Proceedings of the Third International ICWSM Conference*, 2009.

Steven Bird, Ewan Klein, and Edward Loper. *Natural Language Processing with Python*. O'Reilly Media, 2009.

Ulrik Brandes. A faster algorithm for betweenness centrality. *The Journal of Mathematical Sociology*, 25(2):163–177, 2001.

Eric Brill. A simple rule-based part of speech tagger. *Proceedings of the Third Conference on Applied Natural Language Processing*, pages 152–155, 1992.

Chih-Chung Chang and Chih-Jen Li. LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2(3), 2011.

Damian Conway. An algorithmic approach to english pluralization. *Proceedings of the Second Annual Perl Conference*, 1998.

Tom De Smedt and Walter Daelemans. Vreselijk mooi! (terribly beautiful): A subjectivity lexicon for dutch adjectives. *Proceedings of the 8th Language Resources and Evaluation Conference (LREC'12)*, pages 3568—-3572, 2012.

Tom De Smedt, Vincent Van Asch, and Walter Daelemans. Memory-based shallow parser for python. *CLiPS Technical Report Series*, 2, 2010.

Janez Demšar, Blaž Zupan, Gregor Leban, and Tomaz Curk. Orange: From experimental machine learning to interactive data mining. *Knowledge Discovery in Databases*, 3202:537–539, 2004.

Charles Elkan. Using the triangle inequality to accelerate k-means. *Proceedings of the Twentieth International Conference on Machine Learning*, pages 147–153, 2003.

Christiane Fellbaum. *WordNet: An Electronic Lexical Database*. MIT Press, Cambridge, 1998.

Jeroen Geertzen. Jeroen geertzen :: software & demos : Brill-nl, June 2010. URL `http://cosmion.net/jeroen/software/brill\_pos/`.

Aric Hagberg, Daniel Schult, and Pieter Swart. Exploring network structure, dynamics, and function using networkx. *Proceedings of the 7th Python in Science Conference*, pages 11–15, 2008.

Adam Kilgarriff and Gregory Grefenstette. Introduction to the special issue on the web as corpus. *Computational Linguistics*, 29(3):333–347, 2003.

Roeland Ordelman, Franciska de Jong, Arjan van Hessen, and Hendri Hondorp. TwNC: A multifaceted dutch news corpus. *ELRA Newsletter*, 12:3–4, 2007.

Bo Pang and Lillian Lee. A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts. *Proceedings of the ACL*, pages 271–278, 2004.

Tom Schaul, Justin Bayer, Daan Wierstra, Yi Sun, Martin Felder, Frank Sehnke, Thomas Rückstieß, and Jürgen Schmidhuber. Pybrain. *Journal of Machine Learning Research*, pages 743–746, 2010.

Journal of Machine Learning Research volume (2012) 2069-2106 Submitted 7/11; Revised 5/12; Published 6/12

Optimistic Bayesian Sampling in Contextual-Bandit Problems

Benedict C. May BEN.MAY@BRIS.AC.UK
School of Mathematics
University of Bristol
Bristol, BS8 1TW, United Kingdom

Nathan Korda KORDA@MATHS.OX.AC.UK
Anthony Lee LEE@STATS.OX.AC.UK
Oxford-Man Institute
University of Oxford
Eagle House, Walton Well Road
Oxford, OX2 6ED, United Kingdom

David S. Leslie DAVID.LESLIE@BRIS.AC.UK
School of Mathematics
University of Bristol
Bristol, BS8 1TW, United Kingdom

Editor: Nicolò Cesa-Bianchi

Abstract

In sequential decision problems in an unknown environment, the decision maker often faces a dilemma over whether to explore to discover more about the environment, or to exploit current knowledge. We address the exploration-exploitation dilemma in a general setting encompassing both standard and contextualised bandit problems. The contextual bandit problem has recently resurfaced in attempts to maximise click-through rates in web based applications, a task with significant commercial interest.

In this article we consider an approach of Thompson (1933) which makes use of samples from the posterior distributions for the instantaneous value of each action. We extend the approach by introducing a new algorithm, Optimistic Bayesian Sampling (OBS), in which the probability of playing an action increases with the uncertainty in the estimate of the action value. This results in better directed exploratory behaviour.

We prove that, under unrestrictive assumptions, both approaches result in optimal behaviour with respect to the average reward criterion of Yang and Zhu (2002). We implement OBS and measure its performance in simulated Bernoulli bandit and linear regression domains, and also when tested with the task of personalised news article recommendation on a Yahoo! Front Page Today Module data set. We find that OBS performs competitively when compared to recently proposed benchmark algorithms and outperforms Thompson's method throughout.

Keywords: multi-armed bandits, contextual bandits, exploration-exploitation, sequential allocation, Thompson sampling

1. Introduction

In sequential decision problems in an unknown environment, the decision maker often faces a dilemma over whether to explore to discover more about the environment, or to exploit current knowledge. We address this exploration-exploitation dilemma in a general setting encompass-

ing both standard bandit problems (Gittins, 1979; Sutton and Barto, 1998; Auer et al., 2002) and contextual-bandit problems (Graepel et al., 2010; Li et al., 2010; Auer, 2002; Yang and Zhu, 2002). This dilemma has traditionally been solved using either ad hoc approaches like ε-greedy or softmax action selection (Sutton and Barto, 1998, Chapter 2) or computationally demanding lookahead approaches such as Gittins indices (Gittins, 1979) which provably satisfy an optimality criterion with respect to cumulative discounted reward. However, the lookahead approaches become intractable in all but the simplest settings and the ad hoc approaches are generally perceived to over-explore, despite providing provably optimal long term average reward.

In recent years, Upper Confidence Bound (UCB) methods have become popular (Lai and Robbins, 1985; Kaelbling, 1994; Agrawal, 1995; Auer et al., 2002), due to their low computational cost, ease of implementation and provable optimality with respect to the rate of regret accumulation.

In this article we consider an approach of Thompson (1933) which uses posterior distributions for the instantaneous value of each action to determine a probability distribution over the available actions. Thompson considered only Bernoulli bandits, but in general the approach is to sample a value from the posterior distribution of the expected reward of each action, then select the action with the highest sample from the posterior. Since in our generalised bandit setting the samples are conditioned on the regressor, we label this technique as Local Thompson Sampling (LTS). The technique is used by Microsoft in selecting adverts to display during web searches (Graepel et al., 2010), although no theoretical analysis of Thompson sampling in contextual bandit problems has been carried out.

When these posterior samples are represented as a sum of exploitative value and exploratory value, it becomes clear that LTS results in potentially negative exploratory values. This motivates a new algorithm, Optimistic Bayesian Sampling (OBS), which is based on the LTS algorithm, which is modified by replacing negative exploratory value with a zero value.

We prove that, under unrestrictive assumptions, both approaches result in optimal behaviour in the long term consistency sense described by Yang and Zhu (2002). These proofs use elementary and coupling techniques.

We also implement LTS and OBS and measure their performance in simulated Bernoulli bandit and linear regression domains, and also when tested with the task of personalised news article recommendation on the the Yahoo! Front Page Today Module User Click Log Data Set (Yahoo! Academic Relations, 2011). We find that LTS displays competitive performance, a view shared by Chapelle and Li (2011), and also that OBS outperforms LTS throughout.

1.1 Problem Formulation

An agent is faced with a contextual bandit problem as considered by Yang and Zhu (2002). The process runs for an infinite sequence of time steps, $t \in \mathcal{T} = \{1,2,\ldots\}$. At each time step, t, a regressor, $x_t \in \mathcal{X}$, is observed. An action choice, $a_t \in \mathcal{A}$, $\mathcal{A} = \{1,\ldots,A\}, A < \infty$, is made and a reward $r_t \in \mathbb{R}$ is received.

The contextual bandit framework considered assumes that reward can be expressed as

$$r_t = f_{a_t}(x_t) + z_{t,a_t}$$

where the $z_{t,a}$ are zero mean random variables with unknown distributions and $f_a : \mathcal{X} \to \mathbb{R}$ is an unknown continuous function of the regressor specific to action a. The stream of regressors x_t is assumed not to be influenced by the actions or the rewards, and for simplicity we assume that these

are drawn independently from some fixed distribution on $\mathcal{X}$.[1] For our actions to be comparable, we assume that $\forall a \in \mathcal{A}, \forall t \in \mathcal{T}, \forall x \in \mathcal{X}$, $f_a(x) + z_{t,a}$ is supported on the same set, $\mathcal{S}$. Furthermore to avoid boundary cases we assume that $\forall a \in \mathcal{A}$

$$\sup_{x \in \mathcal{X}} f_a(x) < \sup \mathcal{S}. \tag{1}$$

In situations where the $z_{t,a}$ have unbounded support, $\mathcal{S} = \mathbb{R}$, and (1) is vacuous if $\mathcal{X}$ is compact. The condition is meaningful in situations where $\mathcal{S}$ is compact, such as if rewards are in $\{0,1\}$.

Definition 1 *The optimal expected reward function, $f^* : \mathcal{X} \to \mathbb{R}$, is defined by*

$$f^*(x) = \max_{a \in \mathcal{A}} f_a(x).$$

A minimal requirement for any sensible bandit algorithm is the average reward convergence criterion of Yang and Zhu (2002), which identifies whether a sequence of actions receives, asymptotically, rewards that achieve this optimal expected reward. Hence the main theoretical aim in this article is to prove under mild assumptions that LTS and OBS constructs a sequence of actions such that

$$\frac{\sum_{s=1}^{t} f_{a_s}(x_s)}{\sum_{s=1}^{t} f^*(x_s)} \xrightarrow{\text{a.s.}} 1 \text{ as } t \to \infty. \tag{2}$$

The choice of action a_t is based on the current and past regressors, $\{x_1, \ldots, x_t\}$, past action choices, $\{a_1, \ldots, a_{t-1}\}$, and past rewards, $\{r_1, \ldots, r_{t-1}\}$. Denote $\tilde{I}_1 = \emptyset$ and, for all times $\{t \in \mathcal{T} : t \geq 2\}$, denote

$$\tilde{I}_t = (x_1, \ldots, x_{t-1}, r_1, \ldots, r_{t-1}, a_1, \ldots, a_{t-1}).$$

Furthermore denote all of the prior information available as I_0 and also all the information available at time t as I_t $(= I_0 \cup \tilde{I}_t)$.

Definition 2 *The policy, $\left(\pi_t(\cdot)\right)_{t \in \mathcal{T}}$, is a sequence of conditional probability mass functions where $\pi_t(a) = \mathbb{P}(a_t = a | I_t, x_t)$. At each time step t, the policy maps I_t and x_t to a probability mass function giving the probability of each action being selected.*

The policy is constructed in advance of the process, using only I_0, and is the function used to map I_t and x_t to action selection probabilities for each of the actions.

Note also that, under a Bayesian approach, the information sets I_t result in posterior distributions for quantities of potential interest. In particular I_0 defines the assumed functional forms of the f_a, and a prior distribution over the assumed space of functions, which is then updated as information is received, resulting in a Bayesian regression procedure for estimating the reward functions f_a, and hence a posterior distribution and expectation of $f_a(x_t)$ conditional on the information set $I_t \cup \{x_t\}$.

We do not however formulate an exact probability model of how regressors are sampled, rewards are drawn and inference is carried out. Instead we rely on Assumptions 1–5 placed on the Bayesian regression framework, given in Section 3, that will be satisfied by standard models for the x_t, r_t and prior information I_0. In particular, randomness resulting from the regressor and reward sequences are controlled through these assumptions, whereas our proofs control the randomness due to the

1. Note that this assumption of iid sampling from $\mathcal{X}$ is only used in the latter part of the proof of Theorem 1. In fact an ergodicity condition on the convergence of sample averages would suffice, but would increase the notational complexity of the proofs.

action selection method. A useful framework to keep in mind is one in which regressors are drawn independently from a distribution on a compact Euclidean space $\mathcal{X}$, each $z_{t,a}$ is a Gaussian random variable independent of all other random variables, and the prior information I_0 includes that each f_a is a linear function, and a prior distribution over the parameters of these functions; we revisit this model in Section 4.2 to demonstrate how this framework does indeed ensure that all the Assumptions are satisfied. However much more general frameworks will also result in our Assumptions being satisfied, and restricting to a particular probability model at this point will unnecessarily restrict the analysis.

1.2 Algorithm Motivation

The choice of algorithm presented in this article is motivated by both infinite and finite time considerations. The first subsection of this section describes desirable infinite time properties for an algorithm that are of importance in proving optimality condition (2). The second subsection describes, in a heuristic manner, desirable finite time properties to help understanding of the motivation behind our choice of algorithm, as opposed to the many other algorithms that also satisfy the infinite time requirements.

1.2.1 INFINITE TIME CONSIDERATIONS

In conventional interpretations of similar problems (Littman, 1996; Singh et al., 2000; Sutton and Barto, 1998), there are two major aspects of generating a policy. The first is developing an evaluation scheme and the second an action selection scheme.

So that the agent can evaluate actions, a regression procedure is used to map the current regressor and the history I_t to value estimates for the actions. Denote the agent's estimated value of action a at time t when regressor x is presented as $\hat{f}_{t,a}(x)$. Since $\hat{f}_{t,a}$ is intended to be an estimate of f_a, it is desirable that the evaluation procedure is consistent, that is, $\forall a \in \mathcal{A}, \forall x \in \mathcal{X}$, $\hat{f}_{t,a}(x) - f_a(x)$ converges in some sense to 0 as $n_{t,a} \to \infty$, where $n_{t,a}$ is the number of times action a has been selected up to time t. Clearly such convergence will depend on the sequence of regressor values presented. However consistency of evaluation is not the focus of this work, so will be assumed where necessary and the evaluation procedure used for all algorithms compared in the numerical experiments in §4 will be the same. The main focus of this work is on the action selection side of the problem.

Once action value estimates are available, the agent must use an action selection scheme to decide which action to play. So that the consistency of estimation is achieved, it is necessary that the action selection ensures that every action is selected infinitely often. In this work, we consider algorithms generating randomised policies as a way of ensuring infinite exploration is achieved.

In addition to consistent evaluation and infinite exploration, it is also necessary to exploit the obtained information. Hence the action selection method should be greedy in the limit, that is, the policy π_t is designed such that

$$\sum_{a \in \operatorname{argmax}_{a \in \mathcal{A}} \hat{f}_{t,a}(x_t)} \pi_t(a) \to 1 \text{ as } t \to \infty.$$

These considerations result in the consideration of GLIE (greedy in the limit with infinite exploration) policies, for which action selection is greedy in the limit and also guarantees infinite

exploration (Singh et al., 2000). We combine a GLIE policy with consistent evaluation to achieve criterion (2).

1.2.2 FINITE TIME CONSIDERATIONS

As well as convergence criterion (2), our choice of algorithm is also motivated by informal finite time considerations, since many algorithms for which (2) holds are perceived to explore more than is desirable. We note that formal optimality criteria are available, such as expected cumulative discounted reward (Gittins, 1979) and rate of regret accumulation (Auer et al., 2002). However an analysis of Thompson sampling under these criteria has proved elusive, and our heuristic approach inspires a modification of Thompson sampling which compares favourably in numerical experiments (see Section 4). In this section, we discuss the short term heuristics.

In particular, consider the methodology of evaluating both an exploitative value estimate and an 'exploratory bonus' at each time step for each action, and then acting greedily based on the sums of exploitative and exploratory values (Meuleau and Bourgine, 1999). An action's exploitative value estimate corresponds to the expected immediate reward (i.e., expected reward for the current timestep) from selecting the action, given information obtained so far, and therefore the posterior expectation of expected immediate reward is the appropriate exploitative action value estimate.

Definition 3 *Let $p_a(\cdot \mid I_t, x_t)$ denote the posterior distribution of $f_a(x_t)$ given I_t and x_t, and let $Q^{Th}_{t,a}$ be a random variable with distribution $p_a(\cdot \mid I_t, x_t)$. The exploitative value, $\hat{f}_{t,a}(x_t)$, of action a at time t is defined by*

$$\hat{f}_{t,a}(x_t) = \mathbb{E}(Q^{Th}_{t,a} \mid I_t, x_t).$$

Thompson (1933) suggests selecting action a_t with probability equal to the probability that a_t is optimal, given I_t (there is no regressor in Thompson's framework). This principle has recently been used by Graepel et al. (2010), who implement the scheme by sampling, for each a, $Q^{\mathrm{Th}}_{t,a}$ from the posterior distribution $p_a(\cdot \mid I_t, x_t)$ and selecting an action that maximises $Q^{\mathrm{Th}}_{t,a}$. This corresponds to using an exploratory value $\tilde{f}^{\mathrm{Th}}_{t,a}(x_t) := Q^{\mathrm{Th}}_{t,a} - \hat{f}_{t,a}(x_t)$ which is sampled from the posterior distribution of the error in the exploitative action value estimate at the current regressor. We name this scheme Local Thompson Sampling (LTS), where 'local' makes reference to the fact that action selection probabilities are the probabilities that each action is optimal at the current regressor. Under mild assumptions on the posterior expectation and error distribution approximations used, one can show that Local Thompson Sampling guarantees that convergence criterion (2) holds (see Theorem 1).

However the exploratory value $\tilde{f}^{\mathrm{Th}}_{t,a}(x_t)$ under LTS has zero conditional expectation given I_t and x_t (by Definition 3) and can take negative values. Both of these properties are undesirable if one assumes that information is useful for the future. One consequence of this is that, in regular situations, the probability of selecting an action $\hat{a}^*_t \in \operatorname{argmax}_{a \in \mathcal{A}} \hat{f}_{t,a}(x_t)$ decreases as the posterior variance of $f_{\hat{a}^*_t}(x_t) - \hat{f}_{t,\hat{a}^*_t}(x_t)$ increases, that is, if the estimate for an action with the highest exploitative value has a lot of uncertainty then it is less likely to be played than if the estimate had little uncertainty.

To counteract this feature of LTS, we introduce a new procedure, Optimistic Bayesian Sampling (OBS) in which the exploratory value is given by

$$\tilde{f}_{t,a}(x_t) = \max(0, \tilde{f}^{\mathrm{Th}}_{t,a}(x_t) - \hat{f}_{t,a}(x_t)).$$

This exploratory value has positive conditional expectation given I_t and x_t and cannot take negative values. The exploratory bonus results in increased selection probabilities for uncertain actions, a

desirable improvement when compared to LTS. In §3, we show that OBS satisfies the convergence criterion (2) under mild assumptions. Furthermore, simulations described in §4 indicate that the OBS algorithm does indeed outperform LTS, confirming the intuition above.

1.3 Related Work

There are three broad classes of exploration approach: undirected, myopic and belief-lookahead (Asmuth et al., 2009). In undirected exploration, the action selection distribution depends only on the values of the exploitative action value estimates. Examples of undirected exploration include ε-greedy and softmax action selection (see Chapter 2 of Sutton and Barto, 1998). In general, the short term performance of undirected methods is restricted by the fact that estimate uncertainty is not considered.

At the other end of the spectrum, in belief-lookahead methods, such as those suggested by Gittins (1979), a fully Bayesian approach is incorporated in which the action yielding the highest expected cumulative reward over the remainder of the process is selected,[2] thereby considering exploitative and exploratory value both directly and simultaneously and providing the optimal decision rule according to the specific criterion of maximising expected cumulative discounted reward. According to Wang et al. (2005),"in all but trivial circumstances, there is no hope of exactly following an optimal action selection strategy". Furthermore, even when it is possible to evaluate the optimal decision rule, "the optimal solutions are typically hard to compute, rely on artificial discount factors and fail to generalise to realistic reward distributions" (Scott, 2010). There is also the issue of 'incomplete learning'; Brezzi and Lai (2000) showed that, for standard bandit problems, Gittins' index rule samples only one action infinitely often and that this action is sub-optimal with positive probability. If the modelling assumptions and posterior approximations used are accurate, then this is a price worth paying in order to maximise expected cumulative discounted reward. However, if the posterior approximation method admits a significant error, then it may be that a too heavy reliance is placed on early observations. For these reasons, Gittins-type rules are rarely useful in practice.

In myopic methods, the uncertainty of action value estimates is taken into account, although the impact of action selections on future rewards is not considered directly. The exploratory component of myopic methods aims to reduce the uncertainty at the current regressor without explicitly considering future reward. By reducing uncertainty at each point presented as a regressor, uncertainty is reduced globally 'in the right places' without considering the regressor distribution. Myopic action selection can be efficient, easy to implement and computationally cheap. The LTS and OBS methods presented in this paper are myopic methods. The other main class of myopic methods are the upper confidence bound methods, which are now popular in standard and contextual bandit applications, and in some settings can be proved to satisfy an optimality criterion with respect to the rate of accumulation of regret (for an overview, and definitions of various notions of regret, see Cesa-Bianchi and Lugosi, 2006).

Inspired by the work of Lai and Robbins (1985) and Agrawal (1995), Auer et al. (2002) proposed a myopic algorithm, UCB1, for application in standard bandit problems. The exploratory value at time t for action a, which we denote $\tilde{f}_{t,a}$, takes the simple form

$$\tilde{f}_{t,a} = \sqrt{\frac{2\log(t-1)}{n_{t,a}}}.$$

2. Note that this is only meaningful in the case of discounted rewards or if the time sequence is finite.

Infinite exploration is guaranteed by the method, since the exploratory value grows in periods in which the associated action is not selected. Moreover, Auer et al. (2002) prove that the expected finite-time regret is logarithmically bounded for bounded reward distributions, matching the (asymptotically) optimal rate derived by Lai and Robbins (1985) uniformly over time. Auer et al. (2002) also propose a variant of UCB1, named UCB-Tuned, which incorporates estimates of the reward variances, and show it to outperform UCB1 in simulations, although no theoretical results are given for the variant.

Two recently-proposed variants of the UCB1 algorithm are the MOSS (Minimax Optimal Strategy in the Stochastic case) algorithm (Audibert and Bubeck, 2010) and the UCB-V algorithm (Audibert and Bubeck, 2009). The MOSS algorithm is defined for finite problems with known horizon $|\mathcal{T}|$, but the 'doubling trick' described in §2.3 of Cesa-Bianchi and Lugosi (2006) can be used if the horizon is not known. MOSS differs from UCB1 by replacing the $\log(t-1)$ term in the exploratory value with $\log\left(\frac{|\mathcal{T}|}{|\mathcal{A}|n_{t,a}}\right)$ and hence selecting intensively drawn actions less often. The UCB-V algorithm incorporates estimates of reward variance in a similar way to the UCB-Tuned algorithm. The UCB-Tuned, MOSS and UCB-V algorithms provide suitable benchmarks for comparison in Bernoulli bandit problems.

Another class of 'UCB-type' algorithms was proposed initially by Lai and Robbins (1985), with a recent theoretical analysis by Garivier and Cappé (2011). The evaluation of action values involves constrained maximisation of Kullback-Leibler divergences. The primary purpose of the KL-UCB algorithm is to address the non-parametric problem although parametric implementation is discussed and optimal asymptotic regret bounds are proven for Bernoulli rewards. In the parametric case, a total action value corresponds to the highest posterior mean associated with a posterior distribution that has KL divergence less than a pre-defined term increasing logarithmically with time. A variant of KL-UCB, named KL-UCB+ is also proposed by Garivier and Cappé (2011) and is shown to outperform KL-UCB (with respect to expected regret) in simulated Bernoulli reward problems. Both algorithms also serve as suitable benchmarks for comparison in Bernoulli bandit problems.

For contextual bandit problems, Interval estimation (IE) methods, such as those suggested by Kaelbling (1994), Pavlidis et al. (2008) and Li et al. (2010) (under the name LinUCB), have become popular. They are UCB-type methods in which actions are selected greedily based on the upper bound of a confidence interval for the exploitative value estimate at a fixed significance level. The exploratory value used in IE methods is the difference between the upper bound and the exploitative value estimate. The width of the confidence interval at a particular point in the regressor space is expected to decrease the more times the action is selected.

There are numerous finite-time analyses of the contextual bandit problem. The case of linear expected reward functions provides the simplest contextual setting and examples of finite-time analyses include those of the SupLinRel and SupLinUCB algorithms by Auer (2002) and Chu et al. (2011) respectively, in which high probability regret bounds are established. The case of generalised linear expected rewards is considered by Filippi et al. (2010), proving high probability regret bounds for the GLM-UCB algorithm. Slivkins (2011) provides an example of finite-time analysis of contextual bandits in a more general setting, in which a regret bound is proved for the Contextual Zooming algorithm under the assumptions that the joint regressor and action space is a compact metric space and the reward functions are Lipschitz continuous over the aforementioned space.

On the other hand, very little is known about the theoretical properties of Thompson sampling. The only theoretical studies of Thompson sampling that we are aware of are by Granmo (2008)

and Agrawal and Goyal (2011). The former work considers only the two-armed non-contextual Bernoulli bandit and proves that Thompson sampling (the Bayesian Learning Automaton, in their terminology) converges to only pulling the optimal action with probability one. The latter work considers the K-armed non-contextual Bernoulli bandit and proves an optimal rate of regret (uniformly through time) for Thompson sampling. In this work, we focus on proving convergence criterion (2) for the LTS and OBS algorithms in a general contextual bandit setting in §3 and perform numerical experiments in §4 to illustrate the finite time properties of the algorithms.

2. Algorithms

In this section, we describe explicitly how the action selection is carried out at each decision instant for both the LTS and the OBS algorithms.

At each time t, the LTS algorithm requires a mechanism that can, for each action $a \in \mathcal{A}$, be used to sample from the posterior distribution of $f_a(x_t)$ given regressor x_t and information set I_t. Recall that the density of this distribution is denoted as $p_a(\cdot|\, I_t, x_t)$ and a random variable from the distribution as $Q_{t,a}^{\mathrm{Th}}$.

Algorithm 1 Local Thompson Sampling (LTS)

Input: Posterior distributions $\{p_a(\cdot|\, I_t, x_t) : a \in \mathcal{A}\}$
for $a = 1$ **to** A **do**
  Sample $Q_{t,a}^{\mathrm{Th}} \sim p_a(\cdot|\, I_t, x_t)$
end for
Sample a_t uniformly from $\operatorname{argmax}_{a\in\mathcal{A}} Q_{t,a}^{\mathrm{Th}}$

As in the case of the LTS algorithm, at each time t, the OBS algorithm requires a mechanism that can, for each action $a \in \mathcal{A}$, be used to sample from the posterior distribution of $f_a(x_t)$ given regressor x_t and information set I_t. Additionally, the OBS algorithm requires a mechanism for evaluating exploitative value $\hat{f}_{t,a}(x_t)$, where exploitative value is taken to be the posterior expectation of $f_a(x_t)$ given I_t and x_t.

Algorithm 2 Optimistic Bayesian Sampling (OBS)

Input: Posterior distributions $\{p_a(\cdot|\, I_t, x_t) : a \in \mathcal{A}\}$
for $a = 1$ **to** A **do**
  Sample $Q_{t,a}^{\mathrm{Th}} \sim p_a(\cdot|\, I_t, x_t)$
  Evaluate $\hat{f}_{t,a}(x_t) = \mathbb{E}(Q_{t,a}^{\mathrm{Th}}|\, I_t, x_t)$
  Set $Q_{t,a} = \max(Q_{t,a}^{\mathrm{Th}}, \hat{f}_{t,a}(x_t))$
end for
Sample a_t uniformly from $\operatorname{argmax}_{a\in\mathcal{A}} Q_{t,a}$

3. Analysis

Theoretical properties of the LTS and OBS algorithms are analysed in this section. In particular, we focus on proving convergence in the sense of (2) under mild assumptions on the posterior distributions and expectations used. Regret analysis would provide useful insight into the finite time

properties of the LTS and OBS algorithms. However, we consider the problem in a general setting and impose only weak constraints on the nature of the posterior distributions used to sample action values, making the type of regret analysis common for UCB methods difficult, but allowing the convergence result to hold for a wide class of bandit settings and posterior approximations.

3.1 LTS Algorithm Analysis

We begin our convergence analysis by showing that the LTS algorithm explores all actions infinitely often, thus allowing a regression procedure to estimate all the functions f_a. In order to do this we need to make some assumptions.

To guarantee infinite exploration, it is desirable that the posterior distributions, $p_a(\cdot|\cdot,\cdot)$, generating the LTS samples are supported on $(\inf\mathcal{S},\sup\mathcal{S})$, a reasonable assumption in many cases. We make the weaker assumption that each sample can be greater than (or less than) any value in $(\inf\mathcal{S},\sup\mathcal{S})$ with positive probability. For instance, this assumption is satisfied by any distribution supported on $(\inf\mathcal{S},\inf\mathcal{S}+\delta_1)\cup(\sup\mathcal{S}-\delta_2,\sup\mathcal{S})$ for $\delta_1,\delta_2>0$.

It is also desirable that the posterior distributions remain fixed in periods of time in which the associated action is not selected, also a reasonable assumption if inference is independent for different actions. We make the weaker assumption that, in such periods of time, a lower bound exists for the probability that the LTS sample is above (or below) any value in $(\inf\mathcal{S},\sup\mathcal{S})$. Formally, we make the following assumption:

Assumption 1 *Let $a\in\mathcal{A}$ be an arbitrary action, let T be an arbitrary time, let I_T be an arbitrary history to time T, and let $M\in(\inf\mathcal{S},\sup\mathcal{S})$. There exists an $\varepsilon>0$ depending on a, T, I_T and M such that for all $t>T$, all histories*

$$I_t = I_T\cup\{x_T,\ldots,x_{t-1},r_T,\ldots,r_{t-1},a_T,\ldots,a_{t-1}\}$$

such that $a_s\neq a$ for $s\in\{T,\ldots,t-1\}$, and all $x_t\in\mathcal{X}$

$$\mathbb{P}(Q^{Th}_{t,a}>M|I_t,x_t)>\varepsilon$$

and

$$\mathbb{P}(Q^{Th}_{t,a}<M|I_t,x_t)>\varepsilon.$$

Along with Assumption 1, we also assume that the posterior distributions concentrate on functions of the regressor bounded away from $\sup\mathcal{S}$ as their associated actions are selected infinitely often. Formally, we assume that:

Assumption 2 *For each action $a\in\mathcal{A}$, there exist a function $g_a:\mathcal{X}\to(\inf\mathcal{S},\sup\mathcal{S})$ such that*

(i) $\left[Q^{Th}_{t,a}-g_a(x_t)\right]\xrightarrow{\mathbb{P}}0$ *as* $n_{t,a}\to\infty$,

(ii) $\sup_{x\in\mathcal{X}}g_a(x)<\sup\mathcal{S}$.

We do not take $g_a=f_a$ since this allows us to prove infinite exploration even when our regression framework does not support the true functions (e.g., when I_0 supports only linear functions, but the true f_a are actually non-linear functions). Furthermore, the second condition, when combined with Assumption 1, ensures that over periods in which action a is not selected there is a constant lower

bound on the probability that either the LTS or OBS algorithms sample a $Q_{t,a}^{\text{Th}}$ value greater than any $g_{\tilde{a}}(x)$.

Although there is an apparent tension between Assumption 1 and Assumption 2(i), note that Assumption 1 applies to the support of the posterior distributions for periods in which associated actions are not selected, whereas Assumption 2(i) applies to the limits of the posterior distributions as their associated actions are selected infinitely often.

Lemma 2 shows that, if Assumption 1 and 2 hold, then the proposed algorithm does guarantee infinite exploration. The lemma is important as it can be combined with Assumption 2 to imply that, for all $a \in \mathcal{A}$,

$$\left[Q_{t,a}^{\text{Th}} - g_a(x_t)\right] \xrightarrow{\mathbb{P}} 0 \text{ as } t \to \infty$$

since $\forall a \in \mathcal{A}, n_{t,a} \to \infty$ as $t \to \infty$. The proof of Lemma 2 relies on the following lemma (Corollary 5.29 of Breiman, 1992):

Lemma 1 *(Extended Borel-Cantelli Lemma). Let I_t be an increasing sequence of σ-fields and let V_t be I_{t+1}-measurable. Then*

$$\left\{\omega : \sum_{t=0}^{\infty} \mathbb{P}(V_t | I_t) = \infty\right\} = \{\omega : \omega \in V_t \textit{ infinitely often}\}$$

holds with probability 1.

Lemma 2 *If Assumption 1 and 2 hold, then the LTS algorithm exhibits infinite exploration with probability* 1*, that is,*

$$\mathbb{P}\left(\bigcup_{a \in \mathcal{A}} \{n_{t,a} \to \infty \textit{ as } t \to \infty\}\right) = 1.$$

Proof Fix some arbitrary $k \in \{2, \ldots, A\}$. Assume without loss of generality that actions in $\mathcal{A}^{\text{inf}} = \{k, \ldots, A\}$ are selected infinitely often and actions in $\mathcal{A}^{\text{fin}} = \{1, \ldots, k-1\}$ are selected finitely often. By Assumption 2 and the infinite exploration of actions in $\mathcal{A}^{\text{inf}}$, we have that for all actions $a^{\text{inf}} \in \mathcal{A}^{\text{inf}}$ there exists a function $g_{a^{\text{inf}}} : \mathcal{X} \to (\inf \mathcal{S}, \sup \mathcal{S})$ such that

$$\left[Q_{t,a^{\text{inf}}}^{\text{Th}} - g_{a^{\text{inf}}}(x_t)\right] \xrightarrow{\mathbb{P}} 0 \text{ as } t \to \infty.$$

Therefore, for fixed $\delta > 0$, there exists a finite random time, T_δ, that is the earliest time in $\mathcal{T}$ such that for all actions $a^{\text{inf}} \in \mathcal{A}^{\text{inf}}$ we have

$$\mathbb{P}\left(|Q_{t,a^{\text{inf}}}^{\text{Th}} - g_{a^{\text{inf}}}(x_t)| < \delta \middle| I_t, x_t, t > T_\delta\right) > 1 - \delta. \tag{3}$$

Note that, by Assumption 2, we can choose δ to be small enough that such that for all actions $a \in \mathcal{A}$ and regressors $x \in \mathcal{X}$,

$$g_a(x) + \delta < \sup \mathcal{S}. \tag{4}$$

Since all actions in $\mathcal{A}^{\text{fin}}$ are selected finitely often, there exists some finite random time T_f that is the earliest time in $\mathcal{T}$ such that no action in $\mathcal{A}^{\text{fin}}$ is selected after T_f. Let $T = \max\{T_\delta, T_f\}$. From (4) and Assumption 1 we have that for each $a^{\text{fin}} \in \mathcal{A}^{\text{fin}} \backslash 1$ there exists an $\varepsilon_{a^{\text{fin}}} > 0$ such that

$$\mathbb{P}\left(Q_{t,a^{\text{fin}}}^{\text{Th}} < \max_{a \in \mathcal{A}} g_a(x_t) + \delta \middle| I_t, x_t, t > T\right) > \varepsilon_{a^{\text{fin}}}, \tag{5}$$

and also that there exists an $\varepsilon_1 > 0$ such that

$$\mathbb{P}\left(Q_{t,1}^{\mathrm{Th}} > \max_{a \in \mathcal{A}} g_a(x_t) + \delta \middle| I_t, x_t, t > T \right) > \varepsilon_1. \tag{6}$$

Define the events:

$$\begin{aligned}
\overline{G}_{t,a}^{\delta} &= \left\{ Q_{t,a}^{\mathrm{Th}} > \max_{a \in \mathcal{A}} g_a(x_t)) + \delta \right\}, \\
\underline{G}_{t,a}^{\delta} &= \left\{ Q_{t,a}^{\mathrm{Th}} < \max_{a \in \mathcal{A}} g_a(x_t) + \delta \right\}, \\
G_{t,a}^{\delta} &= \left\{ |Q_{t,a}^{\mathrm{Th}} - g_a(x_t)| < \delta \right\}.
\end{aligned}$$

Then the LTS action selection rule implies that

$$\overline{G}_{t,1}^{\delta} \cap \left(\bigcap_{a^{\mathrm{fin}} \in \mathcal{A}^{\mathrm{fin}} \setminus 1} \underline{G}_{t,a^{\mathrm{fin}}}^{\delta} \right) \cap \left(\bigcap_{a^{\mathrm{inf}} \in \mathcal{A}^{\mathrm{inf}}} G_{t,a^{\mathrm{inf}}}^{\delta} \right) \subset \{a_t = a\},$$

so that

$$\mathbb{P}(a_t = 1 | I_t, x_t, T > t) \geq \mathbb{P}\left(\overline{G}_{t,1}^{\delta} \cap \left(\bigcap_{a^{\mathrm{fin}} \in \mathcal{A}^{\mathrm{fin}} \setminus 1} \underline{G}_{t,a^{\mathrm{fin}}}^{\delta} \right) \cap \left(\bigcap_{a^{\mathrm{inf}} \in \mathcal{A}^{\mathrm{inf}}} G_{t,a^{\mathrm{inf}}}^{\delta} \right) \middle| I_t, x_t, t > T \right). \tag{7}$$

The set $\{\overline{G}_{t,1}^{\delta}, \underline{G}_{t,a}^{\delta}, G_{t,b}^{\delta} : a = 2, \ldots, k-1, b = k, \ldots, A\}$ is a conditionally independent set of events given I_t and x_t. Therefore, by (3), (5) and (6), we have

$$\mathbb{P}\left(\overline{G}_{t,1}^{\delta} \cap \left(\bigcap_{a^{\mathrm{fin}} \in \mathcal{A}^{\mathrm{fin}} \setminus 1} \underline{G}_{t,a^{\mathrm{fin}}}^{\delta} \right) \cap \left(\bigcap_{a^{\mathrm{inf}} \in \mathcal{A}^{\mathrm{inf}}} G_{t,a^{\mathrm{inf}}}^{\delta} \right) \middle| I_t, x_t, t > T \right) > \varepsilon^{k-1}(1-\delta)^{A-k+1} \tag{8}$$

where $\varepsilon = \min_{a^{\mathrm{fin}} \in \mathcal{A}^{\mathrm{fin}}} \varepsilon_{a^{\mathrm{fin}}}$. Combining (7) and (8), it follows that

$$\mathbb{P}(a_t = 1 | I_t, x_t, t > T) > \varepsilon^{k-1}(1-\delta)^{A-k+1}$$

so that

$$\begin{aligned}
\sum_{t \in \mathcal{T}} \mathbb{P}(a_t = 1 | I_t, x_t) &\geq \sum_{t=T+1}^{\infty} \mathbb{P}(a_t = 1 | I_t, x_t) \\
&= \sum_{t=T+1}^{\infty} \mathbb{P}(a_t = 1 | I_t, x_t, t > T) \\
&> \sum_{t=T+1}^{\infty} \varepsilon^{k-1}(1-\delta)^{A-k+1} = \infty
\end{aligned}$$

since T is almost surely finite. Hence, by Lemma 1, $\{a_t = 1\}$ occurs infinitely often almost surely, contradicting the assumption that $1 \in \mathcal{A}^{\mathrm{fin}}$. Since action 1 was chosen arbitrarily from the set $\mathcal{A}^{\mathrm{fin}}$,

any action in $\mathcal{A}^{\text{fin}}$ would cause a contradiction. Therefore, $\mathcal{A}^{\text{fin}} = \emptyset$, that is, every action is selected infinitely often almost surely. ■

When we return to the notion of exploitative value estimates $\hat{f}_{t,a}(x_t)$ and hence the concept of a greedy action, then we wish to ascertain whether the algorithm is greedy in the limit. Assumption 2 only implies that the sum of exploitative and exploratory values tends to a particular function of the regressor and not that the exploratory values tend to zero. Although a minor point, the infinite exploration, given by Assumption 1 and 2, needs to be complemented with an assumption that the exploitative value estimates are converging to the same limit as the sampled values $Q_{t,a}^{\text{Th}}$ in order to prove that the policy generated by the LTS algorithm is GLIE. This assumption is not used in proving that the LTS algorithm generates policies satisfying convergence criterion (2) but is used for the equivalent proof for the OBS algorithm (see §3.2).

Assumption 3 *For all actions $a \in \mathcal{A}$*

$$\left[\hat{f}_{t,a}(x_t) - g_a(x_t)\right] \xrightarrow{\mathbb{P}} 0 \text{ as } n_{t,a} \to \infty$$

for g_a defined as in Assumption 2.

Lemma 3 *If Assumptions 1, 2 and 3 hold, then the LTS algorithm policy is GLIE.*

Proof For any $a \in \mathcal{A}$, since

$$\tilde{f}_{t,a}^{\text{Th}} = Q_{t,a}^{\text{Th}} - \hat{f}_{t,a}(x_t),$$

Assumptions 2 and 3 give

$$\tilde{f}_{t,a}^{\text{Th}}(x_t) \xrightarrow{\mathbb{P}} 0 \text{ as } n_{t,a} \to \infty. \tag{9}$$

Since Assumptions 1 and 2 are satisfied, infinite exploration is guaranteed by Lemma 2. This infinite exploration and (9) imply that $\forall a \in \mathcal{A}$

$$\tilde{f}_{t,a}^{\text{Th}}(x_t) \xrightarrow{\mathbb{P}} 0 \text{ as } t \to \infty. \tag{10}$$

Let us denote the set

$$\mathcal{A}_t^* = \operatorname*{argmax}_{a \in \mathcal{A}} \hat{f}_{t,a}(x_t).$$

By splitting value samples into exploitative and exploratory components we have

$$\begin{aligned}
\mathbb{P}\left(a_t \in \mathcal{A}_t^* \middle| I_t, x_t\right) &= \mathbb{P}\left(\max_{a \in \mathcal{A}_t^*} Q_{t,a}^{\text{Th}} > \max_{a \in \mathcal{A} \setminus \mathcal{A}_t^*} Q_{t,a}^{\text{Th}} \middle| I_t, x_t\right) \\
&= \mathbb{P}\left(\max_{a \in \mathcal{A}} \hat{f}_{t,a}(x_t) + \max_{a \in \mathcal{A}_t^*} \tilde{f}_{t,a}^{\text{Th}}(x_t) > \max_{a \in \mathcal{A} \setminus \mathcal{A}_t^*} \left[\hat{f}_{t,a}(x_t) + \tilde{f}_{t,a}^{\text{Th}}(x_t)\right] \middle| I_t, x_t\right) \\
&\geq \mathbb{P}\left(\max_{a \in \mathcal{A}} \hat{f}_{t,a}(x_t) - \max_{a \in \mathcal{A} \setminus \mathcal{A}_t^*} \hat{f}_{t,a}(x_t) > 2 \max_{a \in \mathcal{A}} \left|\tilde{f}_{t,a}^{\text{Th}}(x_t)\right| \middle| I_t, x_t\right) \\
&\xrightarrow{\text{a.s.}} 1 \text{ as } t \to \infty,
\end{aligned}$$

since the right hand side of the last inequality converges in probability to 0 by (10) and

$$\max_{a \in \mathcal{A}} \hat{f}_{t,a}(x_t) > \max_{a \in \mathcal{A} \setminus \mathcal{A}_t^*} \hat{f}_{t,a}(x_t)$$

by definition of $\mathcal{A}_t^*$. Hence, the action selection is greedy in the limit. Lemma 2 ensures infinite exploration, so the policy is GLIE. ■

We have shown we can achieve a GLIE policy even when we do not have consistent regression. However, to ensure the convergence condition (2) is satisfied we need to assume consistency, that is, that the functions g_a (to which the $Q_{t,a}^{\text{Th}}$ converge) are actually the true functions f_a.

Assumption 4 *For all actions $a \in \mathcal{A}$ and regressors $x \in \mathcal{X}$,*

$$g_a(x) = f_a(x).$$

The following Theorem is the main convergence result for the LTS algorithm. Its proof uses the fact that, under the specified assumptions, Lemma 2 implies that, for all actions $a \in \mathcal{A}$,

$$\left[Q_{t,a}^{\text{Th}} - f_a(x_t)\right] \xrightarrow{\mathbb{P}} 0 \text{ as } t \to \infty.$$

We then use a coupling argument (dealing with the dependence in the action selection sequence) to prove that the LTS algorithm policy satisfies convergence criterion (2).

Theorem 1 *If Assumptions 1, 2 and 4 hold, then the LTS algorithm will produce a policy satisfying convergence criterion (2).*

Proof Recall that the optimal expected reward function is defined by $f^*(x) = \max_{a \in \mathcal{A}} f_a(x)$. Fix some arbitrary $\delta > 0$. Denote the event

$$E_t^\delta = \left\{ f^*(x_t) - f_{a_t}(x_t) < 2\delta \right\}$$

so that E_t^δ is the event that true expected reward for the action selected at time t is within 2δ of the optimal expected reward at time t.

The first part of the proof consists of showing that

$$\mathbb{P}(E_t^\delta | I_t, x_t) \xrightarrow{\text{a.s.}} 1 \text{ as } t \to \infty.$$

From Assumptions 2 and 4, and the infinite exploration guaranteed by Lemma 2, $\forall a \in \mathcal{A}$

$$\left[Q_{t,a}^{\text{Th}} - f_a(x_t)\right] \xrightarrow{\mathbb{P}} 0 \text{ as } t \to \infty.$$

Therefore there exists a finite random time, T_δ, that is the earliest time in $\mathcal{T}$ such that $\forall a \in \mathcal{A}$

$$\mathbb{P}\left(|Q_{t,a}^{\text{Th}} - f_a(x_t)| < \delta \middle| I_t, x_t, t > T_\delta \right) > 1 - \delta \tag{11}$$

so that, after T_δ, all sampled $Q_{t,a}^{\text{Th}}$ values are within δ of the true values with high probability.

Define the events

$$F_{t,a}^\delta = \{ |Q_{t,a}^{\text{Th}} - f_a(x_t)| < \delta \}.$$

Then $\{F_{t,a}^{\delta} : a \in \mathcal{A}\}$ is a conditionally independent set of events given I_t and x_t, so that

$$\mathbb{P}\left(\bigcap_{a\in\mathcal{A}} F_{t,a}^{\delta} | I_t, x_t, t > T_{\delta}\right) = \prod_{a\in\mathcal{A}} \mathbb{P}(F_{t,a}^{\delta} | I_t, x_t, t > T_{\delta}) > (1-\delta)^A \tag{12}$$

using inequality (11).

Note that, for any $a_t^* \in \operatorname{argmax}_{a\in\mathcal{A}} f_a(x_t)$,

$$\bigcap_{a\in\mathcal{A}} F_{t,a}^{\delta} \subset \{Q_{t,a_t^*}^{\mathrm{Th}} > f^*(x_t) - \delta\} \tag{13}$$

and, for any $a' \in \{a \in \mathcal{A} : f^*(x_t) - f_a(x_t) > 2\delta\}$,

$$\bigcap_{a\in\mathcal{A}} F_{t,a}^{\delta} \subset \{Q_{t,a'}^{\mathrm{Th}} < f^*(x_t) - \delta\}. \tag{14}$$

Since $\operatorname{argmax}_{a\in\mathcal{A}} f_a(x_t)$ is non-empty and the action selection rule is greedy on the $Q_{t,a}^{\mathrm{Th}}$, statements (13) and (14) give

$$\bigcap_{a\in\mathcal{A}} F_{t,a}^{\delta} \subset \{f^*(x_t) - f_{a_t}(x_t) < 2\delta\} = E_t^{\delta}.$$

and so

$$\mathbb{P}\left(\bigcap_{a\in\mathcal{A}} F_{t,a}^{\delta} | I_t, x_t\right) \leq \mathbb{P}(E_t^{\delta} | I_t, x_t). \tag{15}$$

Inequalities (12) and (15) imply that

$$\mathbb{P}(E_t^{\delta} | I_t, x_t, t > T_{\delta}) > (1-\delta)^A.$$

The condition above holds for arbitrarily small δ so that $\forall x \in \mathcal{X}$

$$\mathbb{P}(E_t^{\delta} | I_t, x_t) \xrightarrow{\text{a.s.}} 1 \text{ as } t \to \infty. \tag{16}$$

This concludes the first part of the proof. We have shown that the probability that the action selected at time t has a true expected reward that is within 2δ of that of the action with the highest true expected reward at time t tends to 1 as $t \to \infty$. We now face the difficulty that the strong law of large numbers cannot be used directly to establish a lower bound on $\lim_{t\to\infty} \frac{1}{t}\sum_{s=1}^{t} f_{a_s}(x_s)$ since the expected reward sequence $\left(f_{a_s}(x_s)\right)_{s\in\mathcal{T}}$ is a sequence of dependent random variables.

The result may be proved using a coupling argument. We will construct an independent sequence of actions b_s that are coupled with a_s, but for which we can apply the strong law of large numbers to $f_{b_s}(x_s)$. By relating the expected reward for playing the b_s sequence to that of the a_s sequence we will show that the a_s sequence satisfies the optimality condition (2).

Fix some arbitrary $\varepsilon > 0$, define the sets

$$\mathcal{A}_t^{\varepsilon} = \{a \in \mathcal{A} : f^*(x_t) - f_a(x_t) < 2\varepsilon\},$$

and let $U_1, U_2, \ldots$ be a sequence of independent and identically distributed $U[0,1]$ random variables. The construction of E_s^{ε} and $\mathcal{A}_s^{\varepsilon}$ implies that $E_s^{\varepsilon} \Leftrightarrow \{a_s \in \mathcal{A}_s^{\varepsilon}\}$. So, by conditioning on the event $\{a_s \in \mathcal{A}_s^{\varepsilon}\}$ and using the LTS action selection rule, it follows that a_s can be expressed as

$$a_s = \begin{cases} \operatorname{argmax}_{a\in\mathcal{A}_s^{\varepsilon}} Q_{s,a}^{\mathrm{Th}} & \text{if } U_s < \mathbb{P}(E_s^{\varepsilon} | I_s, x_s) \\ \operatorname{argmax}_{a\in\mathcal{A}\setminus\mathcal{A}_s^{\varepsilon}} Q_{s,a}^{\mathrm{Th}} & \text{if } U_s > \mathbb{P}(E_s^{\varepsilon} | I_s, x_s) \end{cases}$$

with ties resolved using uniform sampling.

We similarly define b_s based on the U_s as

$$b_s = \begin{cases} \operatorname{argmin}_{a \in \mathcal{A}_s^\varepsilon} f_a(x_s) & \text{if } U_s < 1-\varepsilon \\ \operatorname{argmin}_{a \in \mathcal{A}} f_a(x_s) & \text{if } U_s > 1-\varepsilon, \end{cases}$$

again, with ties resolved using uniform sampling. Note that, since the U_s and x_s are independent and identically distributed, the b_s are independent and identically distributed, and so is the sequence $f_{b_s}(x_s)$.

Note that by (16) there exists a finite random time

$$S_\varepsilon = \sup\left\{ t < \infty : \mathbb{P}(E_t^\varepsilon | I_t, x_t) < 1-\varepsilon \right\}.$$

By considering the definition of S_ε, it follows that

$$\begin{aligned} \{s > S_\varepsilon\} \cap \{U_s < 1-\varepsilon\} &\subset \{U_s < \mathbb{P}(E_s^\varepsilon | I_s, x_s)\} \\ &\subset \left\{ a_s \in \operatorname*{argmax}_{a \in \mathcal{A}_s^\varepsilon} Q_{s,a}^{\mathrm{Th}} \right\} \\ &\subset \{a_s \in \mathcal{A}_s^\varepsilon\} \\ &\subset \left\{ f_{a_s}(x_s) \geq \min_{b \in \mathcal{A}_s^\varepsilon} f_b(x_s) \right\}. \end{aligned} \tag{17}$$

Also, it is the case that

$$\begin{aligned} \{U_s < 1-\varepsilon\} &= \left\{ b_s \in \operatorname*{argmin}_{b \in \mathcal{A}_s^\varepsilon} f_b(x_s) \right\} \\ &\subset \left\{ f_{b_s}(x_s) = \min_{b \in \mathcal{A}_s^\varepsilon} f_b(x_s) \right\}. \end{aligned} \tag{18}$$

Combining (17) and (18), we have that

$$\{s > S_\varepsilon\} \cap \{U_s < 1-\varepsilon\} \subset \left\{ f_{a_s}(x_s) \geq f_{b_s}(x_s) \right\}. \tag{19}$$

Note also that

$$\{U_s > 1-\varepsilon\} \subset \left\{ f_{b_s}(x_s) = \min_{a' \in \mathcal{A}} f_{a'}(x_s) \leq f_{a_s}(x_s) \right\}. \tag{20}$$

It follows from (19), (20) and the definition of f^* that

$$\{s > S_\varepsilon\} \subset \{f^*(x_s) \geq f_{a_s}(x_s) \geq f_{b_s}(x_s)\}$$

and so

$$\frac{1}{t} \sum_{s=S_\varepsilon}^{t} f^*(x_s) \geq \frac{1}{t} \sum_{s=S_\varepsilon}^{t} f_{a_s}(x_s) \geq \frac{1}{t} \sum_{s=S_\varepsilon}^{t} f_{b_s}(x_s). \tag{21}$$

We will now use inequality (21) to prove the result. The definition of b_s implies that

$$\{U_s < 1-\varepsilon\} \subset \{b_s \in \mathcal{A}_s^\varepsilon\}.$$

By considering the definition of $\mathcal{A}_s^\varepsilon$, it follows that

$$\{U_s < 1-\varepsilon\} \subset \{f_{b_s}(x_s) > f^*(x_s) - 2\varepsilon\}. \tag{22}$$

Since

- S_ε is finite
- the U_s are independent and identically distributed
- the $f_{b_s}(x_s)$ are independent and identically distributed

we can use the strong law of large numbers and (22) to get

$$\begin{aligned}\lim_{t\to\infty}\left[\frac{1}{t}\sum_{s=S_\varepsilon}^{t} f_{b_s}(x_s)\right] &= \mathbb{E}_{\mathcal{U}\times\mathcal{X}} f_{b_s}(x_s)\\ &= \mathbb{P}(U_s < 1-\varepsilon)\mathbb{E}_{\mathcal{X}}[f_{b_s}(x_s)|U_s < 1-\varepsilon] + \mathbb{P}(U_s > 1-\varepsilon)\mathbb{E}_{\mathcal{X}}[f_{b_s}(x_s)|U_s > 1-\varepsilon]\\ &> (1-\varepsilon)\Big(\mathbb{E}_{\mathcal{X}} f^*(\cdot) - 2\varepsilon\Big) + \varepsilon\mathbb{E}_{\mathcal{X}}[f_{b_s}(x_s)|U_s > 1-\varepsilon], \end{aligned} \tag{23}$$

where $\mathbb{E}_{\mathcal{U}\times\mathcal{X}}$ denotes expectation taken with respect to the joint distribution of U_t and x_t and $\mathbb{E}_{\mathcal{X}}$ denotes expectation taken with respect to the distribution of x_t (note that both distributions are the same for all values of t).

By the strong law of large numbers, we get

$$\lim_{t\to\infty}\left[\frac{1}{t}\sum_{s=S_\varepsilon}^{t} f^*(x_s)\right] = \mathbb{E}_{\mathcal{X}} f^*(\cdot). \tag{24}$$

Since (21), (23) and (24) hold, we have that

$$\begin{aligned}\mathbb{E}_{\mathcal{X}} f^*(\cdot) &\geq \lim_{t\to\infty}\left[\frac{1}{t}\sum_{s=S_\varepsilon}^{t} f_{a_s}(x_s)\right]\\ &\geq \lim_{t\to\infty}\left[\frac{1}{t}\sum_{s=S_\varepsilon}^{t} f_{b_s}(x_s)\right]\\ &> (1-\varepsilon)\Big(\mathbb{E}_{\mathcal{X}} f^*(\cdot) - 2\varepsilon\Big) + \varepsilon\mathbb{E}_{\mathcal{X}}[f_{b_s}(x_s)|U_s > 1-\varepsilon].\end{aligned}$$

This holds for arbitrarily small ε, hence

$$\lim_{t\to\infty}\left[\frac{1}{t}\sum_{s=S_\varepsilon}^{t} f_{a_s}(x_s)\right] = \mathbb{E}_{\mathcal{X}} f^*(\cdot). \tag{25}$$

It is the case that

$$\begin{aligned}\lim_{t\to\infty}\frac{1}{t}\sum_{s=1}^{t} f_{a_s}(x_s) &= \lim_{t\to\infty}\frac{1}{t}\sum_{s=1}^{S_\varepsilon-1} f_{a_s}(x_s) + \lim_{t\to\infty}\frac{1}{t}\sum_{s=S_\varepsilon}^{t} f_{a_s}(x_s)\\ &= 0 + \lim_{t\to\infty}\left[\frac{1}{t}\sum_{s=S_\varepsilon}^{t} f_{a_s}(x_s)\right]\end{aligned} \tag{26}$$

as $t\to\infty$ since S_ε is finite and $f_{a_s}(x_s) \leq \sup_{x\in\mathcal{X}} f_{a^*}(x) < \infty$.

Since both (25) and (26) hold, it is true that

$$\lim_{t\to\infty}\left[\frac{1}{t}\sum_{s=1}^{t} f_{a_s}(x_s)\right] = \mathbb{E}_{\mathcal{X}} f^*(\cdot) = \lim_{t\to\infty}\left[\frac{1}{t}\sum_{s=1}^{t} f^*(x_s)\right].$$

Hence

$$\frac{\sum_{s=1}^{t} f_{a_s}(x_s)}{\sum_{s=1}^{t} f^*(x_s)} \xrightarrow{\text{a.s.}} 1 \text{ as } t \to \infty.$$

■

3.2 OBS Algorithm Analysis

We analyse the OBS algorithm in a similar way to the LTS algorithm. In order to prove infinite exploration for the OBS algorithm, we must make an additional assumption on the exploitative value estimates. We assume that exploitative values are less than $\sup \mathcal{S}$ by a constant for all regressor values during periods of time in which their associated actions are not selected. This allows us to make statements similar to inequality (5) in the proof of Lemma 2, however relating to OBS samples rather than LTS samples.

Assumption 5 *Let $a \in \mathcal{A}$ be an arbitrary action, let T be an arbitrary time, and let I_T be an arbitrary history to time T. There exists a $\delta > 0$ depending on a, T, and I_T such that for all $t > T$, all histories $I_t = I_T \cup \{x_T, \ldots, x_{t-1}, r_T, \ldots, r_{t-1}, a_T, \ldots, a_{t-1}\}$ such that $a_s \neq a$ for $s \in \{T, \ldots, t-1\}$, and all $x \in \mathcal{X}$,*

$$\sup \mathcal{S} - \hat{f}_{t,a}(x) > \delta.$$

We now show that the OBS algorithm explores all actions infinitely often. Assumptions 2 and 3 imply that, for any action $a \in \mathcal{A}$,

$$\left[Q_{t,a} - g_a(x_t)\right] \xrightarrow{\mathbb{P}} 0 \text{ as } n_{t,a} \to \infty$$

so that OBS samples associated with actions assumed to be selected infinitely often can be treated in the same way as LTS samples are in the proof of Lemma 2. The only slight difference in the proof comes in the treatment of samples associated with actions assumed to be selected finitely often, although Assumption 5 ensures that the logic is similar.

Lemma 4 *If Assumption 1, 2, 3 and 5 hold, then the OBS algorithm exhibits infinite exploration with probability* 1.

Proof Since $Q_{t,a} = \max(Q^{\text{Th}}_{t,a}, \hat{f}_{t,a}(x_t))$, Assumption 2 and 3 give that $\forall a \in \mathcal{A}^{\text{inf}}$

$$\left[Q_{t,a} - g_a(x_t)\right] \xrightarrow{\mathbb{P}} 0 \text{ as } t \to \infty.$$

Let T and δ be defined as in Lemma 2 (with the $Q^{\text{Th}}_{t,a}$ replaced by $Q_{t,a}$). In the proof of Lemma 2, $g^*(x_t) := \max_{a \in \mathcal{A}} g_a(x_t) + \delta$ is used as a target for samples associated with actions in $a^{\text{fin}} \in \mathcal{A} \backslash 1$ to fall below and the sample associated with action 1 to fall above. The assumptions do not restrict from occurring the event that there exists an action a in $\mathcal{A}^{\text{fin}} \backslash 1$ such that, for all $t > T$, $\hat{f}_{t,a}(x_t) > g^*(x_t)$, thus making it impossible for $Q_{t,a}$ to fall below $g^*(x_t)$. However, Assumption 5 can be used to imply that there exists a $\delta_1 > 0$ such that $\forall a^{\text{fin}} \in \mathcal{A}^{\text{fin}}$ and $\forall t > T$

$$\hat{f}_{t,a^{\text{fin}}}(x_t) < \sup \mathcal{S} - \delta_1. \tag{27}$$

Assumption 1 and inequality (27) then imply that, for all actions $a^{\text{fin}} \in \mathcal{A}^{\text{fin}} \setminus 1$, there exists an $\varepsilon_{a^{\text{fin}}} > 0$ such that

$$\mathbb{P}\Big(Q_{t,a^{\text{fin}}} < \max(g^*(x_t), \sup \mathcal{S} - \delta_1) \Big| I_t, x_t, t > T\Big) > \varepsilon_{a^{\text{fin}}}$$

and also that there exists an $\varepsilon_1 > 0$ such that

$$\mathbb{P}\Big(Q_{t,1} > \max(g^*(x_t), \sup \mathcal{S} - \delta_1) \Big| I_t, x_t, t > T\Big) > \varepsilon_1.$$

The proof then follows in a similar manner to that of Lemma 2, with the $Q^{\text{Th}}_{t,a}$ replaced by $Q_{t,a}$. ■

In the case of the LTS algorithm, it is not necessary for the generated policy to be GLIE for Theorem 1 to hold. Assumptions are only made on total action value estimates, that is, the sum of exploitative and exploratory value, and it is not necessary that the exploratory value converges to zero. Exploitative value estimates are not used explicitly for the LTS algorithm and Lemma 3 is included in this work for completeness. In the case of the OBS algorithm, it is important that Assumption 3 holds so that the policy is GLIE, since exploitative values are used explicitly. The total action value can be equal to the exploitative value estimate so it is important that the exploitative estimate converges to the same value as the LTS samples. Obviously, this would hold if the posterior expectation is used as we suggest, however our framework allows for the use any functions of the regressor satisfying Assumptions 3 and 5 when implementing the OBS algorithm and the convergence result will still hold.

Lemma 5 *If Assumption 1, 2, 3 and 5 hold, then the OBS algorithm policy is GLIE.*

Proof The proof is similar to that of Lemma 3, replacing $\tilde{f}^{\text{Th}}_{t,a}$ with $\tilde{f}_{t,a}$, replacing $Q^{\text{Th}}_{t,a}$ with $Q_{t,a}$ and using the fact that

$$\tilde{f}_{t,a}(x_t) = \max(0, \tilde{f}^{\text{Th}}_{t,a}(x_t)).$$

■

Under Assumptions 1–5, we have that the LTS samples, $Q^{\text{Th}}_{t,a}$, and the exploitative values, $\hat{f}_{t,a}(x_t)$ are consistent estimators of the true expected rewards, $f_a(x_t)$ and that infinite exploration is guaranteed by Lemma 4. Therefore, we have that the OBS samples, $Q_{t,a}$ converge in probability to the true expected rewards, $f_a(x_t)$, as $t \to \infty$. We can therefore prove that the OBS algorithm satisfies convergence criterion (2) using a similar method to that used for the proof of Theorem 1.

Theorem 2 *If Assumptions 1–5 hold, then the OBS algorithm will produce a policy satisfying convergence criterion (2).*

Proof By Assumption 2, 3 and 4 and the infinite exploration guaranteed by Lemma 4, we have that $\forall a \in \mathcal{A}$

$$\big[Q_{t,a} - f_a(x_t)\big] \xrightarrow{\mathbb{P}} 0 \text{ as } t \to \infty$$

since $Q_{t,a} = \max(Q^{\text{Th}}_{t,a}, \hat{f}_{t,a}(x_t))$. The remainder of the proof follows as in the case of Theorem 1 (replacing $Q^{\text{Th}}_{t,a}$ with $Q_{t,a}$). ■

4. Case Studies

In this section, we aim to validate claims made in §1.2 regarding the short term performance of the OBS algorithm by means of simulation. We use the notion of cumulative pseudo-regret (Filippi et al., 2010) to assess the performance of an algorithm. The cumulative pseudo-regret measures the expected difference between the reward the algorithm receives and the reward that would be received if the regression functions were known in advance so that an optimal arm can be chosen on every timestep; it is a standard measure of finite-time performance of a bandit algorithm. Our definition differs slightly from that of Filippi et al. (2010) since we do not restrict attention to generalised linear bandits.

Definition 4 *The cumulative (pseudo) regret, R_T, at time T is given by*

$$R_T = \sum_{t=1}^{T} \Big[f^*(x_t) - f_{a_t}(x_t) \Big].$$

We compare the performance of OBS to that of LTS and various recently proposed action selection methods in simulated Bernoulli bandit and linear regression problem settings in §4.1 and §4.2 respectively. We also consider a real-world version of the problem using data that relates to personalised news article recommendation, the Yahoo! Front Page Today Module User Click Log Data Set (Yahoo! Academic Relations, 2011). Graepel et al. (2010) suggest using LTS to deal with the exploration-exploitation dilemma in a similar sponsored search advertising setting. We compare the OBS performance to that of LTS on the Yahoo! data and obtain results indicating that OBS performs better in the short term.

4.1 Bernoulli Bandit

In the multi-armed Bernoulli bandit problem, there is no regressor present. If the agent chooses action a on any timestep then a reward of 1 is received with probability p_a and 0 with probability $1-p_a$. For each action a, the probability p_a can be estimated by considering the frequency of success observed in past selections of the action. The agent needs to explore in order to learn the probabilities of success for each action, so that the action yielding the highest expected reward can be identified. The agent needs to exploit what has been learned in order to maximise expected reward. The multi-armed Bernoulli bandit problem presents a simple example of the exploration-exploitation dilemma, and has therefore been studied extensively.

4.1.1 PROBLEM CONSIDERED

In this case, we let the prior information, I_0, consist of the following:

- The number of actions, A.
- $(\forall a \in \mathcal{A})(\forall t \in \mathcal{T})\big\{ f_a(x_t) = p_a \big\}$ for $p_a \in (0,1)$ unknown.
- $\forall a, \forall t, z_{t,a} = \begin{cases} -p_a & \text{with probability } 1-p_a, \\ 1-p_a & \text{with probability } p_a. \end{cases}$
- For each action $a \in \mathcal{A}$, the prior distribution of f_a is Beta$(1,1)$ (or equivalently U$(0,1)$).

4.1.2 LTS AND OBS IMPLEMENTATION

Let $\tilde{r}_{\tau,a}$ denote the value of the reward received on the timestep where action a was picked for the τth time. For arbitrary $a \in \mathcal{A}$ define

$$s_{t,a} = \sum_{\tau=1}^{n_{t,a}} \tilde{r}_{\tau,a}.$$

Posterior expectations (using flat priors, as indicated by I_0) can be evaluated easily, so we define exploitative value as

$$\hat{f}_{t,a} := \frac{s_{t,a}+1}{n_{t,a}+2}.$$

The posterior distribution of p_a given I_t has a simple form. We sample

$$Q^{\mathrm{Th}}_{t,a} \sim \mathrm{Beta}(s_{t,a}+1, n_{t,a}-s_{t,a}+1).$$

and set

$$Q_{t,a} = \max(Q^{\mathrm{Th}}_{t,a}, \hat{f}_{t,a}).$$

4.1.3 CONVERGENCE

In this section, we check explicitly that Assumptions 1–5 are satisfied in this Bernoulli bandit setting, therefore proving that the LTS and OBS algorithms generate policies satisfying convergence criterion (2).

Lemma 6 *The LTS total value estimate, $Q^{Th}_{t,a}$, satisfies Assumption 1, for all $a \in \mathcal{A}$.*

Proof Let $a \in \mathcal{A}$, $T > 0$, I_T and $M \in (0,1)$ be arbitrary. For any $t > T$ and $I_t = I_T \cup \{r_T, \ldots, r_{t-1}, a_T, \ldots, a_{t-1}\}$ with $a_s \neq a$ for $s \in \{T, \ldots, t-1\}$, the posterior distribution of f_a given I_t will be the same as the posterior distribution of f_a given I_T (since no further information about f_a is contained in I_t). Let

$$\varepsilon := \frac{1}{2} \min\left\{ \mathbb{P}(Q^{\mathrm{Th}}_{T,a} < M \mid I_T), \mathbb{P}(Q^{\mathrm{Th}}_{T,a} > M \mid I_T) \right\}.$$

We then have that

$$\mathbb{P}(Q^{\mathrm{Th}}_{t,a} > M | I_t) > \varepsilon$$

and

$$\mathbb{P}(Q^{\mathrm{Th}}_{t,a} < M | I_t) > \varepsilon.$$

■

Lemma 7 *The LTS total value estimate, $Q^{Th}_{t,a}$, satisfies Assumptions 2–4, for all $a \in \mathcal{A}$.*

Proof Posterior expectations are given by

$$\begin{aligned} \hat{f}_{t,a} &:= \frac{s_{t,a}+1}{n_{t,a}+2} \\ &= \frac{1+\sum_{\tau=1}^{n_{t,a}} \tilde{r}_{\tau,a}}{n_{t,a}+2}. \end{aligned}$$

Using the strong law of large numbers, we then have

$$\lim_{n_{t,a}\to\infty} \hat{f}_{t,a} = \lim_{n_{t,a}\to\infty} \frac{\sum_{\tau=1}^{n_{t,a}} \tilde{r}_{\tau,a}}{n_{t,a}} = \mathbb{E}(r_t|a_t = a) = p_a = f_a. \tag{28}$$

Therefore, it is the case that

$$\mathbb{E}(Q_{t,a}^{\mathrm{Th}}|\, I_t) = \hat{f}_{t,a} \xrightarrow{\text{a.s.}} f_a \text{ as } n_{t,a} \to \infty. \tag{29}$$

By considering the variance of the LTS samples, we get

$$\begin{aligned}
\mathrm{Var}(Q_{t,a}^{\mathrm{Th}}|\, I_t) &= \frac{(s_{t,a}+1)(n_{t,a}-s_{t,a}+1)}{(n_{t,a}+2)^2(n_{t,a}+3)} \\
&< \frac{(n_{t,a}+2)^2}{(n_{t,a}+2)^2(n_{t,a}+3)} \\
&= \frac{1}{(n_{t,a}+3)} \xrightarrow{\text{a.s.}} 0 \text{ as } n_{t,a} \to \infty.
\end{aligned} \tag{30}$$

From (29) and (30), we then have $\forall a \in \mathcal{A}$

$$Q_{t,a}^{\mathrm{Th}} \xrightarrow{\mathbb{P}} f_a \text{ as } n_{t,a} \to \infty. \tag{31}$$

Note that since $f_a = p_a < 1$ for each $a \in \mathcal{A}$, and $|\mathcal{A}| < \infty$, convergence result (31) shows that Assumptions 2 and 4 hold and convergence results (31) and (28) combined show that Assumption 3 holds. ■

Lemma 8 *The exploitative value estimate, $\hat{f}_{t,a}$, satisfies Assumption 5, for all $a \in \mathcal{A}$.*

Proof Let $a \in \mathcal{A}$, $T > 0$ and I_T be arbitrary. For any $t > T$ and $I_t = I_T \cup \{r_T, \ldots, r_{t-1}, a_T, \ldots, a_{t-1}\}$ with $a_s \neq a$ for $s \in \{T, \ldots, t-1\}$,

$$n_{t,a} = n_{T,a} \quad \text{and} \quad s_{t,a} = s_{T,a}.$$

Therefore

$$\hat{f}_{t,a} = \frac{s_{T,a}+1}{n_{T,a}+2} \leq \frac{n_{T,a}+1}{n_{T,a}+2} < 1 - \frac{1}{n_{T,a}+2} = \sup S - \frac{1}{n_{T,a}+2},$$

so that the assumption is satisfied with $\delta = \frac{1}{n_{T,a}+2}$. ■

Proposition 1 *Within the described Bernoulli bandit setting convergence criterion (2) is satisfied when the LTS or the OBS algorithm is used.*

Proof Assumptions 1–5 hold, so the proof follows directly from Theorems 1 and 2. ■

4.1.4 EXPERIMENTAL RESULTS

We parameterise a Bernoulli problem of the described form with a vector of probabilities, $(p_1, \ldots, p_A)$, corresponding to the expected rewards for the actions in $\mathcal{A}$. We simulate the problem in four environments with parameters $(0.8, 0.9)$, $(0.8, 0.8, 0.8, 0.9)$, $(0.45, 0.55)$ and $(0.45, 0.45, 0.45, 0.55)$. It is well known that the variance of a Bernoulli random variable is maximised when the associated probability of success is 0.5. We choose to consider the four environments mentioned to provide 'low variance' and 'high variance' versions of the problem and to investigate the effect of increasing the number of actions.

For each problem environment, the process is run for 8000 independent trials. A time window of $\mathcal{T} = \{1, \ldots, 5000\}$ is considered on each trial. A trial consists of sampling the potential rewards $r_{t,a} \sim \text{Bernoulli}(p_a)$ for each $t \in \mathcal{T}$ and $a \in \mathcal{A}$ and running all algorithms on the same set of potential rewards, whilst recording the regret incurred. We compare the performance of the LTS and OBS algorithms to that of UCB-Tuned, MOSS, UCB-V, KL-UCB and KL-UCB+ in each of the four simulated environments. The UCB-Tuned and MOSS algorithms are implemented exactly as described by Auer et al. (2002) and Audibert and Bubeck (2010) respectively.[3] The UCB-V algorithm is implemented as described by Audibert et al. (2007), with exploration function and tuning constants set to the 'natural values' suggested.[4] The KL-UCB and KL-UCB+ algorithms are implemented as described by Garivier and Cappé (2011), with constant $c = 0$, as used in their numerical experiments.

The results of the simulations are summarised in Figures 1–4. The left hand plots show cumulative regret averaged over the trials. The right hand plots show boxplots indicating the distribution of final cumulative regret over trials. We consider cumulative regret averaged over trials since this provides an estimate for the expected cumulative regret, $\mathbb{E}(R_T)$, where the expectation is taken with respect to the regressor sequence and the reward and action sequences under the proposed algorithm, a much more meaningful measure than the cumulative regret incurred over any one trial. We plot the average cumulative regret on a logarithmic timescale, so that one can get an indication as to whether an algorithm has a optimal rate of regret.

We first note that, in the cases considered, the MOSS and UCB-V algorithms perform relatively poorly, despite proven regret guarantees. The left hand plots in Figures 1 and 2 indicate that the KL-UCB+ algorithm has the best performance (in terms of expected regret) for the 'low variance' problem environments, whereas Figures 3 and 4 indicate that the UCB-Tuned algorithm has the best performance in the 'high variance' problem environments. Both the OBS and LTS algorithms display highly competitive performance in all cases considered, with the OBS algorithm consistently outperforming the LTS algorithm, as predicted in Section 1.2. It is also indicated that increasing the number of actions from 2 to 4 widens this performance gap between OBS and LTS. There

3. We implement the MOSS algorithm with the time horizon known. We note that the algorithm can be run without knowledge of the horizon using the 'doubling trick' (Cesa-Bianchi and Lugosi, 2006), whereby the horizon used in the algorithm is originally set to 2 and then doubled whenever t exceeds the assumed horizon. In preliminary numerical experiments, the version using knowledge of the time horizon slightly outperformed (with respect to averaged final cumulative regret) the 'doubling trick' version in of all problem environments tested, so we choose to use the former in comparisons.

4. For the UCB-V algorithm, we use exploration function $\mathcal{E}_t = \log t$ and constant $c = 1/6$, in the notation of Audibert et al. (2007). In preliminary numerical experiments, this version outperformed the version used in the numerical experiments section of Audibert and Bubeck (2009) (with $c = 1$ instead) in all four problem environments tested, and so is used for comparisons.

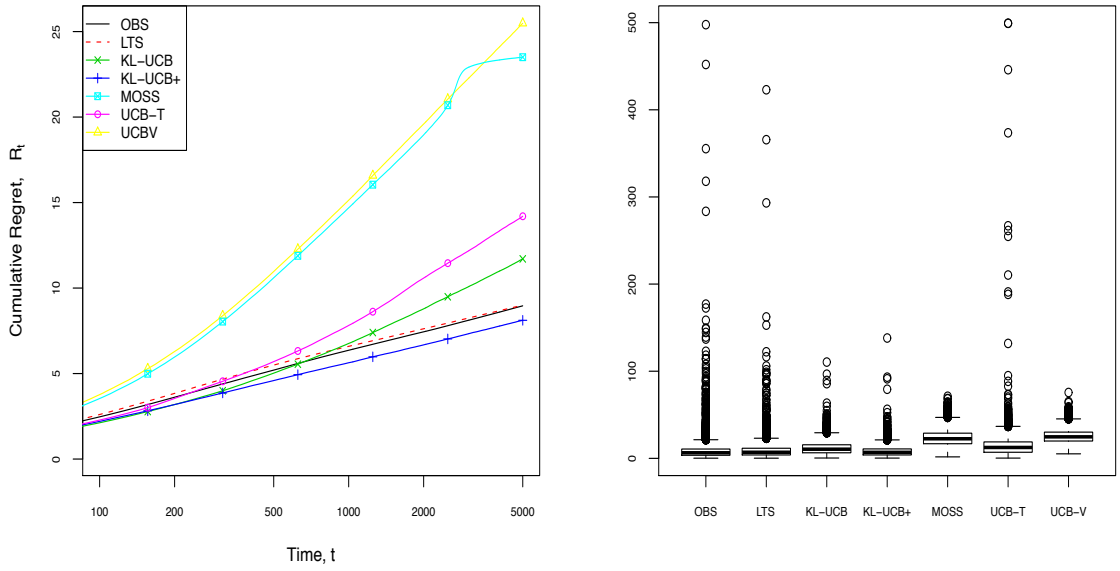


Figure 1: Performance of various algorithms in Bernoulli bandit simulation with parameters $(p_1, p_2) = (0.8, 0.9)$. Left: Cumulative regret averaged over trials. Right: Distribution of cumulative regret at time $t = 5000$. Results based on 8000 independent trials.

is no method tested that outperforms OBS in all four problems and the OBS algorithm displays performance that is never far from the leading algorithm.

The boxplots on the right hand side of the Figures 1–4 indicate that LTS, OBS, UBC-tuned and (to a lesser extent) KL-UCB+ are all 'risky' algorithms, when compared to the others. If one was risk-averse, then the KL-UCB, MOSS and UCBV algorithms are suitable options.[5] It is also worth noting that the regret distribution associated with the OBS algorithm seems to have a fatter upper tail than the LTS algorithm but the LTS algorithm has more variance near the median (which is higher than the OBS median in the four cases considered). A theoretical analysis on the concentration of regret for the OBS and LTS algorithms is desirable so that this can be investigated further, although we leave this to future work.

Finally, in Figure 5, we present plots of the reward ratio (2) through time, for the first 100 trials of the first experimental condition, in order to demonstrate actual results proved in the theoretical part of the paper. The 'almost sure' nature of the convergence of this quantity is observed, in that on some runs there is a period to begin with in which the ratio 'sticks' before asymptoting towards 1, whereas most runs converge quickly towards the asymptote. An identical phenomenon is observed in the other experimental conditions.

5. Note that Audibert and Bubeck (2009) give theoretical results on the concentration of the regret incurred by the UCB-V algorithm, as well as on its expectation.

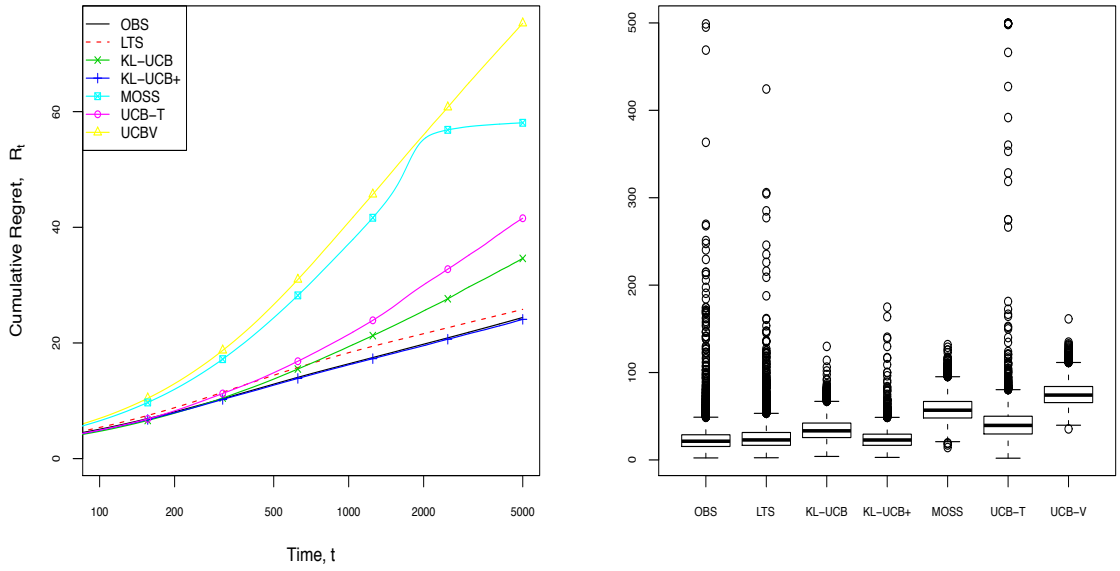


Figure 2: Performance in Bernoulli bandit simulation with parameters $(0.8, 0.8, 0.8, 0.9)$. Note that the curves for the OBS algorithm and the KL-UCB+ algorithm are virtually coincident.

4.2 Linear Regression

In this case, we study a form of the problem in which the expected reward for each action is a linear function of an observable scalar regressor and the reward noise terms are normally distributed. The learning task becomes that of estimating both the intercept and slope coefficients for each of the actions, so that the action yielding the highest expected reward given the regressor can be identified. The exploration-exploitation dilemma is inherent due to uncertainty in regression coefficient estimates caused by the reward noise.

4.2.1 PROBLEM CONSIDERED

In this case, we let the prior information, I_0, consist of the following:

- The number of actions, $A = 4$.
- $(\forall t \in \mathcal{T})\{x_t \sim U(-0.5, 0.5)\}$.
- $(\forall a \in \mathcal{A})(\forall t \in \mathcal{T})\{f_a(x_t) = \beta_{1,a} + \beta_{2,a} x_t\}$ for $\beta_{1,a}, \beta_{2,a} \in \mathbb{R}$ unknown.
- $(\forall a \in \mathcal{A})(\forall t \in \mathcal{T})\{z_{t,a} \sim \mathrm{N}(0, \sigma_a^2)\}$ for $\sigma_a \in \mathbb{R}$ unknown.
- $(\forall a \in \mathcal{A})\{$The (improper) prior distributions for $\beta_{1,a}$ and $\beta_{2,a}$ are flat over $\mathbb{R}\}$.
- $(\forall a \in \mathcal{A})\{$The (improper) prior distribution of σ_a^2 is flat over $\mathbb{R}^+\}$.

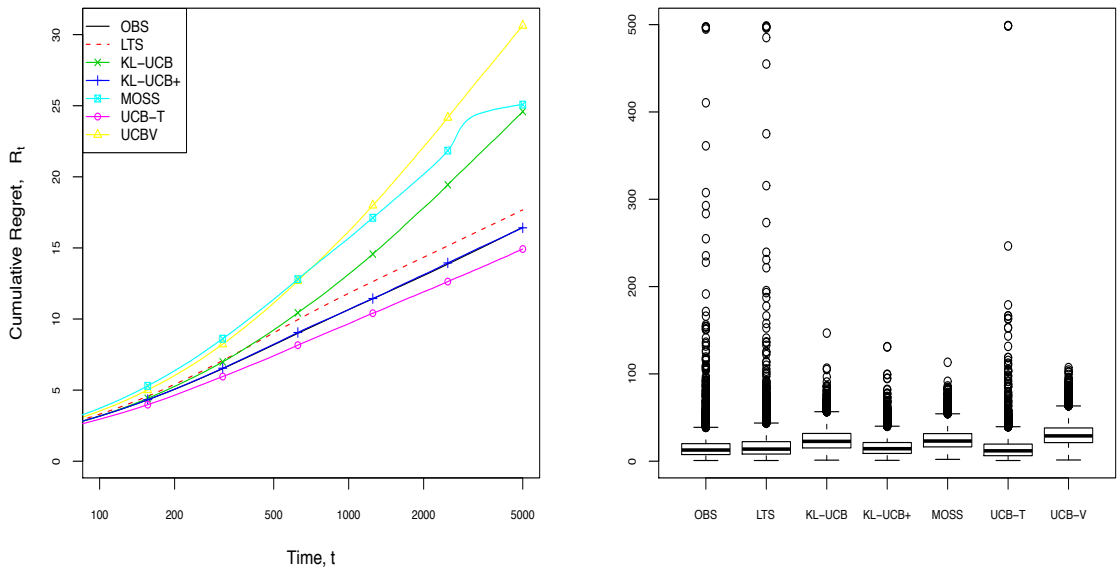


Figure 3: Performance in Bernoulli bandit simulation with parameters $(0.45, 0.55)$. Note that the curves for the OBS algorithm and the KL-UCB+ algorithm are virtually coincident.

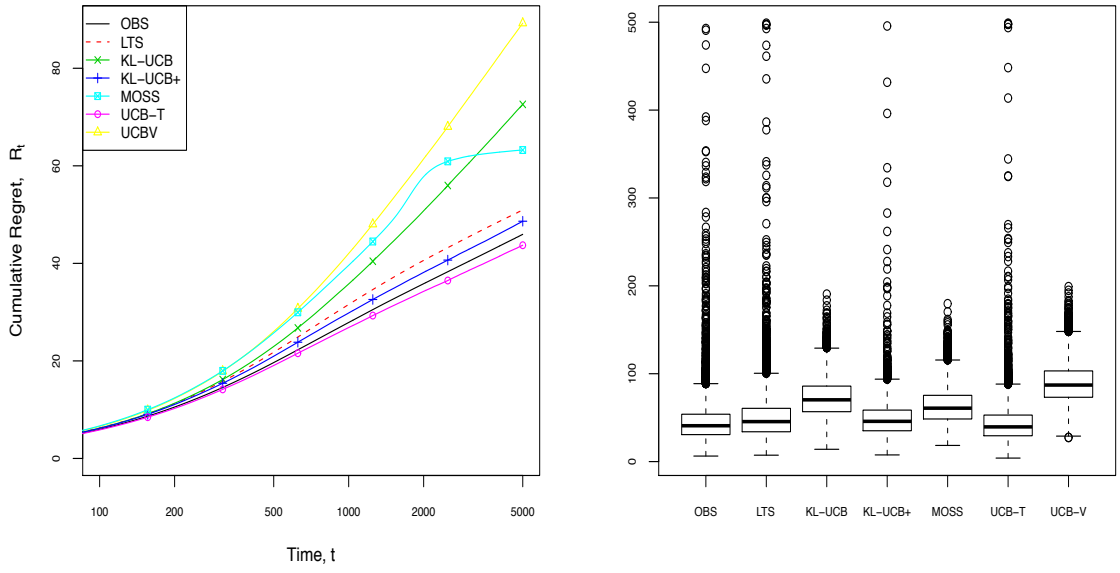


Figure 4: Performance in Bernoulli bandit simulation with parameters $(0.45, 0.45, 0.45, 0.55)$.

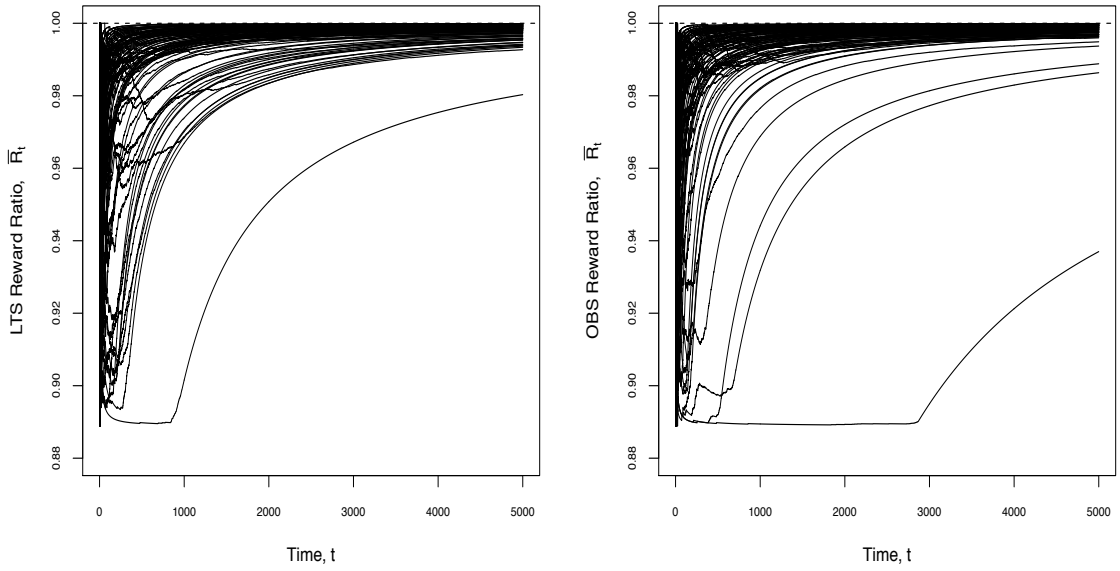


Figure 5: Convergence of the ratio (2) in the first 100 Bernoulli bandit simulations with parameters $(p_1, p_2) = (0.8, 0.9)$.

4.2.2 LTS AND OBS IMPLEMENTATION

Denote estimators at time t of the parameters $\mathbf{b}_a$ and σ_a for $a = 1, \ldots, A$ as $\hat{\mathbf{b}}_{t,a}$ and $\hat{\sigma}_{t,a}$ respectively, where $\mathbf{b}_a = (\beta_{1,a}, \beta_{2,a})^T$. For all $a \in \mathcal{A}$, denote $\mathcal{T}_{t,a} = \{\tau \in \{1, \ldots, t-1\} : a_\tau = a\}$ and the $n_{t,a}$-vectors of regressors and rewards observed at time steps in $\mathcal{T}_{t,a}$ as $\mathbf{x}_{t,a}$ and $\mathbf{r}_{t,a}$ respectively. Denote the $n_{t,a} \times 2$ matrix formed by the concatenation of $\mathbf{1}_{n_{t,a}}$ and $\mathbf{x}_{t,a}$ as $\mathbf{X}_{t,a}$, where $\mathbf{1}_{n_{t,a}}$ is the $n_{t,a}$-vector with every component equal to 1. Let $\hat{\mathbf{b}}_{t,a}$ be given by the least squares equation

$$\hat{\mathbf{b}}_{t,a} := (\mathbf{X}_{t,a}^T \mathbf{X}_{t,a})^{-1} \mathbf{X}_{t,a}^T \mathbf{r}_{t,a}.$$

Let us also denote $\mathbf{x}_t = (1, x_t)^T$. Posterior expectations (using flat priors, as indicated by I_0) can be evaluated easily, so we define exploitative value as

$$\hat{f}_{t,a}(x_t) := \mathbf{x}_t^T \hat{\mathbf{b}}_{t,a}.$$

Let $\hat{\sigma}_{t,a}$ be given by

$$\hat{\sigma}_{t,a} := \sqrt{\frac{1}{n_{t,a} - 2} (\mathbf{r}_{t,a} - \mathbf{X}_{t,a} \hat{\mathbf{b}}_{t,a})^T (\mathbf{r}_{t,a} - \mathbf{X}_{t,a} \hat{\mathbf{b}}_{t,a})}$$

and let $U_{t,a} \sim t_{n_{t,a}-2}$. We define the LTS exploratory value as

$$\tilde{f}_{t,a}^{\mathrm{Th}}(x_t) := \left[\hat{\sigma}_{t,a} \sqrt{\mathbf{x}_t^T (\mathbf{X}_{t,a}^T \mathbf{X}_{t,a})^{-1} \mathbf{x}_t} \right] U_{t,a}. \tag{32}$$

The LTS total value is given by

$$Q^{\mathrm{Th}}_{t,a}(x_t) = \hat{f}_{t,a}(x_t) + \tilde{f}^{\mathrm{Th}}_{t,a}(x_t).$$

The OBS total value is given by

$$Q_{t,a}(x_t) = \max(Q^{\mathrm{Th}}_{t,a}, \hat{f}_{t,a}(x_t)).$$

Note that if $n_{t,a} \in \{0,1,2\}$ then the posterior distribution of $f_a(x_t)$ is improper. In these situations, we sample values from $N(0,10^3)$ to obtain $Q^{\mathrm{Th}}_{t,a}$.

4.2.3 CONVERGENCE

In this section, we check explicitly that Assumptions 1–5 are satisfied in this linear regression setting, therefore proving that the LTS and OBS algorithms generate policies satisfying convergence criterion (2).

Lemma 9 *The LTS total value estimate, $Q^{Th}_{t,a}$, satisfies Assumption 1, for all $a \in \mathcal{A}$.*

Proof Let $a \in \mathcal{A}$, $T > 0$, I_T and $M \in \mathbb{R}$ be arbitrary. For any $t > T$ and $I_t = I_T \cup \{r_T, \ldots, r_{t-1}, a_T, \ldots, a_{t-1}\}$ with $a_s \neq a$ for $s \in \{T, \ldots, t-1\}$, the posterior distribution of $\mathbf{b}_a$ and σ^2_a given I_t will be the same as that given I_T (since no further information about f_a is contained in I_t). In particular for each regressor x, $\hat{f}_{t,a}(x) = \hat{f}_{T,a}(x)$, and $\tilde{f}^{\mathrm{Th}}_{t,a}(x)$ has the same distribution given I_t as it did given I_T. Define

$$\varepsilon := \frac{1}{2} \min_{x \in [-0.5, 0.5]} \min\left\{ \mathbb{P}(\tilde{f}^{\mathrm{Th}}_{T,a}(x) < M - \hat{f}_{T,a}(x) \,|\, I_T), \mathbb{P}(\tilde{f}^{\mathrm{Th}}_{T,a}(x) < M - \hat{f}_{T,a}(x) \,|\, I_T) \right\}.$$

Since $Q^{\mathrm{Th}}_{t,a} = \hat{f}_{t,a}(x_t) + \tilde{f}^{\mathrm{Th}}_{t,a}(x_t)$, we then have that

$$\mathbb{P}(Q^{\mathrm{Th}}_{t,a} > M | I_t) > \varepsilon$$

and

$$\mathbb{P}(Q^{\mathrm{Th}}_{t,a} < M | I_t) > \varepsilon.$$

■

Lemma 10 (taken from Eicker, 1963) is used to prove the consistency of the least squares estimators of the regression coefficients.

Lemma 10 *The least squares estimators $\hat{\mathbf{b}}_{t,a}$, $t = 2, 3, \ldots$ converge in probability to $\mathbf{b}_a$ as $n_{t,a} \to \infty$ if and only if $\lambda_{min}(\mathbf{X}^T_{t,a}\mathbf{X}_{t,a}) \to \infty$ as $n_{t,a} \to \infty$, where $\lambda_{min}(\mathbf{X}^T_{t,a}\mathbf{X}_{t,a})$ is the smallest eigenvalue of $\mathbf{X}^T_{t,a}\mathbf{X}_{t,a}$.*

Lemma 11 *The exploitative value estimate $\hat{f}_{t,a}(x_t) \xrightarrow{\mathbb{P}} f_a(x_t)$ as $n_{t,a} \to \infty$.*

Proof Let $\tilde{x}_{i,a}$ denote the value of the regressor presented on the timestep where action a was picked for the ith time.

$$\mathbf{X}^T_{t,a}\mathbf{X}_{t,a} = \begin{pmatrix} n_{t,a} & \sum_{i=1}^{n_{t,a}} \tilde{x}_{i,a} \\ \sum_{i=1}^{n_{t,a}} \tilde{x}_{i,a} & \sum_{i=1}^{n_{t,a}} \tilde{x}^2_{i,a} \end{pmatrix}.$$

The smallest eigenvalue is given by

$$\lambda_{\min} = \frac{n_{t,a}}{2}\left[\frac{\sum_{i=1}^{n_{t,a}} \tilde{x}_{i,a}^2}{n_{t,a}} + 1 - \sqrt{\left(\frac{\sum_{i=1}^{n_{t,a}} \tilde{x}_{i,a}^2}{n_{t,a}} + 1\right)^2 - 4\left(\frac{\sum_{i=1}^{n_{t,a}} \tilde{x}_{i,a}^2}{n_{t,a}} - \frac{\left(\sum_{i=1}^{n_{t,a}} \tilde{x}_{i,a}\right)^2}{n_{t,a}^2}\right)}\right].$$

Therefore, since $\mathrm{Var}X > 0$, we have that

$$\lim_{n_{t,a}\to\infty} \lambda_{\min} = \lim_{n_{t,a}\to\infty} \frac{n_{t,a}}{2}\left[\mathbb{E}X^2 + 1 - \sqrt{\left(\mathbb{E}X^2+1\right)^2 - 4\mathrm{Var}X}\right] = \infty.$$

Using Lemma 10, we then have

$$\left[\hat{\mathbf{b}}_{t,a} - \mathbf{b}_a\right] \xrightarrow{\mathbb{P}} \mathbf{0} \text{ as } n_{t,a} \to \infty.$$

Multiplying on the left by $\mathbf{x}^T$ then gives

$$\left[\hat{f}_{t,a}(x_t) - f_a(x_t)\right] \xrightarrow{\mathbb{P}} 0 \text{ as } n_{t,a} \to \infty.$$

■

Lemma 12 *The LTS total value estimate, $Q_{t,a}^{Th}$, satisfies Assumptions 2–4, for all $a \in \mathcal{A}$.*

Proof To prove this lemma, we need to show that $\tilde{f}_{t,a}(x_t) \xrightarrow{\mathbb{P}} 0$ as $n_{t,a} \to \infty$ for all actions $a \in \mathcal{A}$. In order to do this, we consider each component in the product that forms $\tilde{f}_{t,a}(x_t)$ (see (32)). Firstly, we consider $U_{t,a}$. It is a well known (as is described in Zwillinger, 2000) that

$$U_{t,a} \xrightarrow{\mathbb{D}} N(0,1) \text{ as } n_{t,a} \to \infty. \tag{33}$$

Next, we consider $\hat{\sigma}_{t,a}$. Using the facts that $\hat{\mathbf{b}}_{t,a} \xrightarrow{\mathbb{P}} \mathbf{b}_a$ as $n_{t,a} \to \infty$, $z_{t,a} = r_t - f_a(x_t)$ and $\mathbb{E}[z_{t,a}] = 0$ we have that

$$\begin{aligned}
\hat{\sigma}_{t,a} &:= \sqrt{\frac{1}{n_{t,a}-2}(\mathbf{r}_{t,a} - \mathbf{X}_{t,a}\hat{\mathbf{b}}_{t,a})^T(\mathbf{r}_{t,a} - \mathbf{X}_{t,a}\hat{\mathbf{b}}_{t,a})} \\
&\xrightarrow{\mathbb{P}} \sqrt{\frac{1}{n_{t,a}-2}(\mathbf{r}_{t,a} - \mathbf{X}_{t,a}\mathbf{b}_a)^T(\mathbf{r}_{t,a} - \mathbf{X}_{t,a}\mathbf{b}_a)} \text{ as } n_{t,a} \to \infty \\
&\xrightarrow{\text{a.s.}} \sqrt{\mathbb{E}[z_{t,a}^2]} \text{ as } n_{t,a} \to \infty \\
&= \sqrt{\mathrm{Var}[z_{t,a}] + [\mathbb{E}[z_{t,a}]]^2} \\
&= \sqrt{\mathrm{Var}[z_{t,a}]} = \sigma_a.
\end{aligned} \tag{34}$$

Finally, let us consider $\mathbf{x}_t^T(\mathbf{X}_{t,a}^T\mathbf{X}_{t,a})^{-1}\mathbf{x}_t$. We start by looking at the determinant of $X_{t,a}^T X_{t,a}$. We have that

$$\begin{aligned}\det X_{t,a}^T X_{t,a} &= n_{t,a}^2\left(\frac{1}{n_{t,a}}\sum_{i=1}^{n_{t,a}}\tilde{x}_{i,a}^2 - \Big(\frac{1}{n_{t,a}}\sum_{i=1}^{n_{t,a}}\tilde{x}_{i,a}\Big)^2\right)\\ &\overset{\text{a.s.}}{\to} n_{t,a}^2\Big(\mathbb{E}[x_t^2]-\mathbb{E}[x_t]^2\Big) \text{ as } n_{t,a}\to\infty\\ &= n_{t,a}^2\mathrm{Var}[x_t]. \end{aligned} \tag{35}$$

Using the standard formula for inverting a 2×2 matrix, we get

$$\begin{aligned}\mathbf{x}_t^T(\mathbf{X}_{t,a}^T\mathbf{X}_{t,a})^{-1}\mathbf{x}_t =& \frac{1}{\det X_{t,a}^T X_{t,a}}\left(\sum_{i=1}^{n_{t,a}}\tilde{x}_{i,a}^2 - 2x_t\sum_{i=1}^{n_{t,a}}\tilde{x}_{i,a} + x_t^2 n_{t,a}\right)\\ =& \frac{1}{\det X_{t,a}^T X_{t,a}}\left(n_{t,a}\Big[\frac{1}{n_{t,a}}\sum_{i=1}^{n_{t,a}}\tilde{x}_{i,a}^2 - \Big(\frac{1}{n_{t,a}}\sum_{i=1}^{n_{t,a}}\tilde{x}_{i,a}\Big)^2\Big]\right.\\ &\left.+\frac{1}{n_{t,a}}\Big(\sum_{i=1}^{n_{t,a}}\tilde{x}_{i,a}\Big)^2 - 2x_t\sum_{i=1}^{n_{t,a}}\tilde{x}_{i,a} + x_t^2 n_{t,a}\right)\\ =& \frac{1}{n_{t,a}} + \frac{1}{\det X_{t,a}^T X_{t,a}}\left(\frac{1}{n_{t,a}}\Big(\sum_{i=1}^{n_{t,a}}\tilde{x}_{i,a}\Big)^2 - 2x_t\sum_{i=1}^{n_{t,a}}\tilde{x}_{i,a} + x_t^2 n_{t,a}\right)\\ =& \frac{1}{n_{t,a}} + \frac{1}{\det X_{t,a}^T X_{t,a}}\left(n_{t,a}\Big[\Big(\frac{1}{n_{t,a}}\sum_{i=1}^{n_{t,a}}\tilde{x}_{i,a}\Big)^2 - 2\frac{1}{n_{t,a}}x_t\sum_{i=1}^{n_{t,a}}\tilde{x}_{i,a} + x_t^2\Big]\right)\\ =& \frac{1}{n_{t,a}} + \frac{1}{\det X_{t,a}^T X_{t,a}}\left(n_{t,a}\Big[\frac{1}{n_{t,a}}\sum_{i=1}^{n_{t,a}}\tilde{x}_{i,a} - x_t\Big]^2\right). \end{aligned} \tag{36}$$

Using (35), (36) and the facts that $\mathrm{Var}[x_t]>0$ and both x_t and $\mathbb{E}[x_t]$ are bounded, we have that

$$\mathbf{x}_t^T(\mathbf{X}_{t,a}^T\mathbf{X}_{t,a})^{-1}\mathbf{x}_t \overset{\text{a.s.}}{\to} \frac{1}{n_{t,a}} + \frac{\Big[\mathbb{E}[x_t]-x_t\Big]^2}{n_{t,a}\mathrm{Var}[x_t]} \overset{\text{a.s.}}{\to} 0 \text{ as } n_{t,a}\to\infty. \tag{37}$$

Equations (33), (34) and (37) imply that $\tilde{f}_{t,a}^{\mathrm{Th}}(x_t)\overset{\mathbb{P}}{\to}0$ as $n_{t,a}\to\infty$. Therefore, since $Q_{t,a}^{\mathrm{Th}} = \hat{f}_{t,a}(x_t) + \tilde{f}_{t,a}^{\mathrm{Th}}(x_t)$, Lemma 11 gives us that

$$Q_{t,a}^{\mathrm{Th}} - f_a(x_t)\to 0 \text{ as } n_{t,a}\to\infty,$$

satisfying Assumptions 2 and 4. This same holds for $\hat{f}_{t,a}(x_t)$, hence Assumption 3 is satisfied too. ■

Lemma 13 *The exploitative value estimate, $\hat{f}_{t,a}(x_t)$, satisfies Assumption 5, for all $a\in\mathcal{A}$.*

Proof Let $a\in\mathcal{A}$, and $T>0$ be arbitrary. For any $t>T$ and $I_t = I_T\cup\{r_T,\ldots,r_{t-1}, a_T,\ldots,a_{t-1}\}$ with $a_s\neq a$ for $s\in\{T,\ldots,t-1\}$, the regression coefficients $\hat{\mathbf{b}}_{t,a}$ are equal to $\hat{\mathbf{b}}_{T,a}$. Hence

$$\max_{x\in[-0.5,0.5]}\hat{f}_{t,a}(x) = \max_{x\in[-0.5,0.5]}\hat{f}_{T,a}(x).$$

The Assumption then follows by noting that $\mathcal{S} = \mathbb{R}$. ■

Proposition 2 *Within the described linear regression setting convergence criterion (2) is satisfied when the LTS or the OBS algorithm is used.*

Proof Assumptions 1–5 hold, so the proof follows directly from Theorems 1 and 2. ■

4.2.4 EXPERIMENTAL RESULTS

The process is run for 10000 independent trials. A time window of $\mathcal{T} = \{1, \ldots, 5000\}$ is considered on each trial. The regression coefficients for the actions are set to $(\beta_{1,a}, \beta_{2,a}) = (0,1), (0,-1), (-0.1,0), (0.1,0)$ for $a = 1,2,3,4$ respectively. The resulting expected reward functions are plotted in Figure 6. For each trial:

- $\forall t \in \mathcal{T}$ sample $x_t \sim \mathrm{U}(-0.5, 0.5)$
- $\forall a \in \mathcal{A}$ and $\forall t \in \mathcal{T}$ sample $z_{t,a} \sim \mathrm{N}(0, \sigma_a^2)$ with $\sigma_a = 0.5$
- $\forall a \in \mathcal{A}$ and $\forall t \in \mathcal{T}$ evaluate potential reward $r_{t,a} = \beta_{1,a} + \beta_{2,a} x_t + z_{t,a}$
- record the regret incurred using various action selection methods.

We compare the performance of LTS and OBS to an interval estimation method (or LinUCB, in the terminology of Li et al., 2010) similar to that described in Pavlidis et al. (2008). However we use the posterior distribution of the mean to evaluate the upper confidence bound rather than using the predictive distribution. Specifically, the action selection rule used is given by

$$a_t = \underset{a \in \mathcal{A}}{\operatorname{argmax}} \left[\hat{f}_{t,a}(x_t) + \hat{\sigma}_{t,a} \sqrt{\mathbf{x}_t^T (\mathbf{X}_{t,a}^T \mathbf{X}_{t,a})^{-1} \mathbf{x}_t} \right] t_{1-\frac{\lambda}{100}, n_{t,a}-2}$$

where $t_{\gamma,n}$ denotes the quantile function of Student's t distribution with n degrees of freedom evaluated at γ. This ensures that the value estimates are consistent, that is, the value estimates converge to the true expected reward as associated actions are selected infinitely often. We implement the IE method with parameter values $\lambda = 0.01$, $\lambda = 5$ and $\lambda = 25$.

The results of the simulation can be seen in Figures 7 and 8. Figure 7 (left) shows cumulative regret averaged over the trials. The OBS algorithm displays the best performance (with respect to cumulative regret averaged over trials) in the problem considered, and this performance is significantly better than that of the LTS algorithm. It is also clear that the IE method performance is highly sensitive to parameter choice. The best parameter choice in this case is $\lambda = 5$, however, it is not clear how this parameter should be chosen based on the prior information provided. In general, if λ is 'too high', then too much emphasis is put on short term performance and if λ is 'too low' then too much emphasis is put on long term performance. This is indicated by the curves for the $\lambda = 25$ and $\lambda = 0.01$ methods respectively. Figure 7 (right) shows boxplots indicating the distribution of final cumulative regret over trials. It is indicated that the IE methods become riskier as the significance parameter used is increased and that the significance parameter provides a way of trading off median efficiency and risk. The only method to compete with OBS on cumulative regret averaged

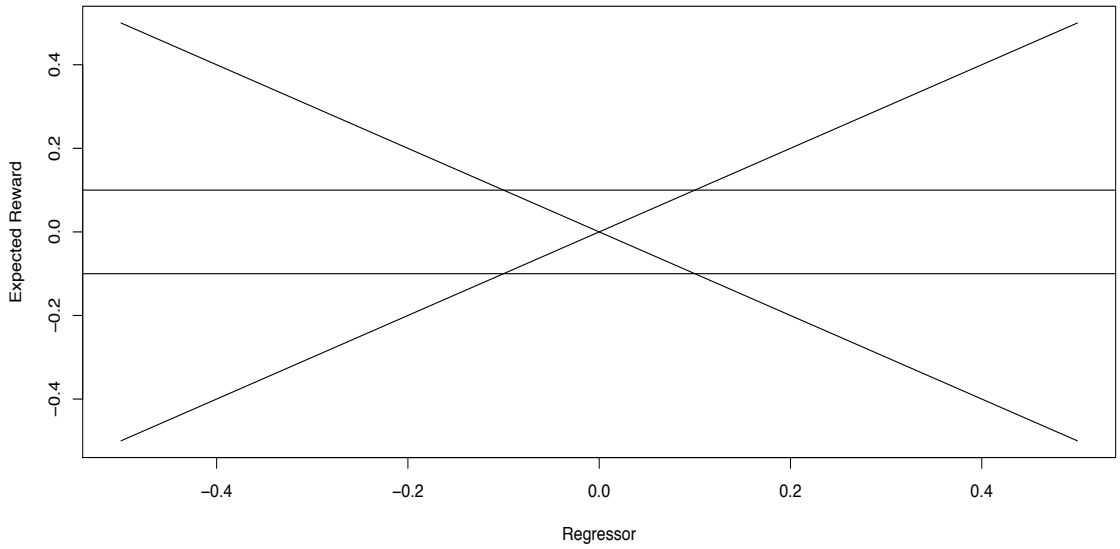


Figure 6: The expected reward functions for the 4 actions in linear regression simulation.

over trials is the $\lambda = 5$ IE method, however the OBS final regret distribution is more concentrated than the $\lambda = 5$ IE method. In Figure 8, we present plots of the reward ratio (2) through time, for the first 100 experiments, in order to demonstrate actual results proved in the theoretical part of the paper. Although convergence of the ratio has not occurred after the 5000 iterations, it is clear that the ratio is improving over time.

4.3 Web-Based Personalised News Article Recommendation

We now consider the problem of selecting news articles to recommend to internet users based on information about the users. In our framework, the recommendation choice corresponds to an action selection and the user information corresponds to a regressor. The objective is to recommend an article that has the highest probability of being clicked.

We test the performance of the LTS and OBS algorithms on a real-world data set, the Yahoo! Front Page Today Module User Click Log Data Set (Yahoo! Academic Relations, 2011). A similar study is performed by Chapelle and Li (2011). However we consider multiple trials over a short time horizon, as opposed to Chapelle and Li's single trial over the full data set, to investigate the short term performance of the algorithms, and in particular to address the claim made in Section 1.2 regarding a potential short term benefit of using OBS over using LTS. It is necessary to average results over multiple trials given the randomised nature of the OBS and LTS algorithms. We also test the LinUCB algorithm of Li et al. (2010) with various parameter settings to provide a benchmark for comparison.

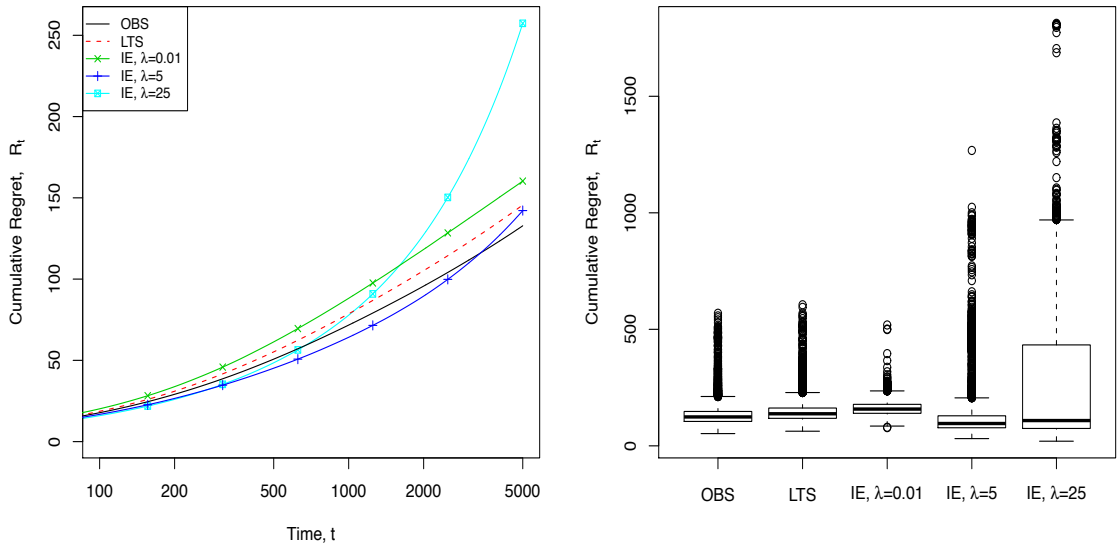


Figure 7: Performance of various algorithms in linear regression simulations. Left: Cumulative regret averaged over trials. Right: Distribution of cumulative regret at $t = 5000$. Results based on 10000 independent trials.

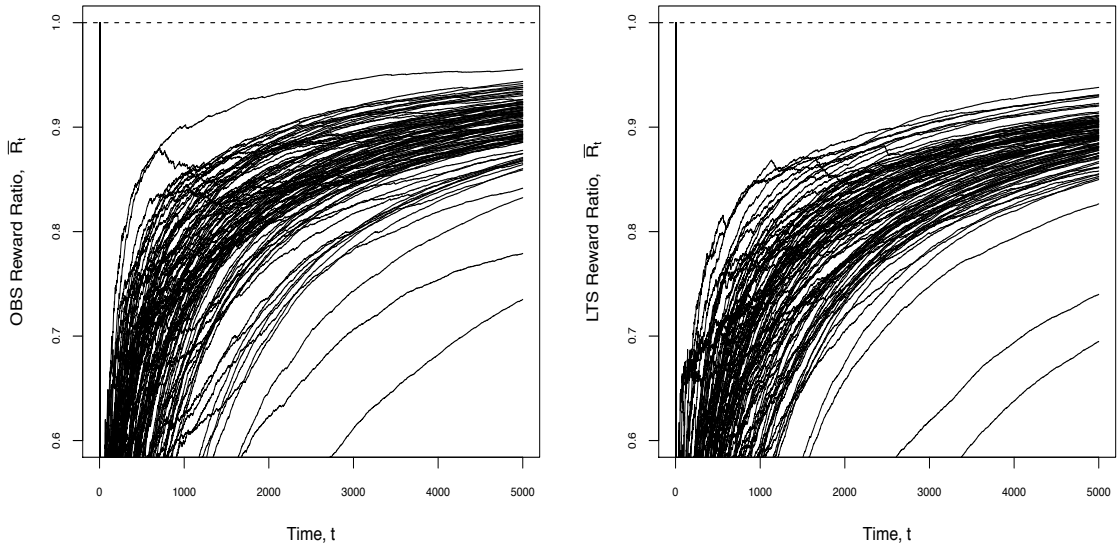


Figure 8: Convergence of the ratio (2) in the first 100 linear regression simulations.

4.3.1 USE OF DATA SET

The data set describes approximately 36M instances of news articles being recommended to internet users on the Yahoo! Front Page Today Module at random times in May 2009. The form and collection of the data set are both described in detail by Li et al. (2010). For each recommendation, the data contains information concerning which article was recommended, whether the recommendation was clicked and a feature vector describing the user. The recommended articles are chosen uniformly at random from a dynamic pool of about 20 choices, with articles being added and removed at various points of the process. The user features, $\mathbf{x}_t$, are given as vectors of length 6 with one component fixed to 1, and are constructed as described by Li et al. (2010). The reward is defined to be 1 if the recommendation is clicked and 0 otherwise.

The use of past data presents a problem in evaluating a decision-making algorithm. Specifically, within the data a random article is recommended on each instance, which might well be different to the article that the decision-making algorithm selects during testing. This problem can be avoided by implementing the unbiased offline evaluator procedure of Li et al. (2011). Under this procedure, if the action selected by the algorithm does not match the action selected in the data point, the current data point, and subsequent data points, are ignored until a data point which matches user data and action selection occurs. The observed reward from this data point is then awarded to the algorithm, and the user data from the next recommendation instance in the data is used in the next evaluation step.

4.3.2 ALGORITHM IMPLEMENTATION

The LTS and OBS algorithms are implemented using the logistic regression model of Chapelle and Li (2011). It is assumed that there is an unknown weight vector, $\mathbf{w}_a$, for each article $a \in \mathcal{A}$ such that

$$\mathbb{P}(r_t = 1 | a_t = a, \mathbf{x}_t = \mathbf{x}) = (1 + \exp(-\mathbf{w}_a^T \mathbf{x}))^{-1}.$$

Approximate posterior distributions for each $\mathbf{w}_a$ are estimated to be Gaussian with mean and variance updates as described in Algorithm 3 of Chapelle and Li (2011). For our numerical experiment, we set the unspecified regularisation parameter of Chapelle and Li (2011) to 100. The LTS algorithm can easily be implemented by sampling weight vectors from the posteriors and selecting the article with the weight vector forming the highest scalar product with the current user feature vector. The OBS algorithm can easily be implemented by also considering posterior means of these scalar products. We also test the LinUCB algorithm, as implemented by Chapelle and Li (2011), with parameter α set to each of 0.5, 1 and 2.

4.3.3 NUMERICAL EXPERIMENTS

As previously mentioned, our focus is short term performance averaged over numerous trials. We focus on the case of only 4 articles, and therefore remove all instances outwith these 4 articles from the data set. On each of $2,500$ trials, we run each of the 5 algorithms until $5,000$ interactions are accepted using data from the start of the supplied data set (Yahoo! Academic Relations, 2011); we use only data from the start of the data set to avoid confounding the algorithm evaluations with the non-stationarity of the data.

The concept of regret is difficult to use as a performance measure in this setting, since there is no true model given for comparison. We instead consider the percentage of past timesteps resulting in clicks, otherwise known as the click-through rate (CTR), and percentage benefit of OBS over LTS

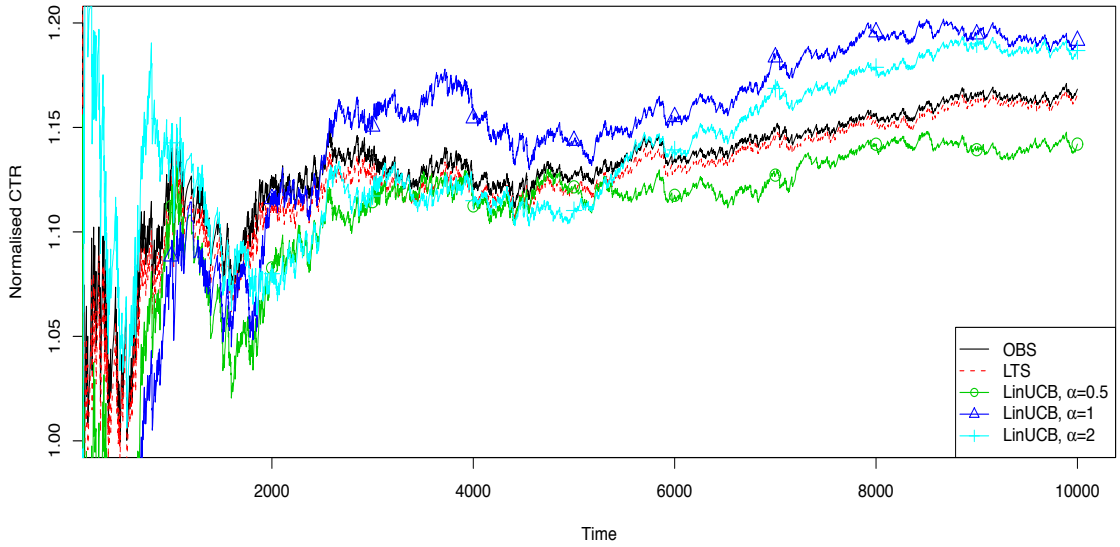


Figure 9: Normalised Click Through Rate through time for various algorithms. Results averaged over 2,500 independent trials.

with respect to CTR. Again, to avoid issues of non-stationarity, we normalise all CTRs by dividing by the CTR achieved (on these four articles) in the original data set.

The results of the experiment can be found in Figures 9 and 10. Figure 9 shows the normalised CTR for all 5 algorithms, averaged over all 2500 runs. It is clear that the performance of the LinUCB algorithm is sensitive to parameter choice; the version with parameter set to 1 performs much better than the version set to 0.5, and it is not clear in advance of implementing the algorithm which parameter will be optimal. As a caveat on these results, it is worth noting that the portion of the data set used for each trial is the same, and also that the LinUCB algorithms are deterministic given past information (except in the case of a tie in action values), so it is hard to extrapolate general results relating to the performance of LinUCB algorithms. Furthermore Chapelle and Li (2011) explain that the performance of the LinUCB algorithm degrades significantly with increasing feedback delay, while the LTS and OBS algorithms are more robust to the delay, so the strong performance of the highest-performing LinUCB algorithm in this experiment should not be taken as conclusive evidence of high real-world performance. Unfortunately it is not possible to produce plots comparable to Figures 5 and 8 in this case since the true optimal actions are not known. Figure 10 shows the difference in performance of OBS and LTS, expressed as a percentage of LTS performance, averaged over all 2500 runs. It is clear that the OBS algorithm outperforms the LTS algorithm across the time period considered, validating the intuition in Section 1.2. The short term improvement is small, but in many web-based application, a small difference in performance can be significant (Graepel et al., 2010).

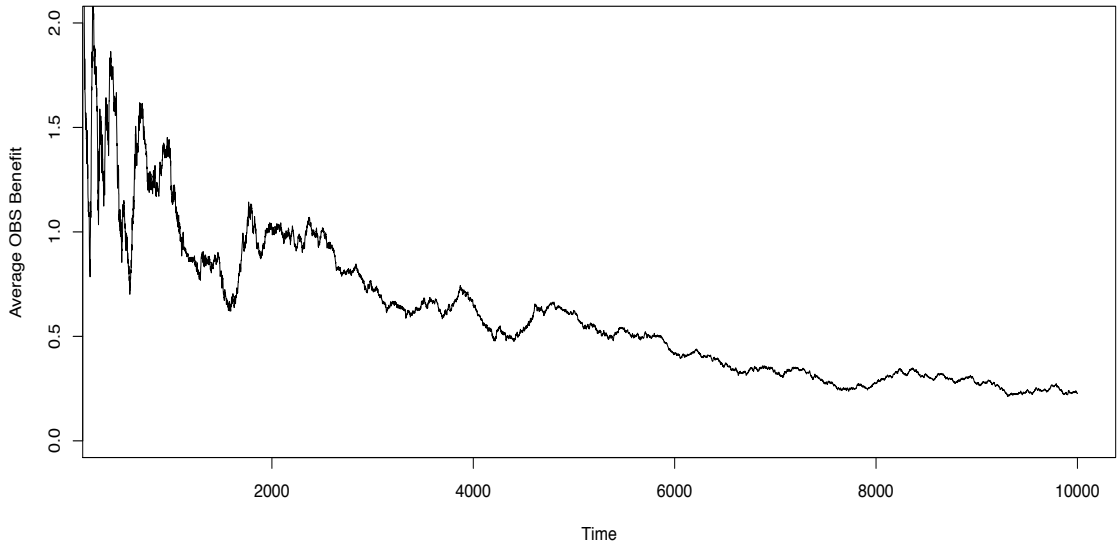


Figure 10: OBS CTR as a percentage improvement of LTS CTR through time. Results averaged over $2,500$ independent trials.

5. Discussion

The assumptions made for the theoretical results in Section 3 are mild in the sense that one would expect them to hold if the true posterior distributions and expectations are used. It is worth noting that convergence criterion (2) is satisfied even when approximations to the posterior distributions and expectations for the $f_a(x_t)$ are used with the LTS and OBS algorithms, so long as the relevant assumptions are satisfied. Hence, convergence is guaranteed for a large class of algorithms.

We have seen that both the LTS and the OBS algorithms are easy to implement in the cases considered. They are also computationally cheap and robust to the use of posterior approximations, when compared to belief-lookahead methods, such as Gittins indices. The simulation results for the OBS algorithm are very encouraging. In every case, the OBS algorithm outperformed the LTS algorithm and performed well compared to recent benchmarks.

Acknowledgments

We thank the reviewers and editor for their comments, which have contributed significantly to the quality of this article. This research was undertaken as part of the ALADDIN (Autonomous Learning Agents for Decentralised Data and Information Networks) project and is jointly funded by a BAE Systems and EPRSC (Engineering and Physical Sciences Research Council) strategic partnership (EP/C548051/1). Many of the ideas in this paper grew out of extensive discussions with Nicos Pavlidis and Niall Adams.

References

R. Agrawal. Sample mean based index policies with $O(\log n)$ regret for the multi-armed bandit problem. *Advances in Applied Probability*, 27:1054–1078, 1995.

S. Agrawal and N. Goyal. Analysis of Thompson sampling for the multi-armed bandit problem. arXiv:1111.1797v1, 2011.

J. Asmuth, L. Li, M. L. Littman, A. Nouri, and D. Wingate. A Bayesian sampling approach to exploration in reinforcement learning. In *Proc. 25th Conf. on Uncertainty in Artificial Intelligence*, pages 19–26, Corvallis, Oregon, 2009. AUAI Press.

J.-Y. Audibert and S. Bubeck. Exploration–exploitation tradeoff using variance estimates in multi-armed bandits. *Theoretical Computer Science*, 410:1876–1902, 2009.

J.-Y. Audibert and S. Bubeck. Regret bounds and minimax policies under partial monitoring. *Journal of Machine Learning Research*, 11:2785–2836, 2010.

J.-Y. Audibert, R. Munos, and C. Szepesvári. Tuning bandit algorithms in stochastic environments. In *International Conference on Algorithmic Learning Theory (ALT)*, pages 150–165. Springer, 2007.

P. Auer. Using confidence bounds for exploitation-exploration trade-offs. *Journal of Machine Learning Research*, 3:397–422, 2002.

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

L. Breiman. *Probability*. Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 1992.

M. Brezzi and T.L. Lai. Incomplete learning from endogenous data in dynamic allocation. *Econometrica*, 68:1511–1516, 2000.

N. Cesa-Bianchi and G. Lugosi. *Prediction, Learning, and Games*. Cambridge University Press, 2006.

O. Chapelle and L. Li. An empirical evaluation of Thompson sampling. In *25th Annu. Conf. on Neural Information Processing Systems*, NIPS 2011, Granada, Spain, 2011.

W. Chu, L. Li, L. Reyzin, and R. E. Schapire. Contextual bandits with linear payoff functions. In *14th Internat. Conf. on Artificial Intelligence and Statistics*, AISTATS, Fort Lauderdale, Florida, USA, 2011.

F. Eicker. Asymptotic normality and consistency of the least squares estimators for families of linear regressions. *Annals of Mathematical Statistics*, 34:447–456, 1963.

S. Filippi, O. Cappé, A. Garivier, and C. Szepesvári. Parametric bandits: The generalized linear case. In *24th Annu. Conf. on Neural Information Processing Systems*, pages 586–594. Curran Associates, Inc., 2010.

A. Garivier and O. Cappé. The KL-UCB algorithm for bounded stochastic bandits and beyond. In *24th Annu. Conf. on Learning Theory*, COLT 2011, Budapest, Hungary, 2011.

J. C. Gittins. Bandit processes and dynamic allocation indices. *Journal of the Royal Statistical Society, B*, 41:148–177, 1979.

T. Graepel, J. Q. Candela, T. Borchert, and R. Herbrich. Web-scale Bayesian click-through rate prediction for sponsored search advertising in Microsoft's Bing search engine. In *Proc. of the 27th Internat. Conf. on Machine Learning*, pages 13–20, Haifa, Israel, 2010. Omnipress.

O.-C. Granmo. A Bayesian learning automaton for solving two-armed bernoulli bandit problems. In *7th Internat. Conf. on Machine Learning and Applications*, San Diego, California, USA, 2008. IEEE Computer Society.

L. P. Kaelbling. Associative reinforcement learning: Functions in k-DNF. *Machine Learning*, 15: 279–298, 1994.

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

L. Li, W. Chu, J. Langford, and R. E. Schapire. A contextual-bandit approach to personalized news article recommendation. In *Proc. 19th Internat. Conf. on World Wide Web*, WWW 2010, pages 661–670, Raleigh, North Carolina, USA, 2010. ACM.

L. Li, W. Chu, J. Langford, and X. Wang. Unbiased offline evaluation of contextual-bandit-based news article recommendation algorithms. In *Proc. 4th ACM Internat. Conf. on Web Search and Data Mining*, WSDM '11, pages 297–306, 2011.

M. L. Littman. A generalized reinforcement-learning model: Convergence and applications. In *Proc. 13th Internat. Conf. on Machine Learning*, pages 310–318. Morgan Kaufmann, 1996.

N. Meuleau and P. Bourgine. Exploration of multi-state environments: Local measures and back-propagation of uncertainty. *Machine Learning*, 35:117–154, 1999.

N. G. Pavlidis, D. K. Tasoulis, and D. J. Hand. Simulation studies of multi-armed bandits with covariates. In *Proc. 10th Internat. Conf. on Computer Modelling*, Cambridge, UK, 2008.

S. L. Scott. A modern Bayesian look at the multi-armed bandit. *Applied Stochastic Models in Business and Industry*, 26:639–658, 2010.

S. Singh, T. Jaakkola, M. L. Littman, and C. Szepesvári. Convergence results for single-step on-policy reinforcement-learning algorithms. *Machine Learning*, 39:287–308, 2000.

A. Slivkins. Contextual bandits with similarity information. arXiv:0907.3986v3, 2011.

R. S. Sutton and A. G. Barto. *Reinforcement Learning: An Introduction*. MIT Press, Cambridge, MA, 1998.

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

T. Wang, D. Lizotte, M. Bowling, and D. Schuurmans. Bayesian sparse sampling for on-line reward optimization. In *Proc. 22nd Internat. Conf. on Machine Learning*, pages 956–963, New York, NY, USA, 2005. ACM.

Yahoo! Academic Relations. Yahoo! front page today module user click log dataset, version 1.0, 2011. URL `http://webscope.sandbox.yahoo.com`.

Y. Yang and D. Zhu. Randomized allocation with nonparametric estimation for a multi-armed bandit problem with covariates. *Annals of Statistics*, 30:100–121, 2002.

D. Zwillinger. *CRC Standard Probability and Statistics Tables and Formulae*. CRC Press, 2000.

Journal of Machine Learning Research 13 (2012) 2107-2143 Submitted 5/11; Revised 1/12; Published 6/12

A Comparison of the Lasso and Marginal Regression

Christopher R. Genovese GENOVESE@STAT.CMU.EDU
Jiashun Jin JIASHUN@STAT.CMU.EDU
Larry Wasserman LARRY@STAT.CMU.EDU
Department of Statistics
Carnegie Mellon University
Pittsburgh, PA 15213, USA

Zhigang Yao ZHY16@PITT.EDU
Department of Statistics
University of Pittsburgh
Pittsburgh, PA 15260, USA

Editor: Sathiya Keerthi

Abstract

The lasso is an important method for sparse, high-dimensional regression problems, with efficient algorithms available, a long history of practical success, and a large body of theoretical results supporting and explaining its performance. But even with the best available algorithms, finding the lasso solutions remains a computationally challenging task in cases where the number of covariates vastly exceeds the number of data points.

Marginal regression, where each dependent variable is regressed separately on each covariate, offers a promising alternative in this case because the estimates can be computed roughly two orders faster than the lasso solutions. The question that remains is how the statistical performance of the method compares to that of the lasso in these cases.

In this paper, we study the relative statistical performance of the lasso and marginal regression for sparse, high-dimensional regression problems. We consider the problem of learning which coefficients are non-zero. Our main results are as follows: (i) we compare the conditions under which the lasso and marginal regression guarantee exact recovery in the fixed design, noise free case; (ii) we establish conditions under which marginal regression provides exact recovery with high probability in the fixed design, noise free, random coefficients case; and (iii) we derive rates of convergence for both procedures, where performance is measured by the number of coefficients with incorrect sign, and characterize the regions in the parameter space recovery is and is not possible under this metric.

In light of the computational advantages of marginal regression in very high dimensional problems, our theoretical and simulations results suggest that the procedure merits further study.

Keywords: high-dimensional regression, lasso, phase diagram, regularization

1. Introduction

Consider a regression model,

$$Y = X\beta + z, \tag{1}$$

with response $Y = (Y_1, \ldots, Y_n)^T$, $n \times p$ design matrix X, coefficients $\beta = (\beta_1, \ldots, \beta_p)^T$, and noise variables $z = (z_1, \ldots, z_n)^T$. A central theme in recent work on regression is that sparsity plays a

critical role in effective high-dimensional inference. Loosely speaking, we call the model *sparse* when most of β's components equal 0, and we call it *high dimensional* when $p \gg n$.

An important problem in this context is variable selection: determining which components of β are non-zero. For general β, the problem is underdetermined, but recent results have demonstrated that under particular conditions on X, to be discussed below, sufficient sparsity of β allows (i) exact recovery of β in the noise-free case (Tropp, 2004) and (ii) consistent selection of the non-zero coefficients in the noisy-case (Chen et al., 1998; Cai et al., 2010; Candés and Tao, 2007; Donoho, 2006; Donoho and Elad, 2003; Fan and Lv, 2008; Fuchs, 2005; Knight and Fu, 2000; Meinshausen and Bühlmann, 2006; Tropp, 2004; Wainwright, 2006; Zhao and Yu, 2006; Zou, 2006). Many of these results are based on showing that under sparsity constraints, the lasso—a convex optimization procedure that controls the ℓ^1 norm of the coefficients—has the same solution as an (intractable) combinatorial optimization problem that controls the number of non-zero coefficients.

Recent years, the lasso (Tibshirani, 1996; Chen et al., 1998) has become one of the main practical and theoretical tools for sparse high-dimensional variable selection problems. In the regression problem, the lasso estimator is defined by

$$\widehat{\beta}_{\text{lasso}} = \underset{\beta}{\operatorname{argmin}} \|Y - X\beta\|_2^2 + \lambda\|\beta\|_1, \tag{2}$$

where $\|\beta\|_1 = \sum_j |\beta_j|$ and $\lambda \geq 0$ is a tuning parameter that must be specified. The lasso gives rise to a convex optimization problem and thus is computationally tractable even for moderately large problems. Indeed, the LARS algorithm (Efron et al., 2004) can compute the entire solution path as a function of λ in $O(p^3 + np^2)$ operations. Gradient descent algorithms for the lasso are faster in practice, but have the same computational complexity. The motivation for our study is that when p is very large, finding the lasso solutions is computationally demanding.

Marginal regression, which is also called correlation learning, simple thresholding (Donoho, 2006), and sure screening (Fan and Lv, 2008), is an older and computationally simpler method for variable selection in which the outcome variable is regressed on each covariate separately and the resulting coefficient estimates are screened. To compute the marginal regression estimates for variable selection, we begin by computing the marginal regression coefficients which, assuming X has been standardized, are

$$\widehat{\alpha} \equiv X^T Y.$$

Then, we threshold $\widehat{\alpha}$ using the tuning parameter $t > 0$:

$$\widehat{\beta}_j = \widehat{\alpha}_j 1\{|\widehat{\alpha}_j| \geq t\}. \tag{3}$$

The procedure requires $O(np)$ operations, two orders faster than the lasso for $p \gg n$. This is a decided advantage for marginal regression because the procedure is tractable for much larger problems than is the lasso. The question that remains is how the *statistical* performance of marginal regression compares to that of the lasso. In this paper, we begin to address this question.

We study the relative performance of the lasso and marginal regression for variable selection in three regimes: (a) exact variable selection in the noise-free and noisy cases with fixed design and coefficients, (b) exact variable selection in the noise-free case with fixed design and random coefficients, and (c) statistical variable selection in the noisy case where performance is measured by the number of coefficients with incorrect sign. Our goal is to reopen the case for marginal regression as a plausible alternative to the lasso for large problems. If marginal regression exhibits

comparable statistical performance, theoretically and empirically, then its computational advantages make it a good choice in practice. Put another way: for very high dimensional problems, marginal regression only needs to tie to win.

Our main results are as follows:

- In the fixed design (X fixed), noise free ($z = 0$), and fixed effects (β fixed) case, both procedures guarantee exact reconstruction of $|\operatorname{sgn}\beta|$ under distinct but generally overlapping conditions.

  We analyze these conditions and give examples where each procedure fails while the other succeeds. The lasso has the advantage of providing exact reconstruction for a somewhat larger class of coefficients, but marginal regression has a better tolerance for collinearity and is easier to tune. These results are discussed in Sections 2.

- In the fixed design, noise free, and random effects (β random) case, we give conditions under which marginal regression gives exact reconstruction of $|\operatorname{sgn}\beta|$ with overwhelming probability.

  Our condition is closely related to both the Faithfulness condition (Spirtes et al., 1993; Meinshausen and Bühlmann, 2006) and the Incoherence condition (Donoho and Elad, 2003). The latter depends only on X, making it easy to check in practice, but in controlling the worst case it is quite conservative. The former depends on the unknown β but is less stringent. Our condition strikes a compromise between the two. These results are discussed in Section 3.

- In the random design, noisy, random effects case, we obtain convergence rates of the two procedures in Hamming distance between the sign vectors $\operatorname{sgn}\beta$ and $\operatorname{sgn}\widehat{\beta}$.

  Under a stylized family of signals, we derive a new "phase diagram" that partitions the parameter space into regions in which (i) exact variable selection is possible (asymptotically); (ii) reconstruction of most relevant variables, but not all, is possible; and (iii) successful variable selection is impossible. We show that both the lasso and marginal regression, properly calibrated, perform similarly in each region. These results are described in Section 4.

To support these theoretical results, we also present simulation studies in Section 5. Our simulations show that marginal regression and the lasso perform similarly over a range of parameters in realistic models. Section 6 gives the proofs of all theorems and lemmas in the order they appear.

Notation. For a real number x, let $\operatorname{sgn}(x)$ be -1, 0, or 1 when $x < 0$, $x = 0$, and $x > 0$; and for vector $u, v \in \mathrm{R}^k$, define $\operatorname{sgn}(u) = (\operatorname{sgn}(u_1), \ldots, \operatorname{sgn}(u_k))^T$, and let (u, v) be the inner product. We will use $\|\cdot\|$, with various subscripts, to denote vector and matrix norms, and $|\cdot|$ to represent absolute value, applied component-wise when applied to vectors. With some abuse of notation, we will write $\min u$ ($\min |u|$) to denote the minimum (absolute) component of a vector u. Inequalities between vectors are to be understood component-wise as well.

Consider a sequence of noiseless regression problems with deterministic design matrices, indexed by sample size n,

$$Y^{(n)} = X^{(n)}\beta^{(n)}. \tag{4}$$

Here, $Y^{(n)}$ is an $n \times 1$ response vector, $X^{(n)}$ is an $n \times p^{(n)}$ matrix and $\beta^{(n)}$ is a $p^{(n)} \times 1$ vector, where we typically assume $p^{(n)} \gg n$. We assume that $\beta^{(n)}$ is sparse in the sense that it has $s^{(n)}$ nonzero components where $s^{(n)} \ll p^{(n)}$. By rearranging $\beta^{(n)}$ without loss of generality, we can partition each

$X^{(n)}$ and $\beta^{(n)}$ into "signal" and "noise" pieces, corresponding to the non-zero or zero coefficients, as follows:

$$X^{(n)} = \left(X_S^{(n)}, X_N^{(n)}\right) \qquad \beta^{(n)} = \begin{pmatrix} \beta_S \\ \beta_N \end{pmatrix}.$$

Finally, define the Gram matrix $C^{(n)} = (X^{(n)})^T X^{(n)}$ and partition this as

$$C^{(n)} = \begin{pmatrix} C_{SS}^{(n)} & C_{SN}^{(n)} \\ C_{NS}^{(n)} & C_{NN}^{(n)} \end{pmatrix},$$

where of course $C_{NS}^{(n)} = (C_{SN}^{(n)})^T$. Except in Sections 4–5, we suppose $X^{(n)}$ is normalized so that all diagonal coordinates of $C^{(n)}$ are 1.

These $^{(n)}$ superscripts become tedious, so for the remainder of the paper, we suppress them unless necessary to show variation in n. The quantities X, C, p, s, ρ, as well as the tuning parameters λ (for the lasso; see (2)) and t (for marginal regression; see (3)) are all thus implicitly dependent on n.

2. Noise-Free Conditions for Exact Variable Selection

We restrict our attention to a sequence of regression problems in which the signal (non-zero) components of the coefficient vector have large enough magnitude to be distinguishable from zero. Specifically, assume that $\beta_S \in \mathcal{M}_\rho^s$ for a sequence $\rho(\equiv \rho^{(n)}) > 0$ (and not converging to zero too quickly) with

$$\mathcal{M}_a^k = \left\{ x = (x_1, \ldots, x_k)^T \in \mathrm{R}^k : \ |x_j| \geq a \text{ for all } 1 \leq j \leq k \right\},$$

for positive integer k and $a > 0$. (We use $\mathcal{M}_\rho$ to denote the space $\mathcal{M}_\rho^s \equiv \mathcal{M}_{\rho^{(n)}}^{s^{(n)}}$.)

We will begin by specifying conditions on C, ρ, λ, and t such that in the noise-free case, exact reconstruction of β is possible for the lasso or marginal regression, for all coefficient vectors for which the (non-zero) signal coefficients $\beta_S \in \mathcal{M}_\rho$. These in turn lead to conditions on C, p, s, ρ, λ, and t such that in the case of homoscedastic Gaussian noise, the non-zero coefficients can be selected consistently, meaning that for all sequences $\beta_S^{(n)} \in \mathcal{M}_{\rho^{(n)}}^{s^{(n)}} \equiv \mathcal{M}_\rho$,

$$P\left(\left| \mathrm{sgn}(\widehat{\beta}^{(n)}) \right| = \left| \mathrm{sgn}(\beta^{(n)}) \right| \right) \to 1,$$

as $n \to \infty$. (This property was dubbed *sparsistency* by Pradeep Ravikumar, 2007.) Our goal in this section is to compare the conditions for the two procedures. We focus on the noise-free case, although we comment on the noisy case briefly.

2.1 Exact Reconstruction Conditions for the Lasso in the Noise-Free Case

We begin by reviewing three conditions in the noise-free case that are now standard in the literature on the lasso:

Condition E. *The minimum eigenvalue of C_{SS} is positive.*

Condition I. $\max \left| C_{NS} C_{SS}^{-1} \, \mathrm{sgn}(\beta_S) \right| \leq 1$.

Condition J. $\min \left|\beta_S - \lambda C_{SS}^{-1} \operatorname{sgn}(\beta_S)\right| > 0$.

Because C_{SS} is symmetric and non-negative definite, Condition E is equivalent to C_{SS} being invertible. Later we will strengthen this condition. Condition I is sometimes called the *irrepresentability* condition; note that it depends only on $\operatorname{sgn}\beta$, a fact that will be important later.

For the noise-free case, Wainwright (2006, Lemma 1) proves that Conditions E, I, and J are necessary and sufficient for exact reconstruction of the sign vector, that is, for the existence of a lasso solution $\widehat{\beta}$ such that $\operatorname{sgn}\widehat{\beta} = \operatorname{sgn}\beta$. (See also Zhao and Yu, 2006). Note that this result is stronger than correctly selecting the non-zero coefficients, as it gets the signs correct as well.

It will be useful in what follows to give strong forms of these conditions. Maximizing the left side of Condition I over all 2^s sign patterns gives $\|C_{NS}C_{SS}^{-1}\|_\infty$, the maximum-absolute-row-sum matrix norm. It follows that Condition I holds for all $\beta_S \in \mathcal{M}_\rho$ if and only if $\|C_{NS}C_{SS}^{-1}\|_\infty \leq 1$. Similarly, one way to ensure that Condition J holds over $\mathcal{M}_\rho$ is to require that every component of $\lambda C_{SS}^{-1} \operatorname{sgn}(\beta_S)$ be less than ρ. The maximum component of this vector over $\mathcal{M}_\rho$ equals $\lambda\|C_{SS}^{-1}\|_\infty$, which must be less than ρ. A simpler relation, in terms of the smallest eigenvalue of C_{SS} is

$$\frac{\sqrt{s}}{\operatorname{eigen}_{\min}(C_{SS})} = \sqrt{s}\|C_{SS}^{-1}\|_2 \geq \|C_{SS}^{-1}\|_\infty \geq \|C_{SS}^{-1}\|_2 = \frac{1}{\operatorname{eigen}_{\min}(C_{SS})},$$

where the inequality follows from the symmetry of C_{SS} and standard norm inequalities. This yields the following:

Condition E'. *The minimum eigenvalue of C_{SS} is no less than $\lambda_0 > 0$, where λ_0 does not depend on n.*

Condition I'. $\|C_{NS}C_{SS}^{-1}\|_\infty \leq 1-\eta$, *for $0 < \eta < 1$ small and independent of n.*

Condition J'. $\lambda < \dfrac{\rho}{\|C_{SS}^{-1}\|_\infty}$. *(Under Condition E', we can instead use $\lambda < \rho\frac{\lambda_0}{\sqrt{s}}$.)*

Theorem 1 *In the noise-free case, Conditions E' (or E), I' (or I), and J' imply that for all $\beta_S \in \mathcal{M}_\rho$, there exists a lasso solution $\widehat{\beta}$ with $\operatorname{sgn}(\widehat{\beta}) = \operatorname{sgn}(\beta)$.*

These conditions can be weakened in various ways, but we chose these because they transition nicely to the noisy case. For instance, Wainwright (2006) shows that a slight extension of Conditions E', I', and J' gives sparsistency in the case of homoscedastic Gaussian noise.

2.2 Exact Reconstruction Conditions for Marginal Regression in the Noise-Free Case

As above, define $\widehat{\alpha} = X^T Y$ and $\widehat{\beta}_j = \widehat{\alpha}_j 1\{|\widehat{\alpha}_j| \geq t\}$, $1 \leq j \leq p$. Exact reconstruction for variable selection requires that $\widehat{\beta}_j \neq 0$ whenever $\beta_j \neq 0$, or equivalently $|\widehat{\alpha}_j| \geq t$ whenever $\beta_j \neq 0$. In the literature on causal inference (Spirtes et al., 1993), this assumption has been called *faithfulness* and is also used in Bühlmann et al. (2009) and Fan and Lv (2008). The usual justification for this assumption is that if β is selected at random from some distribution, then faithfulness holds with high probability. Robins et al. (2003) has criticized this assumption because results which hold under faithfulness cannot hold in any uniform sense. We feel that despite the lack of uniformity, it is still useful to investigate results that hold under faithfulness, since as we will show, it holds with high probabilty under weak conditions.

By simple algebra, we have that

$$\widehat{\alpha} = \begin{pmatrix} \widehat{\alpha}_S \\ \widehat{\alpha}_N \end{pmatrix} = \begin{pmatrix} X_S^T X_S \beta_S \\ X_N^T X_S \beta_S \end{pmatrix}.$$

The following condition is thus required to correctly identify the non-zero coefficients:

Condition F. $\max |C_{NS}\beta_S| < \min |C_{SS}\beta_S|$. (5)

Because this is reminiscent of (although distinct from) the faithfulness condition mentioned above, we will refer to Condition F as the *Faithfulness Condition*.

Theorem 2 *Condition F is necessary and sufficient for exact reconstruction to be possible for some* $t > 0$ *with marginal regression.*

Unfortunately, as the next theorem shows, Condition F cannot hold for all $\beta_S \in \mathcal{M}_\rho$. Applying the theorem to C_{SS} shows that for any $\rho > 0$, there exists a $\beta_S \in \mathcal{M}_\rho$ that violates equation (5).

Theorem 3 *Let D be an* $s \times s$ *positive definite, symmetric matrix that is not diagonal. Then for any* $\rho > 0$*, there exists a* $\beta \in \mathcal{M}_\rho^s$ *such that* $\min |D\beta| = 0$.

Despite the seeming pessimism of Theorem 3, the result is not as grim as it seems. Since $C\beta \equiv X^T Y$, the theorem says that if we fix X and let $Y = X\beta$ range through all possible $\beta \in \mathcal{M}_\rho^s$, there exists a Y such that $\min |X^T Y| = 0$. However, to mitigate the pessimism, note that once X and Y are are observed, if we see that $\min |X^T Y| > 0$, we can rule out the result of Theorem 3.

2.3 Comparison of the Conditions for Exact Recovery of Sign Vector in the Noise-Free Case

In this subsection, we use several simple examples to get insight into how the exact-recovery conditions for the lasso and marginal regression compare. The examples illustrate the following points:

- (Examples 1 and 2) The conditions for the two procedures are generally overlapping.
- (Example 3) When $C_{SS} = I$, the lasso conditions are relatively weaker.
- (Example 4) Although the conditions for marginal regression do not hold uniformly over any $\mathcal{M}_\rho$, they have the advantage that they do not require invertibility of C_{SS} and hence are less sensitive to small eigenvalues.

The bottom line is that the two conditions appear to be closely related, and that there are cases where each succeeds while the other fails.

Example 1. For $s = 2$, assume that

$$C_{SS} = \begin{pmatrix} 1 & \rho \\ \rho & 1 \end{pmatrix}, \qquad \beta_S = \begin{pmatrix} 2 \\ 1 \end{pmatrix}.$$

For $a = (a_1, a_2)$ a row of C_{NS}, Conditions I and J require that we choose $\lambda > 0$ small enough so that $|a_1 + a_2| \leq 1 + \rho$, while Condition *F* requires $|2a_1 + a_2| \leq \min\{(2+\rho), |1+2\rho|\}$. For many choices of ρ, both of these inequalities are satisfied (e.g., $\rho = -0.75$). Figure 1 shows the sets of (a_1, a_2)

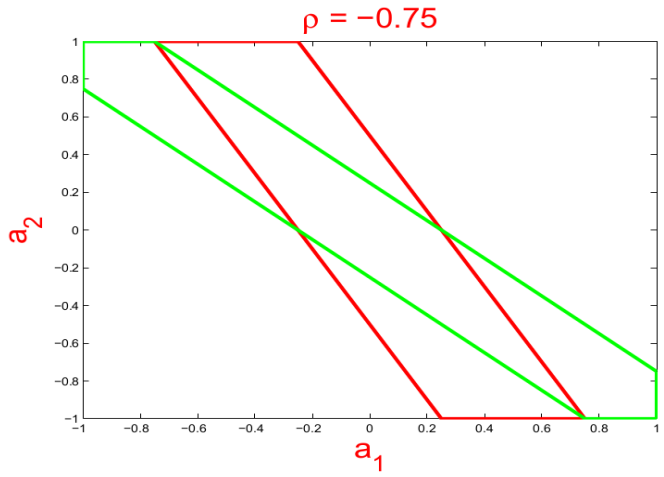


Figure 1: Let C_{SS} and β_S be as in Section 2.3 Example 2, where $\rho = -0.75$. For a row of C_{NS}, say, (a_1, a_2). The interior of the parallelogram and that of the hexagon are the regions of (a_1, a_2) satisfying the conditions of the lasso and marginal regression, respectively; see Example 1.

for which the respective conditions are satisfied. The two regions show significant overlap, and to a large extent, the conditions continue to overlap as ρ and β_S vary. Examples for larger s can be constructed by letting C_{SS} be a block diagonal matrix, where the size of each main diagonal block is small. For each row of C_{NS}, the conditions for the lasso and marginal regression are similar to those above, though more complicated.

Example 2. In the special case where $\beta_S \propto 1_S$, Condition I for the lasso becomes $|C_{NS}C_{SS}^{-1}1_N| \leq 1_N$, where the inequality should be interpreted as holding component-wise, and the condition for marginal regression (Condition F) is $\max\{|C_{NS}1_S|\} \leq \min\{|C_{SS}1_S|\}$. Note that if in addition 1_S is an eigen-vector of C_{SS}, then the two conditions are equivalent to each other. This includes but is not limited to the case of $s = 2$.

Example 3. Fix n and consider the special case in which $C_{SS} = I$. For the lasso, Condition E' (and thus E) is satisfied, Condition J' reduces to $\lambda < \rho$, and Condition I becomes $\|C_{NS}\| \leq 1$. Under these conditions, the lasso gives exact reconstruction, but Condition F can fail. To see how, let $\widetilde{\beta} \in \{-1, 1\}^s$ be the vector such that $\max|C_{NS}\widetilde{\beta}| = \|C_{NS}\|_\infty$ and let ℓ be the index of the row at which the maximum is attained, choosing the row with the biggest absolute element if the maximum is not unique. Let u be the maximum absolute element of row ℓ of C_{NS} with index j. Define a vector δ to be zero except in component j, which has the value $\rho\widetilde{\beta}_j/(u\|C_{NS}\|_\infty)$. Let $\beta = \rho\widetilde{\beta} + \rho\delta$. Then,

$$|(C_{NS}\beta)_\ell| = \rho\left(\|C_{NS}\|_\infty + \frac{1}{\|C_{NS}\|_\infty}\right) > \rho = \min|\beta_S|,$$

so Condition F fails.

On the other hand, suppose Condition F holds for all $\beta_S \in \{-1, 1\}^s$. (It cannot hold for all $\mathcal{M}_\rho$ by Theorem 3). Then, for all $\beta_S \in \{-1, 1\}^s$, $\max|C_{NS}\beta_S| \leq 1$, which implies that $\|C_{NS}\|_\infty \leq 1$.

Choosing $\lambda < \rho$, we have Conditions E', I, and J' satisfied, showing by Theorem 1 that the lasso gives exact reconstruction. It follows that the conditions for the lasso are weaker in this case.

Example 4. For simplicity, assume that $\beta_S \propto 1_S$, although the phenomenon to be described below is not limited to this case. For $1 \le i \le s$, let λ_i and ξ_i be the i-th eigenvalue and eigenvector of C_{SS}. Without loss of generality, we can take ξ_i to have unit ℓ^2 norm. By elementary algebra, there are constants $c_1, \ldots, c_s$ such that $1_S = c_1\xi_1 + c_2\xi_2 + \ldots + c_s\xi_s$. It follows that

$$C_{SS}^{-1} \cdot 1_S = \sum_{i=1}^{s} \frac{c_i}{\lambda_i}\xi_i \qquad \text{and} \qquad C_{SS} \cdot 1_S = \sum_{i=1}^{s} (c_i\lambda_i)\xi_i.$$

Fix a row of C_{NS}, say, $a = (a_1, \ldots, a_s)$. Respectively, the conditions for the lasso and marginal regression require

$$|(a, \sum_{i=1}^{s} \frac{c_i}{\lambda_i}\xi_i)| \le 1 \qquad \text{and} \qquad |(a, 1_S)| \le |\sum_{i=1}^{s} c_i\lambda_i\xi_i|. \tag{6}$$

Without loss of generality, we assume that λ_1 is the smallest eigenvalue of C_{SS}. Consider the case where λ_1 is small, while all other eigenvalues have a magnitude comparable to 1. In this case, the smallness of λ_1 has a negligible effect on $\sum_{i=1}^{s}(c_i\lambda_i)\xi_i$, and so has a negligible effect on the condition for marginal regression. However, the smallness of λ_1 may have an adverse effect on the performance of the lasso. To see the point, we note that $\sum_{i=1}^{s} \frac{c_i}{\lambda_i}\xi_i \approx \frac{c_1}{\lambda_1}\xi_1$. Compare this with the first term in (6). The condition for the lasso is roughly $|(a, \xi_1)| \le c_1\lambda_1$, which is rather restrictive since λ_1 is small.

Figure 2 shows the regions in $a = (a_1, a_2, a_3)$, a row of C_{NS}, where the respective exact recover sequences hold for

$$C_{SS} = \begin{pmatrix} 1 & -1/2 & c \\ -1/2 & 1 & 0 \\ c & 0 & 1 \end{pmatrix}.$$

To better visualize these regions, we display their 2-D section (i.e., setting the first coordinate of a to 0). The Figures suggest that as λ_1 gets smaller, the region corresponding to the lasso shrinks substantially, while that corresponding to marginal regression remains the same.

While the examples in this subsection are low dimensional, they shed light on the high dimensional setting as well. For instance, the approach here can be extended to the following high-dimensional model: (a) $|\mathrm{sgn}(\beta_j)| \overset{iid}{\sim} \mathrm{Bernoulli}(\varepsilon)$, (b) each row of the design matrix X are iid samples from $N(0, \Omega/n)$, where Ω is a $p \times p$ correlation matrix that is sparse in the sense that each row of Ω has relatively few coordinates, and (c) $1 \ll p\varepsilon \ll n \ll p$ (note that $p\varepsilon$ is the expected number of signals). Under this model, it can be shown that (1) C_{SS} is approximately a block-wise diagonal matrix where each block has a relatively small size, and outside these blocks, all coordinates of C_{SS} are uniformly small and have a negligible effect, and (2) each row of C_{NS} has relatively few large coordinates. As a result, the study on the exact reconstruction conditions for the lasso and marginal regression in this more complicated model can be reduced to a low dimensional setting, like those discussed here. And the point that there is no clear winner between the to procedures continues to hold in greater generality.

2.4 Exact Reconstruction Conditions for Marginal Regression in the Noisy Case

We now turn to the noisy case of model (1), taking z to be $N(0, \sigma_n^2 \cdot I_n)$, where we assume that the parameter σ_n^2 is known. The exact reconstruction condition for the lasso in the noisy case has been

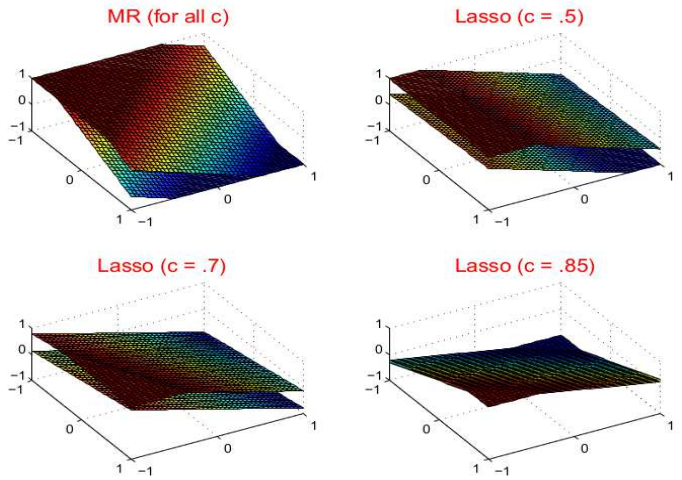


Figure 2: The regions sandwiched by two hyper-planes are the regions of $a = (a_1, a_2, a_3)$ satisfying the respective exact-recovery conditions for marginal regression (MR, panel 1) and for the lasso (panels 2–4). See Section 2.3 Example 4. Here, $c = 0.5, 0.7, 0.85$ and the smallest eigenvalues of C_{SS} are $\lambda_1(c) = 0.29, 0.14, 0.014$. As c varies, the regions for marginal regression remain the same, while the regions for the lasso get substantially smaller.

studied extensively in the literature (see for example Tibshirani, 1996). So in this section, we focus on marginal regression. First, we consider a natural extension of Condition F to the noisy case:

Condition F′. $\max |C_{NS}\beta_S| + 2\sigma_n\sqrt{2\log p} < \min |C_{SS}\beta_S|.$ (7)

Second, when Condition F' holds, we show that with an appropriately chosen threshold t (see (3)), marginal regression fully recovers the support with high probability. Finally, we discuss how to determine the threshold t empirically.

Condition F' implies that it is possible to separate relevant variables from irrelevant variables with high probability. To see this, let $X = [x_1, x_2, \ldots, x_p]$, where x_i denotes the i-th column of X. Sort $|(Y, x_i)|$ in the descending order, and let $r_i = r_i(Y, X)$ be the ranks of $|(Y, x_i)|$ (assume no ties for simplicity). Introduce

$$\widehat{S}_n(k) = \widehat{S}_n(k; X, Y, p) = \{i : \ r_i(X, Y) \le k\}, \qquad k = 1, 2, \ldots, p.$$

Recall that $S(\beta)$ denotes the support of β and $s = |S|$. The following lemma says that, if s is known and Condition F' holds, then marginal regression is able to fully recover the support S with high probability.

Lemma 4 *Consider a sequence of regression models as in (4). If for sufficiently large n, Condition F' holds and $p^{(n)} \ge n$, then*

$$\lim_{n\to\infty} P\left(\widehat{S}_n(s^{(n)}; X^{(n)}, Y^{(n)}, p^{(n)}) \ne S(\beta^{(n)})\right) = 0.$$

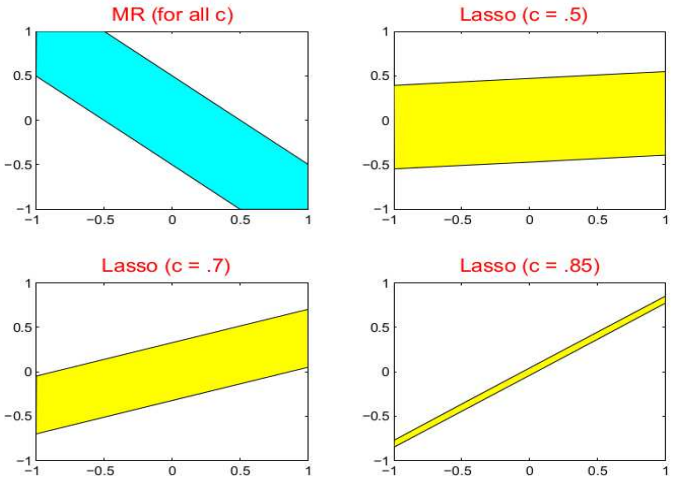


Figure 3: Displayed are the 2-D sections of the regions in Figure 2, where we set the first coordinate of a to 0. As c varies, the regions for marginal regression remain the same, but those for the lasso get substantially smaller as $\lambda_1(c)$ decrease. x-axis: a_2. y-axis: a_3.

Lemma 4 is proved in Section 6. We remark that if both s and $(p-s)$ tend to ∞ as n tends to ∞, then Lemma 4 continues to hold if we replace $2\sigma_n\sqrt{2\log p}$ in (7) by $\sigma_n(\sqrt{\log(p-s)}+\sqrt{\log s})$. See the proof of the lemma for details.

The key assumption of Lemma 4 is that s is known so that we know how to set the threshold t. Unfortunately, s is generally unknown. We propose the following procedure to estimate s. Fix $1 \le k \le p$, let i_k be the unique index satisfying $r_{i_k}(X,Y) = k$. Let $\widehat{V}_n(k) = \widehat{V}_n(k;X,Y,p)$ be the linear space spanned by $x_{i_1}, x_{i_2}, \ldots, x_{i_k}$, and let $\widehat{H}_n(k) = \widehat{H}_n(k;X,Y,p)$ be the projection matrix from R^n to $\widehat{V}_n(k)$ (here and below, the $\widehat{\ }$ sign emphasizes the dependence of indices i_k on the data). Define

$$\widehat{\delta}_n(k) = \widehat{\delta}_n(k;X,Y,p) = \|(\widehat{H}_n(k+1) - \widehat{H}_n(k))Y\|, \qquad 1 \le k \le p-1.$$

The term $\widehat{\delta}_n^2(k)$ is closely related to the F-test for testing whether $\beta_{i_{k+1}} \neq 0$. We estimate s by

$$\widehat{s}_n = \widehat{s}_n(X,Y,p) = \max\left\{1 \le k \le p : \widehat{\delta}_n(k) \ge \sigma_n\sqrt{2\log n}\right\} + 1$$

(in the case where $\widehat{\delta}_n(k) < \sigma_n\sqrt{2\log n}$ for all k, we define $\widehat{s}_n = 1$).

Once $\widehat{s}_n$ is determined, we estimate the support S by

$$\widehat{S}(\widehat{s}_n, X, Y, p) = \{i_k : k = 1, 2, \ldots, \widehat{s}_n\}.$$

It turns out that under mild conditions, $\widehat{s}_n = s$ with high probability. In detail, suppose that the support $S(\beta)$ consists of indices $j_1, j_2, \ldots, j_s$. Fix $1 \le k \le s$. Let $\widetilde{V}_S$ be the linear space spanned by $x_{j_1}, \ldots, x_{j_s}$, and let $\widetilde{V}_{S,(-k)}$ be the linear space spanned by $x_{j_1}, \ldots, x_{j_{k-1}}, x_{j_{k+1}}, \ldots, x_{j_s}$. Project $\beta_{j_k} x_{j_k}$ to the linear space $\widetilde{V}_S \cap \widetilde{V}_{S,(-k)}^{\perp}$. Let $\Delta_n(k, \beta, X, p)$ be the ℓ^2 norm of the resulting vector (which can be interpreted as the strength of the k-th signal after the collinearity between the k-th predictor and

other predictors removed), and let

$$\Delta_n^*(\beta, X, p) = \min_{1 \le k \le s} \Delta_n(k, \beta, X, p).$$

The following theorem says that if $\Delta_n^*(\beta, X, p)$ is slightly larger than $\sigma_n\sqrt{2\log n}$, then $\widehat{s}_n = s$ and $\widehat{S}_n = S$ with high probability. In other words, marginal regression fully recovers the support with high probability. Theorem 5 is proved in Section 6.

Theorem 5 *Consider a sequence of regression models as in (4). Suppose that for sufficiently large n, Condition F' holds,* $p^{(n)} \ge n$, *and*

$$\lim_{n\to\infty}\left(\frac{\Delta_n^*(\beta^{(n)}, X^{(n)}, p^{(n)})}{\sigma_n} - \sqrt{2\log n}\right) = \infty.$$

Then

$$\lim_{n\to\infty} P\left(\widehat{s}_n(X^{(n)}, Y^{(n)}, p^{(n)}) \ne s^{(n)}\right) \to 0,$$

and

$$\lim_{n\to\infty}\left(\widehat{S}_n(\widehat{s}_n(X^{(n)}, Y^{(n)}, p^{(n)}); X^{(n)}, Y^{(n)}, n, p^{(n)}) \ne S(\beta^{(n)})\right) \to 0.$$

Theorem 5 says that the tuning parameter for marginal regression (i.e., the threshold t) can be set successfully in a data driven fashion. In comparison, how to set the tuning parameter λ for the lasso has been a longstanding open problem in the literature.

We briefly discuss the case where the noise variance σ_n^2 is unknown. The topic is addressed in some of recent literature (e.g., Candés and Tao, 2007; Sun and Zhang, 2011). It is noteworthy that in some applications, σ_n^2 can be calibrated during data collection and so it can be assumed as known (Candés and Tao, 2007, Rejoinder). It is also noteworthy that in Sun and Zhang (2011), they proposed a procedure to jointly estimate β and σ_n^2 using scaled lasso. The estimator was show to be consistent with σ_n^2 in rather general situations, but unfortunately it is computationally more expensive than either the lasso or marginal regression. How to find an estimator that is consistent with σ_n^2 in general situations and has low computational cost remains an open problem, and we leave the study to the future.

With that being said, we conclude this section by mentioning that both the lasso and marginal regression have their strengths and weakness, and there is no clear winner between these two in general settings. For a given data set, whether to use one or the other is a case by case decision, where a close investigation of (X, β) is usually necessary.

3. The Deterministic Design, Random Coefficient Regime

In this section, we study how generally the Faithfulness Condition holds. We approach this question by modeling the coefficients β as random (the matrix X remains fixed) and deriving a condition (F") under which the Faithfulness Condition holds with high probability. The discussion in this section is closely related to the work by Donoho and Elad (2003) on the Incoherence condition. Compared to the Faithfulness Condition, the advantage of the Incoherence Condition is that it does not involve the unknown support of β, so it is checkable in practice. The downside of the Incoherence Condition

is that it aims to control the worst case so it is conservative. In this section, we derive a condition—Condition F"—which can be viewed as a middle ground between the Faithfulness Condition and the Incoherence Condition: it is not tied to the unknown support so it is more tractable than the Faithfulness Condition, and it is also much less stringent than the Incoherence Condition.

In detail, we model β as follows. Fix $\varepsilon \in (0,1)$, $a > 0$, and a distribution π, where

$$\text{the support of } \pi \subset (-\infty, -a] \cup [a, \infty). \tag{8}$$

For each $1 \le i \le p$, we draw a sample B_i from Bernoulli(ε). When $B_i = 0$, we set $\beta_i = 0$. When $B_i = 1$, we draw $\beta_i \sim \pi$. Marginally,

$$\beta_i \overset{iid}{\sim} (1-\varepsilon)\nu_0 + \varepsilon\pi, \tag{9}$$

where ν_0 denotes the point mass at 0. This models the case where we have no information on the signals, so they appear at locations generated randomly. In the literature, it is known that the least favorable distribution for variable selection has the form as in (9), where π is in fact degenerate. See Candès and Plan (2009) for example.

We study for which quadruples (X, ε, π, a) the Faithfulness Condition holds with high probability. Recall that the design matrix $X = [x_1, \ldots, x_p]$, where x_i denotes the i-th column. Fix $t \ge 0$ and $\delta > 0$. Introduce

$$g_{ij}(t) = E_\pi[e^{tu\cdot(x_i,x_j)}] - 1, \qquad \bar{g}_i(t) = \sum_{j \ne i} g_{ij}(t),$$

where the random variable $u \sim \pi$ and (x_i, x_j) denotes the inner product of x_i and x_j. As before, we have suppressed the superscript $^{(n)}$ for $g_{ij}(t)$ and $\bar{g}_i(t)$. Define

$$A_n(\delta, \varepsilon, \bar{g}) = A_n(\delta, \varepsilon, \bar{g}; X, \pi) = \min_{t>0}\left(e^{-\delta t} \sum_{i=1}^{p} [e^{\varepsilon \bar{g}_i(t)} + e^{\varepsilon \bar{g}_i(-t)}] \right),$$

where $\bar{g}$ denotes the vector $(\bar{g}_1, \ldots, \bar{g}_p)^T$. Note that $1 + g_{ij}(t)$ is the moment generating function of π evaluated at the point $(x_i, x_j)t$. In the literature, it is conventional to use moment generating function to derive sharp inequalities on the tail probability of sums of independent random variables. The following lemma is proved in Section 6.

Lemma 6 *Fix n, X, $\delta > 0$, $\varepsilon \in (0,1)$, and distribution π. Then*

$$P(\max |C_{NS}\beta_S| \ge \delta) \le (1-\varepsilon) A_n(\delta, \varepsilon, \bar{g}; X, \pi), \tag{10}$$

and

$$P(\max |(C_{SS} - I_S)\beta_S| \ge \delta) \le \varepsilon A_n(\delta, \varepsilon, \bar{g}; X, \pi). \tag{11}$$

Now, suppose the distribution π satisfies (8) for some $a > 0$. Take $\delta = a/2$ on the right hand side of (10)-(11). Except for a probability of $A_n(a/2, \varepsilon, \bar{g})$,

$$\max |C_{NS}\beta_S| \le a/2, \qquad \min |C_{SS}\beta_S| \ge \min |\beta_S| - \max |(C_{SS} - I)\beta_S| \ge a/2,$$

so $\max |C_{NS}\beta_S| \le \min |C_{SS}\beta_S|$ and the Faithfulness Condition holds. This motivates the following condition, where (a, ε, π) may depend on n.

Condition F$''$. $\lim_{n\to\infty} A_n(a_n/2, \varepsilon_n, \bar{g}^{(n)}; X^{(n)}, \pi_n) = 0.$

The following theorem says that if Condition F" holds, then Condition F holds with high probability.

Theorem 7 *Consider a sequence of noise-free regression models as in (4), where the noise component $z^{(n)} = 0$ and $\beta^{(n)}$ is generated as in (9). Suppose Condition F" holds. Then as n tends to ∞, except for a probability that tends to 0,*

$$\max |C_{NS}\beta_S| \leq \min |C_{SS}\beta_S|.$$

Theorem 7 is the direct result of Lemma 6 so we omit the proof.

3.1 Comparison of Condition F" with the Incoherence Condition

Introduced in Donoho and Elad (2003) (see also Donoho and Huo, 2001), the *Incoherence* of a matrix X is defined as

$$\max_{i \neq j} |C_{ij}|,$$

where $C = X^T X$ is the Gram matrix as before. The notion is motivated by the study in recovering a sparse signal from an over-complete dictionary. In the special case where X is the concatenation of two orthonormal bases (e.g., a Fourier basis and a wavelet basis), $\max_{i \neq j} |C_{ij}|$ measures how coherent two bases are and so the term of incoherence; see Donoho and Elad (2003) and Donoho and Huo (2001) for details. Consider Model (1) in the case where both X and β are deterministic, and the noise component $z = 0$. The following results are proved in Chen et al. (1998), Donoho and Elad (2003) and Donoho and Huo (2001).

- The lasso yields exact variable selection if $s < \frac{1+\max_{i \neq j} |C_{ij}|}{2 \max_{i \neq j} |C_{ij}|}$.
- Marginal regression yields exact variable selection if $s < \frac{c}{2 \max_{i \neq j} |C_{ij}|}$ for some constant $c \in (0,1)$, and that the nonzero coordinates of β have comparable magnitudes (i.e., the ratio between the largest and the smallest nonzero coordinate of β is bounded away from ∞).

In comparison, the Incoherence Condition only depends on X so it is checkable. Condition F depends on the unknown support of β. Checking such a condition is almost as hard as estimating the support S. Condition F" provides a middle ground. It depends on β only through (ε, π). In cases where we either have a good knowledge of (ε, π) or we can estimate them, Condition F" is checkable (for literature on estimating (ε, π), see Jin, 2007 and Wasserman, 2006 for the case where we have an orthogonal design, and Ji and Jin, 2012, Section 2.6 for the case where $X^T X$ is sparse in the sense that each row of $X^T X$ has relatively few large coordinates. We note that even when successful variable selection is impossible, it may be still possible to estimate (ε, π) well).

At the same time, the Incoherence Condition is conservative, especially when s is large. In fact, in order for either the lasso or marginal regression to have an exact variable selection, it is required that

$$\max_{i \neq j} |C_{ij}| \leq O\left(\frac{1}{s}\right),$$

In other words, all coordinates of the Gram matrix C need to be no greater than $O(1/s)$. This is much more conservative than Condition F.

However, we must note that the Incoherence Condition aims to control the worst case: it sets out to guarantee *uniform* success of a procedure across all β under minimum constraints. In comparison, Condition F aims to control a single case, and Condition F" aims to control almost all the cases in a specified class. As such, Condition F" provides a middle ground between Condition F and the

Incoherence Condition, applying more broadly than the former, while being less conservative than the later.

Below, we use two examples to illustrate that Condition F" is much less conservative than the Incoherence Condition. In the first example, we consider a weakly dependent case where $\max_{i\neq j}|C_{ij}| \leq O(1/\log(p))$. In the second example, we suppose the matrix C is sparse, but the nonzero coordinates of C may be large.

3.1.1 THE WEAKLY DEPENDENT CASE

Suppose that for sufficiently large n, there are two sequence of positive numbers $a_n \leq b_n$ such that the support of π_n is contained in $[-b_n,-a_n]\cup[a_n,b_n]$, and that

$$\frac{b_n}{a_n}\cdot\max_{i\neq j}|C_{ij}| \leq c_1/\log(p), \qquad c_1>0 \text{ is a constant.}$$

For $k\geq 1$, denote the k-th moment of π_n by

$$\mu_n^{(k)} = \mu_n^{(k)}(\pi_n).$$

Introduce $m_n = m_n(X)$ and $v_n^2 = v_n^2(X)$ by

$$m_n(X) = p\varepsilon_n\cdot\max_{1\leq i\leq p}\left\{\left|\frac{1}{p}\sum_{j\neq i}C_{ij}\right|\right\}, \qquad v_n^2(X) = p\varepsilon_n\cdot\max_{1\leq i\leq p}\left\{\frac{1}{p}\sum_{j\neq i}C_{ij}^2\right\}.$$

Corollary 3.1 *Consider a sequence of regression models as in (4), where the noise component $z^{(n)}=0$ and $\beta^{(n)}$ is generated as in (9). If there are constants $c_1>0$ and $c_2\in(0,1/2)$ such that*

$$\frac{b_n}{a_n}\cdot\max_{i\neq j}\{|C_{ij}|\} \leq c_1/\log(p^{(n)}),$$

and

$$\overline{\lim_{n\to\infty}}\left(\frac{\mu_n^{(1)}(\pi_n)}{a_n}m_n(X^{(n)})\right)\leq c_2, \qquad \overline{\lim_{n\to\infty}}\left(\frac{\mu_n^{(2)}(\pi_n)}{a_n^2}v_n^2(X^{(n)})\log(p^{(n)})\right)=0, \tag{12}$$

then

$$\lim_{n\to\infty}A_n(a_n/2,\varepsilon_n,\bar{g}^{(n)};X^{(n)},\pi_n)=0,$$

and Condition F" holds.

Corollary 3.1 is proved in Section 6. For interpretation, we consider the special case where there is a generic constant $c>0$ such that $b_n\leq ca_n$. As a result, $\mu_n^{(1)}/a_n\leq c$, $\mu_n^{(2)}/a_n^2\leq c^2$. The conditions reduce to that, for sufficiently large n and all $1\leq i\leq p$,

$$|\frac{1}{p}\sum_{j\neq i}^{p}C_{ij}|\leq O(\frac{1}{p\varepsilon_n}), \qquad \frac{1}{p}\sum_{j\neq i}^{p}C_{ij}^2 = o(1/p\varepsilon_n).$$

Note that by (9), $s=s^{(n)}\sim\text{Binomial}(p,\varepsilon_n)$, so $s\approx p\varepsilon_n$. Recall that the Incoherence Condition is

$$\max_{i\neq j}|C_{ij}|\leq O(1/s).$$

In comparison, the Incoherence Condition requires that each coordinate of $(C-I)$ is no greater than $O(1/s)$, while Condition F" only requires that the average of each row of $(C-I)$ is no greater than $O(1/s)$. The latter is much less conservative.

3.1.2 THE SPARSE CASE

Let $N_n^*(C)$ be the maximum number of nonzero off-diagonal coordinates of C:

$$N_n^*(C) = \max_{1\le i\le p}\{N_n(i)\}, \qquad N_n(i) = N_n(i;C) = \#\{j:\ j\neq i, C_{ij}\neq 0\}.$$

Suppose there is a constant $c_3 > 0$ such that

$$\underline{\lim}_{n\to\infty}\left(\frac{-\log(\varepsilon_n N_n^*(C))}{\log(p^{(n)})}\right) \ge c_3. \tag{13}$$

Also, suppose there is a constant $c_4 > 1$ such that for sufficiently large n,

$$\text{the support of } \pi_n \text{ is contained in } [-c_4 a_n, a_n]\cup[a_n, c_4 a_n]. \tag{14}$$

The following corollary is proved in Section 6.

Corollary 3.2 *Consider a sequence of noise-free regression models as in (4), where the noise component $z^{(n)} = 0$ and $\beta^{(n)}$ is randomly generated as in (9). Suppose (13)-(14) hold. If there is a constant $\delta > 0$ such that*

$$\max_{i\neq j}|C_{ij}| \le \delta, \qquad \textit{and} \qquad \delta < \frac{c_3}{2c_4},$$

then

$$\lim_{n\to\infty} A_n(a_n/2, \varepsilon_n, \bar{g}^{(n)}; X^{(n)}, \pi_n) = 0,$$

and Condition F" holds.

For interpretation, consider a special case where $\varepsilon_n = p^{-\vartheta}$. In this case, the condition reduces to $N_n^*(C) \ll p^{\vartheta - 2c_4\delta}$. As a result, Condition F" is satisfied if each row of $(C-I)$ contains no more than $p^{\vartheta-2c_4\delta}$ nonzero coordinates each of which $\le \delta$. Compared to the Incoherence Condition $\max_{i\neq j}|C_{ij}| \le O(1/s) = O(p^{-\vartheta})$, our condition is much weaker.

In conclusion, if we alter our attention from the worst-case scenario to the average scenario, and alter our aim from exact variable selection to exact variable selection with probability ≈ 1, then the condition required for success—Condition F"—is much more relaxed than the Incoherence Condition.

4. Hamming Distance for the Gaussian Design and the Phase Diagram

So far, we have focused on exact variable selection. In many applications, exact variable selection is not possible. Therefore, it is of interest to study the Type I and Type II errors of variable selection (a Type I error is a misclassified 0 coordinate of β, and a Type II error is a misclassified nonzero coordinate).

In this section, we use the Hamming distance to measure the variable selection errors. Back to Model (1),

$$Y = X\beta + z, \qquad z\sim N(0, I_n), \tag{15}$$

where without loss of generality, we assume $\sigma_n = 1$. As in the preceding section (i.e., (9)), we suppose

$$\beta_i \overset{iid}{\sim} (1-\varepsilon)\nu_0 + \varepsilon\pi.$$

For any variable selection procedure $\widehat{\beta} = \widehat{\beta}(Y;X)$, the Hamming distance between $\widehat{\beta}$ and the true β is

$$d(\widehat{\beta}|X) = d(\widehat{\beta};\varepsilon,\pi|X) = \sum_{j=1}^{p} E_{\varepsilon,\pi}(E_z[1(\mathrm{sgn}(\widehat{\beta}_j) \neq \mathrm{sgn}(\beta_j))|X]).$$

Note that by Chebyshev's inequality,

$$P(\text{non-exact variable selection by } \widehat{\beta}(Y;X)) \leq d(\widehat{\beta}|X).$$

So a small Hamming distance guarantees exact variable selection with high probability.

How to characterize precisely the Hamming distance is a challenging problem. We approach this by modeling X as random. Assume that the coordinates of X are iid samples from $N(0,1/n)$:

$$X_{ij} \overset{iid}{\sim} N(0,1/n). \tag{16}$$

The choice of the variance ensures that most diagonal coordinates of the Gram matrix $C = X^T X$ are approximately 1. Let $P_X(x)$ denote the joint density of the coordinates of X. The expected Hamming distance is then

$$d^*(\widehat{\beta}) = d^*(\widehat{\beta};\varepsilon,\pi) = \int d(\widehat{\beta};\varepsilon,\pi|X=x)P_X(x)dx.$$

We adopt an asymptotic framework where we calibrate p and ε with

$$p = n^{1/\theta}, \qquad p\varepsilon_n = n^{(1-\vartheta)/\theta} \equiv p^{1-\vartheta}, \qquad 0 < \theta, \vartheta < 1. \tag{17}$$

This models a situation where $p \gg n$ and the vector β gets increasingly sparse as n grows. Note that the parameter ϑ calibrates the sparsity level of the signals. We assume π_n in (9) is a point mass

$$\pi_n = \nu_{\tau_n}. \tag{18}$$

In the literature (e.g., Donoho and Jin, 2004; Meinshausen and Rice, 2006), this model was found to be subtle and rich in theory. In addition, compare two experiments, in one of them $\pi_n = \nu_{\tau_n}$, and in the other the support of π_n is contained in $[\tau_n,\infty)$. Since the second model is easier for inference than the first one, the optimal Hamming distance for the first one gives an upper bound for that for the second one.

With ε_n calibrated as above, the most interesting range for τ_n is $O(\sqrt{2\log p})$: when $\tau_n \gg \sqrt{2\log p}$, exact variable selection can be easily achieved by either the lasso or marginal regression. When $\tau_n \ll \sqrt{2\log p}$, no variable selection procedure can achieve exact variable selection. See, for example, Donoho and Jin (2004). In light of this, we calibrate

$$\tau_n = \sqrt{2(r/\theta)\log n} \equiv \sqrt{2r\log p}, \qquad r > 0. \tag{19}$$

Note that the parameter r calibrates the signal strength. With these calibrations, we can rewrite

$$d_n^*(\widehat{\beta};\varepsilon,\pi) = d_n^*(\widehat{\beta};\varepsilon_n,\tau_n).$$

Definition 8 *Denote $L(n)$ by a multi-log term which satisfies that $\lim_{n\to\infty}(L(n)\cdot n^{\delta}) = \infty$ and that $\lim_{n\to\infty}(L(n)\cdot n^{-\delta}) = 0$ for any $\delta > 0$.*

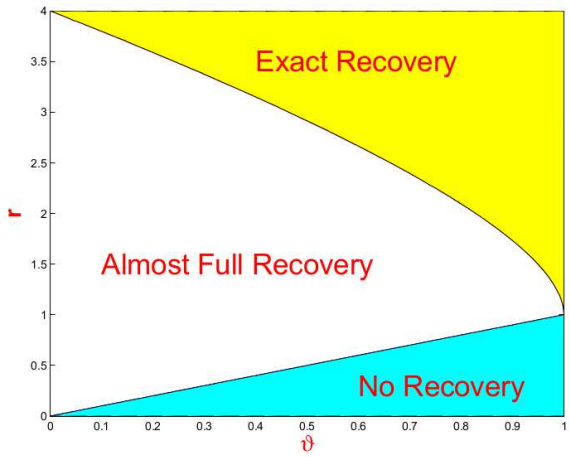


Figure 4: The regions as described in Section 4. In the region of Exact Recovery, both the lasso and marginal regression yield exact recovery with high probability. In the region of Almost Full Recovery, it is impossible to have large probability for exact variable selection, but the Hamming distance of both the lasso and marginal regression $\ll p\varepsilon_n$. In the region of No Recovery, optimal Hamming distance $\sim p\varepsilon_n$ and all variable selection procedures fail completely. Displayed is the part of the plane corresponding to $0 < r < 4$ only.

We are now ready to spell out the main results. Define

$$\rho(\vartheta) = (1+\sqrt{1-\vartheta})^2, \qquad 0 < \vartheta < 1.$$

The following theorem is proved in Section 6, which gives the lower bound for the Hamming distance.

Theorem 9 *Fix $\vartheta \in (0,1)$, $\theta > 0$, and $r > 0$ such that $\theta > 2(1-\vartheta)$. Consider a sequence of regression models as in (15)-(19). As $n \to \infty$, for any variable selection procedure $\widehat{\beta}^{(n)}$,*

$$d_n^*(\widehat{\beta}^{(n)}; \varepsilon_n, \tau_n) \geq \begin{cases} L(n)p^{1-\frac{(\vartheta+r)^2}{4r}}, & r \geq \vartheta, \\ (1+o(1))\cdot p^{1-\vartheta}, & 0 < r < \vartheta. \end{cases}$$

Let $\widehat{\beta}_{mr}$ be the estimate of using marginal regression with threshold

$$t_n = \begin{cases} \frac{\vartheta+r}{2\sqrt{r}}\sqrt{2\log p}, & \text{if } r > \vartheta, \\ t_n = \sqrt{2\widetilde{r}\log p}, & \text{if } r < \vartheta, \end{cases} \tag{20}$$

where $\widetilde{r}$ is some constant $\in (\vartheta, 1)$ (note that in the case of $r < \vartheta$, the choice of t_n is not necessarily unique). We have the following theorem.

Theorem 10 *Fix $\vartheta \in (0,1)$, $r > 0$, and $\theta > (1-\vartheta)$. Consider a sequence of regression models as in (15)-(19). As $p \to \infty$, the Hamming distance of marginal regression with the threshold t_n given in*

(20) satisfies

$$d_n^*(\widehat{\beta}_{mr}^{(n)};\varepsilon_n,\tau_n) \le \begin{cases} L(n)p^{1-\frac{(\vartheta+r)^2}{4r}}, & r \ge \vartheta, \\ (1+o(1))\cdot p^{1-\vartheta}, & 0 < r < \vartheta. \end{cases}$$

In practice, the parameters (ϑ, r) are usually unknown, and it is desirable to set t_n in a data-driven fashion. Towards this end, we note that our primary interest is in the case of $r > \vartheta$ (as when $r < \vartheta$, successful variable selection is impossible). In this case, the optimal choice of t_n is $(\vartheta + r)/(2r)\tau_p$, which is the Bayes threshold in the literature. The Bayes threshold can be set by the approach of controlling the local False Discovery Rate (Lfdr), where we set the FDR-control parameter as $1/2$; see Efron et al. (2001) for details.

Similarly, choosing the tuning parameter $\lambda_n = 2(\frac{\vartheta+r}{2\sqrt{r}} \wedge \sqrt{r})\sqrt{2\log p}$ in the lasso, we have the following theorem.

Theorem 11 *Fix $\vartheta \in (0,1)$, $r > 0$, and $\theta > (1-\vartheta)$. Consider a sequence of regression models as in (15)-(19). As $p \to \infty$, the Hamming distance of the lasso with the tuning parameter $\lambda_n = 2t_n$ where t_n is given in (20), satisfies*

$$d_n^*(\widehat{\beta}_{lasso}^{(n)};\varepsilon_n,\tau_n) \le \begin{cases} L(n)p^{1-\frac{(\vartheta+r)^2}{4r}}, & r \ge \vartheta, \\ (1+o(1))\cdot p^{1-\vartheta}, & 0 < r < \vartheta. \end{cases}$$

The proofs of Theorems 10-11 are routine and we omit them.

Theorems 9-11 say that in the ϑ-r plane, we have three different regions, as displayed in Figure 4.

- Region I (*Exact Recovery*): $0 < \vartheta < 1$ and $r > \rho(\vartheta)$.
- Region II (*Almost Full Recovery*): $0 < \vartheta < 1$ and $\vartheta < r < \rho(\vartheta)$.
- Region III (*No Recovery*): $0 < \vartheta < 1$ and $0 < r < \vartheta$.

In the Region of Exact Recovery, the Hamming distance for both marginal regression and the lasso are algebraically small. Therefore, except for a probability that is algebraically small, both marginal regression and the lasso give exact recovery.

In the Region of Almost Full Recovery, both the Hamming distance of marginal regression and the lasso are much smaller than the number of relevant variables (which $\approx p\varepsilon_n$). Therefore, almost all relevant variables have been recovered. Note also that the number of misclassified irrelevant variables is comparably much smaller than $p\varepsilon_n$. In this region, the optimal Hamming distance is algebraically large, so for any variable selection procedure, the probability of exact recovery is algebraically small.

In the Region of No Recovery, the Hamming distance $\sim p\varepsilon_n$. In this region, asymptotically, it is impossible to distinguish relevant variables from irrelevant variables, and any variable selection procedure fails completely.

In practice, given a data set, one wishes to know that which of these three regions the true parameters belong to. Towards this end, we note that in the current model, the coordinates of X^TY are approximately iid samples from the following two-component Gaussian mixture

$$(1-\varepsilon_p)\phi(x) + \varepsilon_n\phi(x-\tau_n),$$

where $\phi(x)$ denotes the density of $N(0,1)$. In principle, the parameters (ε_n, τ_n) can be estimated (see the comments we made in Section 3.1 on estimating (ε, π)). The estimation can then be used to determine which regions the true parameters belong to.

	$k = 4$		$k = 10$	
(a_1, a_2)	lasso	MR	lasso	MR
(0, 0)	0	0	0.8	3.8
(-0.85, 0.85)	0	4	0.6	10.4
(0.85, -0.85)	0	4	0.6	11.2
(-0.4, 0.8)	4	0	10	3.6
(0.4, -0.8)	4	0	10	4.8

Table 1: Comparison of the lasso and marginal regression for different choices of (a_1, a_2) and k. The setting is described in Experiment 1a. Each cell displays the corresponding Hamming error.

The results improve on those by Wainwright (2006). It was shown in Wainwright (2006) that there are constants $c_2 > c_1 > 0$ such that in the region of $\{0 < \vartheta < 1, r > c_2\}$, the lasso yields exact variable selection with overwhelming probability, and that in the region of $\{0 < \vartheta < 1, r < c_2\}$, no procedure could yield exact variable selection. Our results not only provide the exact rate of the Hamming distance, but also tighten the constants c_1 and c_2 so that $c_1 = c_2 = (1 + \sqrt{1-\vartheta})^2$. The lower bound argument in Theorem 9 is based on computing the L^1-distance. This gives better results than in Wainwright (2006) which uses Fano's inequality in deriving the lower bounds.

To conclude this section, we briefly comment on the phase diagram in two closely related settings. In the first setting, we replace the identity matrix I_p in (16) by some general correlation matrix Ω, but keep all other assumptions unchanged. In the second setting, we assume that as $n \to \infty$, both ratios $p\varepsilon_p/n$ and n/p tend to a constant in $(0,1)$, while all other assumptions remain the same. For the first setting, it was shown in Ji and Jin (2012) that the phase diagram remains the same as in the case of $\Omega = I_p$, provided that Ω is sparse; see Ji and Jin (2012) for details. For the second setting, the study is more more delicate, so we leave it to the future work.

5. Simulations and Examples

We conducted a small-scale simulation study to compare the performance of the lasso and marginal regression. The study includes three different experiments (some have more than one sub-experiments). In the first experiment, the rows of X are generated from $N(0, \frac{1}{n}C)$ where C is a diagonal block-wise matrix. In the second one, we take the Gram matrix $C = X'X$ to be a tridiagonal matrix. In the third one, the Gram matrix has the form of $C = \Lambda + a\xi\xi'$ where Λ is a diagonal matrix, $a > 0$, and ξ is a $p \times 1$ unit-norm vector. Intrinsically, the first two are covered in the theoretic discussion in Section 2.3, but the last one goes beyond that. Below, we describe each of these experiments in detail.

Experiment 1. In this experiment, we compare the performance of the lasso and marginal regression with the noiseless linear model $Y = X\beta$. We generate the rows of X as iid samples from

Method	(a_2, a_3)	$k=2$ $c=0.5$	$k=2$ $c=0.7$	$k=2$ $c=0.85$	$k=7$ $c=0.5$	$k=7$ $c=0.7$	$k=7$ $c=0.85$
MR	(0,0)	0	0	0	3	3.8	4.6
Lasso	(0, 0)	0	0	2	0	0	7
MR	(-0.4, -0.1)	1	1	1	5.4	5.8	5.4
Lasso	(-0.4, -0.1)	0	0	2	0.4	2	7
MR	(0.4, 0.1)	1	1.2	1.2	5.4	5.8	6
Lasso	(0.4, 0.1)	0	0	2	1.2	1.4	7.6
MR	(-0.5, -0.4)	2	2	2	9.6	7.8	7.6
Lasso	(-0.5, -0.4)	1	0	2	3.6	0.2	7
MR	(0.5, 0.4)	2	2	2	9.4	7.4	7.8
Lasso	(0.5, 0.4)	1	0	2	3.4	0	7

Table 2: Comparison of the lasso and marginal regression for different choices of (c, a_2, a_3). The setting is described in Experiment 1b. Each cell displays the corresponding Hamming error.

$N(0,(1/n)C)$, where C is a diagonal block-wise correlation matrix having the form

$$C = \begin{pmatrix} C_{\text{sub}} & 0 & 0 & \dots 0 \\ 0 & C_{\text{sub}} & 0 & \dots 0 \\ & \cdots & \cdots & \\ 0 & 0 & 0 & \dots C_{\text{sub}} \end{pmatrix}.$$

Fixing a small integer m, we take C_{sub} to be the $m \times m$ matrix as follows:

$$C_{\text{sub}} = \begin{pmatrix} D & a^T \\ a & 1 \end{pmatrix},$$

where a is an $(m-1)\times 1$ vector and D is an $(m-1)\times(m-1)$ matrix to be introduced below. Also, fixing another integer $k \geq 1$, according to the block-wise structure of C, we let β be the vector (without loss of generality, we assume p is divisible by m)

$$\beta = (\delta_1 u^T, \delta_2 u^T, \dots, \delta_{p/m} u^T)^T,$$

where $u = (v^T, 0)$ for some $(m-1)\times 1$ vector v and $\delta_i = 0$ for all but k different i, where $\delta_i = 1$.

The goal of this experiment is to investigate how the theoretic results in Section 2.3 shed light on models with more practical interests. To see the point, note that when $k \ll n$, the signal vector β is sparse, and we expect to see that

$$X'X\beta \approx C\beta, \tag{21}$$

where the right hand side corresponds to the idealized model where $X'X = C$. In this idealized model, if we restrict our attention to any block where the corresponding δ_i is 1, then we have exactly the same model as in Example 1 of Section 2.3, with $C_{SS} = D$ and $\beta_S = v$. As a result, the theoretic results discussed in Section 2.3 apply, at least when the approximation error in Equation (21) is negligible. Experiment 1 contains two sub-experiments, Experiment 1a and 1b.

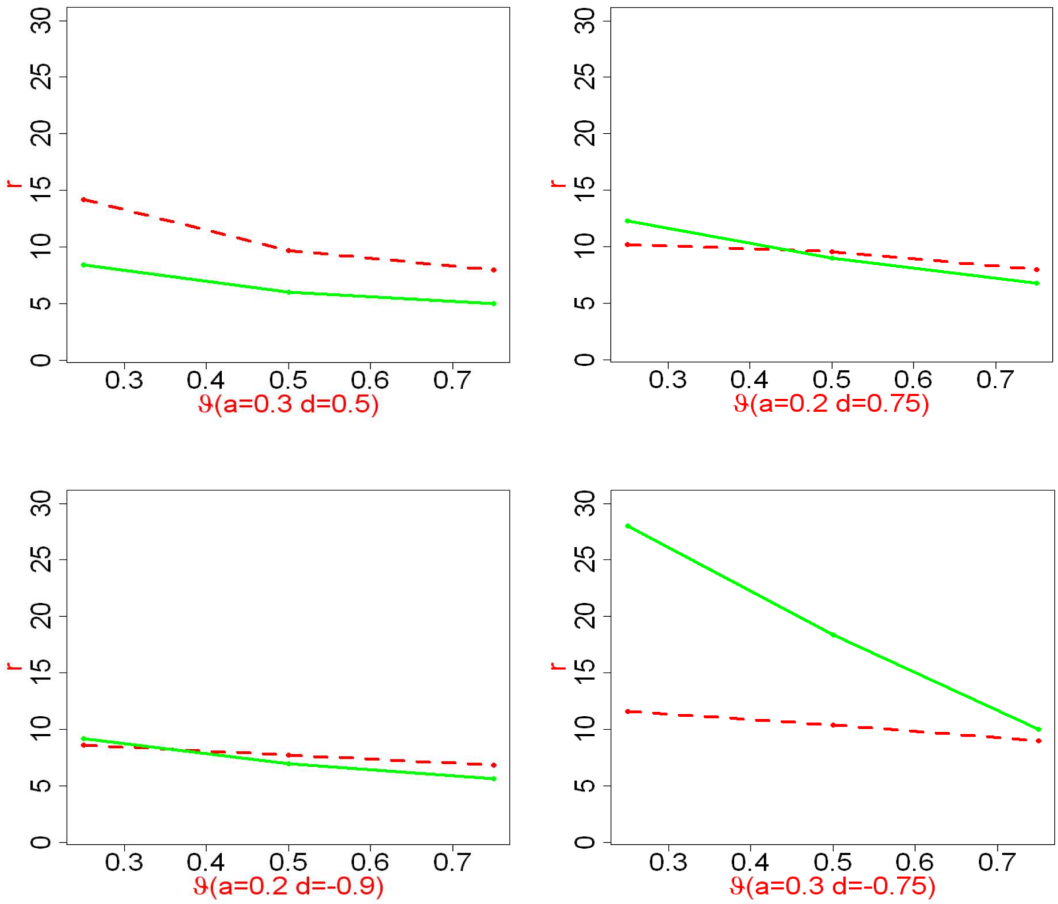


Figure 5: Critical values of exact recovery for the lasso (dashed) and marginal regression (solid). See Experiment 2 for the setting and the definition of critical value. For any given set of parameters (ϑ, a, d), the method with a smaller critical value has the better performance in terms of Hamming errors.

In Experiment 1a, we take $(p,n,m) = (999,900,3)$. At the same time, for some numbers a_1 and a_2, we set a, v, and D by

$$a = (a_1, a_2)^T, \qquad v = (2,1)^T, \qquad D = \begin{pmatrix} 1 & -.75 \\ -.75 & 1 \end{pmatrix}.$$

We investigate the experiment with two different values of k ($k = 4$ and $k = 10$) and five different choices of (a_1, a_2): $(0,0), \pm(-0.85, 0.85)$, and $\pm(-0.4, 0.8)$. When $k = 4$, we let $\delta_i = 1$ if and only if $i \in \{40, 208, 224, 302\}$, and when $k = 10$, we let $\delta_i = 1$ if and only if $i \in \{20, 47, 83, 86, 119, 123, 141, 250, 252, 281\}$ (such indices are generated randomly; also, note that i are the indices for the blocks, not the indices for the signals).

Consider for a second the idealized case where $X'X = C$ (i.e., n is very large). If we restrict our attention to any block of β where the corresponding δ_i is 1, the setting reduces to that of Example 1 of Section 2.3. In fact, in Figure 1, our first choice of (a_1, a_2) falls inside both the parallelogram and hexagon, our next two choices fall inside the hexagon but outside the parallelogram, and our last two choices fall outside the hexagon but inside the parallelogram. Therefore, at least when k is sufficiently small (so that the setting can be well-approximated by that in the idealized case), we expect to see that the lasso outperforms marginal regression with the second and the third choices, and expect to see the other way around with the last two choices of (a_1, a_2). In the first choice, both methods are expected to perform well.

We now investigate how well these expectations are met. For each combination of these parameters, we generate data and compare the Hamming errors of the lasso and marginal regression, where for each method, the tuning parameters are set ideally. The 'ideal' tuning parameter is obtained through rigorous search from a range. The error rates over 10 repetitions are tabulated in Table 1. More repetitions is unnecessary, partially because the standard deviations of the simulation results are small, and partially because the program is slow (for that we need to choose the 'ideal' tuning parameter through rigorous search. Take the lasso for example. For rigorous search of the 'ideal' tuning parameter, we need to run the glmnet R package many times).

The results suggest that the performances of each method are reasonably close to what are expected for the idealized model, especially in the case of $k = 4$. Take the cases $(a_1, a_2) = \pm(0.85, -0.85)$ for example. The lasso yields exact recovery, while marginal regression, in each of the four blocks where the corresponding δ_i is 1, recovers correctly the stronger signal and mistakenly kills the weaker one. The situation is reversed in the cases where $(a_1, a_2) = \pm(0.4, -0.8)$. The discussion for the case of $k = 10$ is similar, but the approximation error in Equation (21) starts to kick in.

In Experiment 1b, we take $(p, n, m) = (900, 1000, 4)$. Also, for some numbers c, a_2, and a_3, we set a, v, and D as

$$a^T = (0, a_2, a_3)^T, \qquad v = (1,1,1)^T, \qquad D = \begin{pmatrix} 1 & -1/2 & c \\ -1/2 & 1 & 0 \\ c & 0 & 1 \end{pmatrix}.$$

The primary goal of this experiment is to investigate how different choices of c affect the performance of the lasso and marginal regression. To see the point, note that in the idealized situation where $X'X = C$, the model reduces to the one discussed in Figure 3, if we restrict our attention to any block of β where $\delta_i = 1$. The theoretic results in Example 4 of Section 2.3 predict that, the performance of the lasso gets increasingly unsatisfactory as c increases, while that of marginal regression stay more or less the same. At the same time, which of this method performs better depends on (a_2, a_3, c), see Figure 3 for details.

We select two different k for experiment: $k = 2$ and $k = 7$. When $k = 2$, we let $\delta_i = 1$ if and only if $i \in \{60, 139\}$, and when $k = 7$, we let $\delta_i = 1$ if and only if $i \in \{34, 44, 58, 91, 100, 183, 229\}$. Also, we investigate five different choices of (a_2, a_3): $(0,0)$, $(0,0)$, $\mp(0.4, 0.1)$, and $\mp(0.5, 0.4)$, and three different c: $c = 0.5, 0.7$, and 0.85. For each combination of these parameters, we apply both the lasso and marginal regression and obtain the Hamming errors of both methods, where similarly, the tuning parameters for each method are set ideally. The error rates over 10 repetitions are tabulated in Table 2. The results suggest that different choices of c have a major role over the lasso, but does

not have a big influence over marginal regression. The results fit well with the theory illustrated in Section 2.3; see Figure 3 for comparisons.

Experiment 2. In this experiment, we use the linear regression model $Y = X\beta + z$ where $z \sim N(0, I_n)$. We use a different criterion rather than the Hamming errors to compare two methods: with the same parameter settings, the method that yields exact recovery in a larger range of parameters is better. Towards this end, we take $p = n = 500$, and $X = \Omega^{1/2}$, where Ω is the $p \times p$ tridiagonal matrix satisfying

$$\Omega(i, j) = 1\{i = j\} + a \cdot 1\{|i - j| = 1\},$$

and the parameter $a \in (-1/2, 1/2)$ so the matrix is positive definite. At the same time, we generate β as follows. Let ϑ range between 0.25 and 0.75 with an increment of 0.25. For each ϑ, let s be the smallest even number $\geq p^{1-\vartheta}$. We then randomly pick $s/2$ indices $i_1 < i_2 < \ldots < i_{s/2}$. For parameters $r > 0$ and $d \in (-1, 1)$ to be determined, we let $\tau = \sqrt{2r \log p}$ and let $\beta_j = \tau$ if $j \in \{i_1, i_2, \ldots, i_{s/2}\}$, $\beta_j = d\tau$ if $j - 1 \in \{i_1, i_2, \ldots, i_{s/2}\}$, and $\beta_j = 0$ otherwise.

To gain insight on how two procedures perform in this setting, we consider the noiseless counterpart for just a second. Without loss of generality, we assume that the minimum inter-distance of indice $i_1, i_2, \ldots, i_k \geq 4$. Let $\widetilde{Y} = X'Y$. For any i_k, $1 \leq k \leq s/2$, if we restrict $\widetilde{Y}$ to $\{i_k - 1, i_k, i_k + 1, i_k + 2\}$ and call the resulting vector y, then

$$y = A\alpha,$$

where A is the 4 matrix satisfying $A(i, j) = 1\{i = j\} + a \cdot 1\{|i - j| = 1\}$, $1 \leq i, j \leq 4$, and α is the 4×1 vector such that $\alpha_1 = \alpha_4 = 0$, $\alpha_2 = \tau$, and $\alpha_3 = d\tau$. In this simple model, the performance of the lasso and marginal regression can be similarly analyzed as in Section 2.3.

Now, for each of the combination of (d, ϑ), we use the method of exhausting search to determine the smallest r such that the lasso or marginal regression yields exact recovery with 50 repetition of simulations, respectively (similarly, the tuning parameters of each method are set ideally). For each method, we call the resultant value of r the *critical value for exact recovery*. For each ϑ and choices of (a, d), we find the critical values for both methods. The results are summarized in Figure 5. For a given triplet (ϑ, a, d), the method that gives a larger critical value is inferior to the one with a smaller critical value (as the region of parameters where it yields exact recovery is smaller). Figure 5 suggests that the parameters (a, d) play an important role in determining the performance of the lasso and marginal regression. For example, the performance of both procedures worsen when a get larger. This is because that as a increases, the Gram matrix moves away from that the identity matrix, and the problem of variable selection becomes increasingly harder. Also, the sign of $a \cdot d$ plays an interesting role. For example, when $a \cdot d < 0$, it is known that the marginal regression faces a so-called challenge of signal cancellation (see for example Wasserman and Roeder, 2009). It seems that the lasso handles signal cancellation better than does marginal regression. However, when (a, d) range, there is no clear winner between two methods.

Experiment 3. So far, we have focused on settings where the regression problem can be decomposed into many parallel small-size regression problems. While how to decompose remains unknown, such insight is valuable, as we can always compare the performance of two methods over each of these small-size regression problems using the theory developed in Section 2.3; the overall performance of each method is then the sum of that on these small-size problems.

With that being said, in this experiment, we investigate an example where such a "decomposition" does not exist or at least is non-obvious. Consider an experiment where $Y = X\beta + z$, and

$z \sim N(0, I_n)$. We take $p = n$ and $X = \Omega^{1/2}$, where Ω is a correlation matrix having the form

$$\Omega = \Lambda + a\xi\xi'.$$

which is a rank one perturbation of the diagonal matrix Λ. Here, ξ is the $p \times 1$ vector where its $p/2$ even coordinates are 1, and the remaining coordinates are b, where $a > 0$ and b are parameters calibrating the norm and direction of the rank one perturbation, respectively. Experiment 3 contains two sub-experiments, 3a and 3b.

In Experiment 3a, we investigate how the choices of parameters (a,b) and the signal strength affect the performance of the lasso and marginal regression. Let $p = n = 3000$, and let $\beta_i = \tau$ when $i \in \{k : k = 8 \times (\ell - 1) + 1, 1 \le \ell \le 150\}$ and $\beta_i = 0$ otherwise, where τ calibrates the signal strength. For each of the four choices $(a,b) = (0.01, 0.3), (0.01, 0.5), (0, 0.5), (0.5, -0.1)$, we compare the lasso and marginal regression for $\tau = 2, 3, \ldots, 8$. The Hamming errors are shown in Figure 6. The results suggest that the parameters (a,b) play a key role in the performance of both the lasso and marginal regression. For example, when a increases, the performance of both methods worsen, due to that the Gram matrix moves away from the identity matrix. Also, for relatively small a, it seems that marginal regression outperforms the lasso (see Panel 1 and 2 of Figure 6).

In Experiment 3b, we take a different angle and investigate how the levels of the signal sparsity affect the performance of the lasso and marginal regression. Consider a special case where where $b = 1$. In this case, ξ reduces to the vector of ones, and the Gram matrix is an equi-correlation matrix. This setting can be found in many literature on variable selection. Take $n = p = 500$. We generate the coordinates of β from the mixing distribution of point mass at 0 and the point mass at τ:

$$\beta_i \overset{iid}{\sim} (1 - \varepsilon)\nu_0 + \varepsilon\nu_\tau,$$

where ε calibrates the sparsity level and τ calibrates the signal strength (in this experiment, we take $\tau = 5$). In Figure 7, we plot the Hamming errors of 10 repetition versus the number of variables retained (which can be thought of different choices of tuning parameters). Interestingly, it seems that the performance of two methods are strikingly similar, with relatively small differences (one way or the other) when the parameters (a, ε) are moderate (neither too close to 0 nor to 1). This is interesting as when ρ is moderate, the design matrix X is significantly non-orthogonal. Additionally, the results suggest that the sparsity parameter ε has a major influence over the relative performance of two methods. When ε get larger (so the signals get denser), marginal regression tends to outperform the lasso. The underlying reason is that when both the correlation and signals are positive, the strength of individual signals are amplified due to correlation, and so have a positive effect on marginal regression.

At the same time, it seems that the correlation parameter a also have a major effect over the performance of two methods, and the error rate of both methods increase as a increases. However, somewhat surprisingly, the parameter a does not seem to have a major effect on the relative performance of two methods.

We conclude this section by mentioning that from time to time, one would like to know for the data at hand, which method is preferable. Generally, this is a hard problem, and generally, there is no clear winner between the lasso and marginal regression. However, there are something can be learned from these simulation examples.

First, the study in this section suggest an interesting perspective, which can be explained as follows. Suppose that the Gram matrix is sparse in the sense that each row has relatively few large

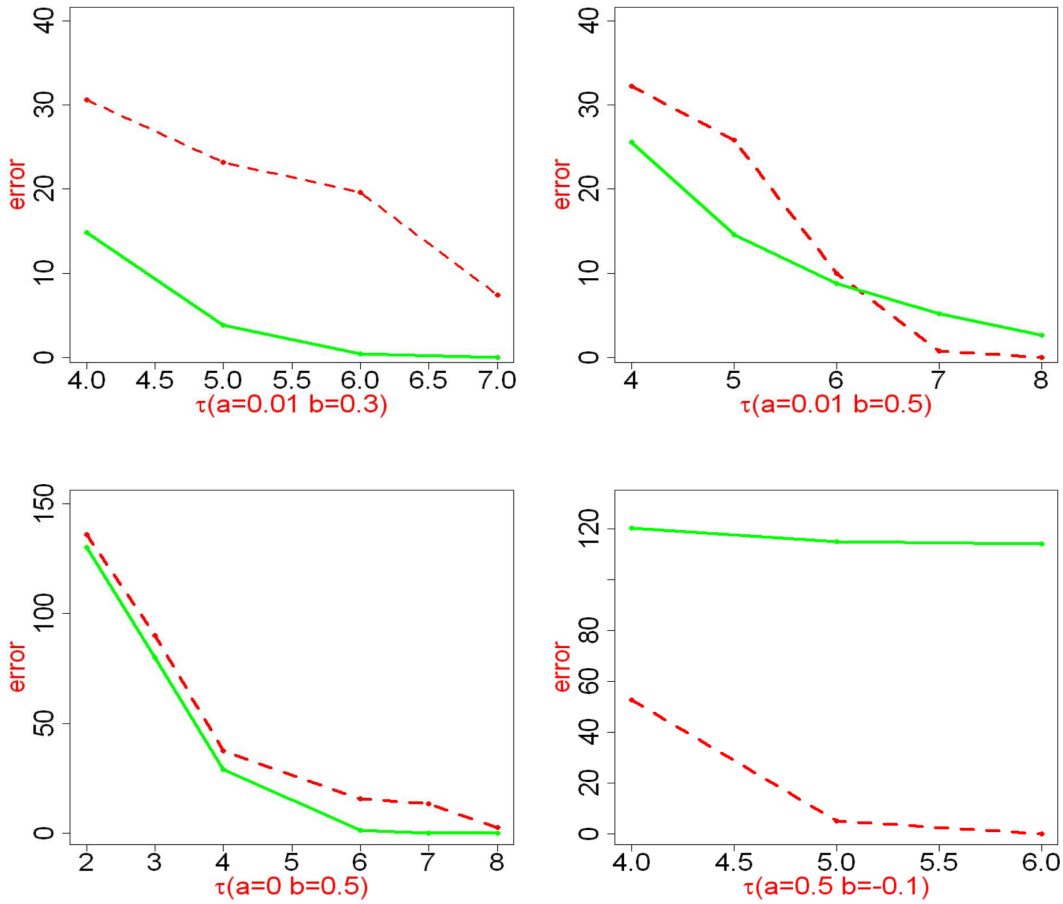


Figure 6: Comparison of Hamming errors by the lasso (dashed) and marginal regression (solid). The setting is described in Experiment 3a.

coordinates, and that the signal is also sparse. It turns out two types of sparsity interact with each other, and the large-scale regression problem reduces to many small-size regression models, each of which is obtained by restricting the rows of $X'Y$ to a small set of indices. In general, each of such of small-size regression models can be discussed in a similar fashion as those in Section 2.3. The results of these small-size regression problems then decide which of these two methods outperform the other. Take Experiment 1 for example. The performance of each method is determined by that of applying the method block-wise to the regression problem. This echos our previous argument in Section 2.3, where the relative performance of two methods for small-size problems are discussed in detail. Second, it seems that the lasso is comparably more vulnerable to extreme correlation, as discussed in Section 2.3 as well as in Example 1b. Last, it seems that in at least some examples, marginal regression is more vulnerable to the so-called "signal cancellation", which is illustrated in Proposition 3 as well as Example 2 in this section.

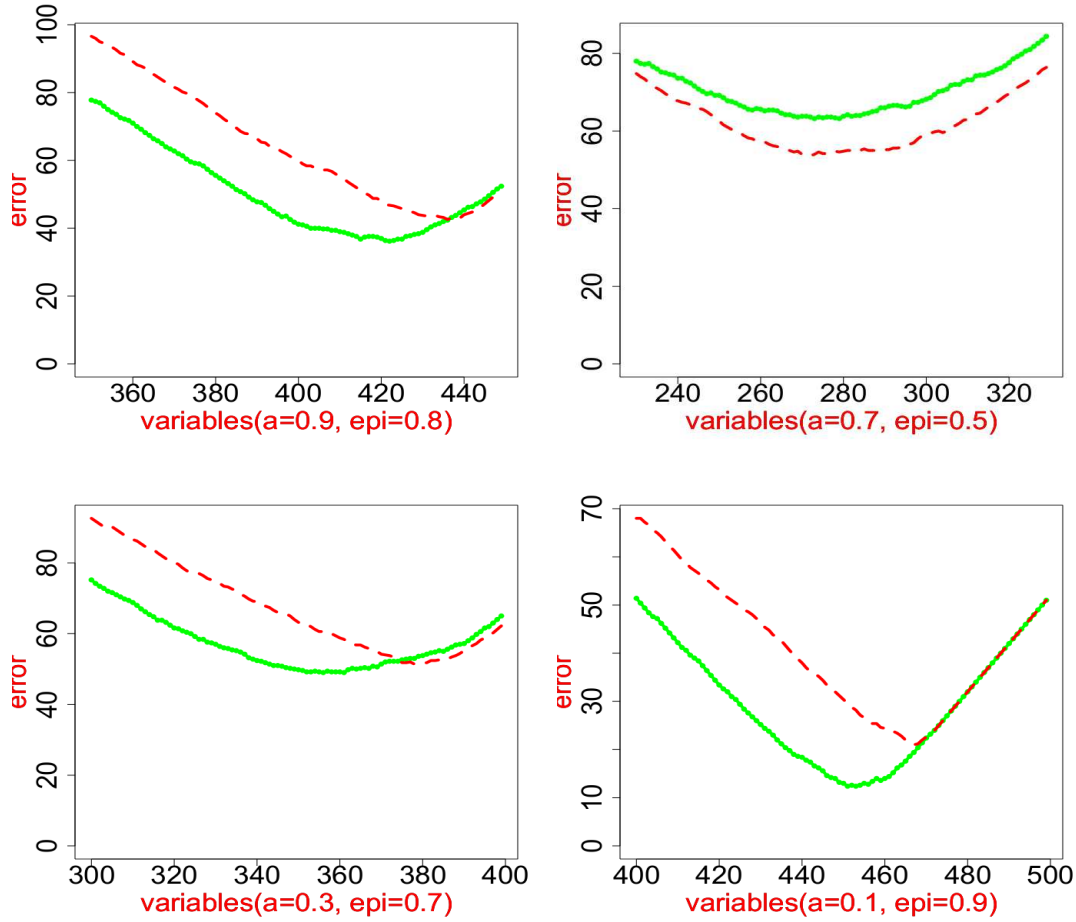


Figure 7: Comparison of Hamming errors by the lasso (dashed) and marginal regression (solid). The x-axis shows the number of retained variables. The setting is described in Experiment 3b.

6. Proofs

This section contains the technical proofs of all theorems and lemmas of the preceding sections.

6.1 Proof of Theorem 3

First, let k_i denote the number of non-zero diagonal entries in row i of D. Because D is symmetric but not diagonal, at least two rows must have non-zero k_i. Assume without loss of generality that the rows and columns of D are arranged so that the rows with non-zero k_i form the initial minor. It follows that the initial minor is itself a positive definite symmetric matrix. And because any such matrix A satisfies $|A_{ij}| < \max_k D_{kk}$ for $j \neq i$, there exists a row i of D with $k_i > 0$ and $|D_{ij}| < D_{ii}$ for any $j \neq i$.

Define β as follows:

$$\beta_j = \begin{cases} \frac{\rho D_{ii}}{D_{ij}} & \text{if } j \neq i \text{ and } D_{ij} \neq 0 \\ \rho & \text{if } j \neq i \text{ and } D_{ij} = 0 \\ -k_i \rho & \text{if } j = i.. \end{cases}$$

Because $|D_{ij}| \leq D_{ii}$, this satisfies $|\beta_j| \geq \rho$, so $\beta \in \mathcal{M}_\rho^s$. Moreover,

$$(D\beta)_i = \sum_j D_{ij}\beta_j = -k_i D_{ii}\rho + \sum_{\substack{j \neq i \\ D_{ij} \neq 0}} \frac{\rho D_{ii}}{D_{ij}} D_{ij} = 0.$$

This proves the theorem.

6.2 Proof of Lemma 4

By the definition of $\widehat{S}_n(s)$, it is sufficient to show that except for a probability that tends to 0,

$$\max |X_N^T Y| < \min |X_S^T Y|.$$

Since $Y = X\beta + z = X_S\beta_S + z$, we have $X_N^T Y = X_N^T(X_S\beta_S + z) = C_{NS}\beta_S + X_N^T z$. Note that $x_i^T z \sim N(0, \sigma_n^2)$. By Boolean algebra and elementary statistics,

$$P(\max |X_N^T z| > \sigma_n \sqrt{2\log p}) \leq \sum_{i \in N} P(|(x_i, z)| \geq \sigma_n \sqrt{2\log p}) \leq \frac{C}{\sqrt{\log p}} \frac{p-s}{p}.$$

It follows that except for a probability of $o(1)$,

$$\max |X_N^T Y| \leq \max |C_{NS}\beta_S| + \max |X_N^T z| \leq \max |C_{NS}\beta_S| + \sigma_n \sqrt{2\log p}.$$

Similarly, except for a probability of $o(1)$,

$$\min |X_S^T Y| \geq \min |C_{SS}\beta_S| - \max |X_S^T z| \geq \min |C_{SS}\beta_S| - \sigma_n \sqrt{2\log p}.$$

Combining these gives the claim. □

6.3 Proof of Theorem 5

Once the first claim is proved, the second claim follows from Lemma 4. So we only show the first claim. Write for short $\widehat{S}_n(s) = \widehat{S}_n(s^{(n)}; X^{(n)}, Y^{(n)}, p^{(n)})$, $s = s^{(n)}$, and $S = S(\beta^{(n)})$. All we need to show is

$$\lim_{n \to \infty} P(\widehat{s}_n \neq s) = 0.$$

Introduce the event

$$D_n = \{\widehat{S}_n(s) = S\}.$$

It follows from Lemma 4 that

$$P(D_n^c) \to 0.$$

Write

$$P(\widehat{s}_n \neq s) \leq P(D_n) P(\widehat{s}_n \neq s | D_n) + P(D_n^c).$$

It is sufficient to show $\lim_{n\to\infty} P(\widehat{s}_n \neq s|D_n) = 0$, or equivalently,

$$\lim_{n\to\infty} P(\widehat{s}_n > s|D_n) = 0 \qquad \text{and} \qquad \lim_{n\to\infty} P(\widehat{s}_n < s|D_n) = 0. \tag{22}$$

Consider the first claim of (22). Write for short $t_n = \sigma_n\sqrt{2\log n}$. Note that the event $\{\widehat{s}_n > s|D_n\}$ is contained in the event of $\cup_{k=s}^{p-1}\{\widehat{\delta}_n(k) \geq t_n|D_n\}$. Recalling $P(D_n^c) = o(1)$,

$$P(\widehat{s}_n > s) \leq \sum_{k=s}^{p-1}(\widehat{\delta}(k) \geq t_n|D_n) \lesssim \sum_{k=s}^{p-1} P(\widehat{\delta}_n(k) \geq t_n), \tag{23}$$

where we say two positive sequences $a_n \lesssim b_n$ if $\overline{\lim}_{n\to\infty}(a_n/b_n) \leq 1$.

Fix $s \leq k \leq p-1$. By definitions, $\widehat{H}(k+1) - \widehat{H}(k)$ is the projection matrix from R^n to $\widehat{V}_n(k+1) \cap \widehat{V}_n(k)^{\perp}$. So conditional on the event $\{\widehat{V}_n(k+1) = \widehat{V}_n(k)\}$, $\delta_n(k) = 0$, and conditional on the event $\{\widehat{V}_n(k+1) \subsetneq \widehat{V}_n(k)\}$, $\delta_n^2(k) \sim \sigma_n^2\chi^2(1)$. Note that $P(\chi^2(1) \geq 2\log n) = o(1/n)$. It follows that

$$\begin{aligned}\sum_{k=s}^{p-1} P(\widehat{\delta}_n(k) \geq t_n) &= \sum_{k=s}^{p-1} P(\widehat{\delta}_n(k) \geq t_n|\widehat{V}_n(k) \subsetneq \widehat{V}_n(k+1))P(\widehat{V}_n(k) \subsetneq \widehat{V}_n(k+1)) \\ &= o(\frac{1}{n})\sum_{k=s}^{p-1} P(\widehat{V}_n(k) \subsetneq \widehat{V}_n(k+1)).\end{aligned} \tag{24}$$

Moreover,

$$\begin{aligned}\sum_{k=s}^{p-1} P(\widehat{V}_n(k) \subsetneq \widehat{V}_n(k+1)) &= \sum_{k=s}^{p-1} E[1(\dim(\widehat{V}_n(k+1)) > \dim(\widehat{V}_n(k)))] \\ &= E[\sum_{k=s}^{p-1} 1(\dim(\widehat{V}_n(k+1)) > \dim(\widehat{V}_n(k)))].\end{aligned}$$

Note that for any realization of the sequences $\widehat{V}_n(1), \ldots, \widehat{V}_n(p)$, $\sum_{k=s}^{p-1} 1(\dim(\widehat{V}_n(k+1)) > \dim(\widehat{V}_n(k))) \leq n$. It follows that

$$\sum_{k=s}^{p-1} P(\widehat{V}_n(k) \subsetneq \widehat{V}_n(k+1)) \leq n. \tag{25}$$

Combining (23)-(25) gives the claim.

Consider the second claim of (22). By the definition of $\widehat{s}_n$, the event $\{\widehat{s}_n < s|D_n)\}$ is contained in the event $\{\widehat{\delta}_n(s-1) < t_n|D_n\}$. By definitions, $\widehat{\delta}_n(s-1) = \|(\widehat{H}(s) - \widehat{H}(s-1))Y\|$, where $\|\cdot\| = \|\cdot\|_2$ denotes the ℓ^2 norm. So all we need to show is

$$\lim_{n\to\infty} P(\|(\widehat{H}(s) - \widehat{H}(s-1))Y\| < t_n|D_n) = 0. \tag{26}$$

Fix $1 \leq k \leq p$. Recall that i_k denotes the index at which the rank of $|(Y, x_{i_k})|$ among all $|(Y, x_j)|$ is k. Denote $\widetilde{X}(k)$ by the n by k matrix $[x_{i_1}, x_{i_2}, \ldots, x_{i_k}]$, and denote $\widetilde{\beta}(k)$ by the k-vector $(\beta_{i_1}, \beta_{i_2}, \ldots, \beta_{i_k})^T$. Conditional on the event D_n, $\widehat{S}_n(s) = S$, and $\beta_{i_1}, \beta_{i_2}, \ldots, \beta_{i_s}$ are all the nonzero coordinates of β. So according to our notations,

$$X\beta = \widetilde{X}(s)\widetilde{\beta}(s) = \widetilde{X}(s-1)\widetilde{\beta}(s-1) + \beta_{i_s}x_{i_s} \tag{27}$$

Now, first, note that $\widehat{H}(s)\widetilde{X}(s) = \widetilde{X}(s)$ and $\widehat{H}(s-1)\widetilde{X}(s-1) = \widetilde{X}(s-1)$. Combine this with (27). It follows from direct calculations that

$$(\widehat{H}(s) - \widehat{H}(s-1))X\beta = (I - \widehat{H}(s-1))x_{i_s}. \tag{28}$$

Second, since $x_{i_s} \in \widehat{V}_n(s)$, $(I - \widehat{H}(s))x_{i_s} = 0$. So

$$(I - \widehat{H}_{s-1})x_{i_s} = (I - \widehat{H}(s))x_{i_s} + (\widehat{H}(s) - \widehat{H}(s-1))x_{i_s} = (\widehat{H}_s - \widehat{H}_{s-1})x_{i_s}. \tag{29}$$

Last, split x_{i_s} into two terms, $x_{i_s} = x_{i_s}^{(1)} + x_{i_s}^{(2)}$ such that $x_{i_s}^{(1)} \in \widehat{V}_n(s-1)$ and $x_{i_s}^{(2)} \in \widehat{V}_n(s) \cap (\widehat{V}_n(s-1))^{\perp}$. It follows that $(\widehat{H}(s) - \widehat{H}(s-1))x_{i_s}^{(1)} = 0$, and so

$$(\widehat{H}(s) - \widehat{H}(s-1))x_{i_s} = (\widehat{H}(s) - \widehat{H}(s-1))x_{i_s}^{(2)}. \tag{30}$$

Combining (28)-(30) gives

$$(\widehat{H}(s) - \widehat{H}(s-1))X\beta = (\widehat{H}(s) - \widehat{H}(s-1))x_{i_s}^{(2)}.$$

Recall that $Y = X\beta + z$, it follows that

$$(\widehat{H}_s - \widehat{H}_{s-1})Y = (\widehat{H}(s) - \widehat{H}(s-1))(\beta_{i_s}x_{i_s}^{(2)} + z). \tag{31}$$

Now, take an orthonormal basis of R^n, say $\widehat{q}_1, \widehat{q}_2, \ldots, \widehat{q}_n$, such that $\widehat{q}_1 \in \widehat{V}_n(s) \cap \widehat{V}_n(s-1)^{\perp}$, $\widehat{q}_2, \ldots, \widehat{q}_s \in \widehat{V}_n(s-1)$, and $\widehat{q}_{s+1}, \ldots, \widehat{q}_n \in \widehat{V}_n(s)^{\perp}$. Recall that $x_{i_s}^{(2)}$ is contained in the one dimensional linear space $\widehat{V}_n(s) \cap \widehat{V}_n(s-1)^{\perp}$, so without loss of generality, assume $(x_{i_s}^{(2)}, \widehat{q}_1) = \|x_{i_s}^{(2)}\|$. Denote the square matrix $[\widehat{q}_1, \ldots, \widehat{q}_n]$ by $\widehat{Q}$. Let $\widetilde{z} = \widehat{Q}z$ and let $\widetilde{z}_1$ be the first coordinate of $\widetilde{z}$. Note that marginally $\widetilde{z}_1 \sim N(0, \sigma_n^2)$. Over the event D_n, it follows from the construction of $\widehat{Q}$ and basic algebra that

$$\|(\widehat{H}(s) - \widehat{H}(s-1))(\beta_{i_s}x_{i_s}^{(2)} + z)\|^2 = (\|\beta_{i_s}x_{i_s}^{(2)}\| + \widetilde{z}_1)^2. \tag{32}$$

Combine (31) and (32),

$$\|(\widehat{H}(s) - \widehat{H}(s-1))Y\|^2 = (\|\beta_{i_s}x_{i_s}^{(2)}\| + \widetilde{z}_1)^2, \qquad \text{over the event } D_n.$$

As a result,

$$P(\|(\widehat{H}(s) - \widehat{H}(s-1))Y\| < t_n | D_n) = P((\|\beta_{i_s}x_{i_s}^{(2)}\| + \widetilde{z}_1)^2 < t_n | D_n). \tag{33}$$

Recall that conditional on the event D_n, $\widehat{S}_n(s) = S$. So by the definition of $\Delta_n^* = \Delta_n(\beta, X, p)$,

$$\|\beta_{i_s}x_{i_s}^{(2)}\| \geq \Delta_n^*,$$

and

$$P((\|\beta_{i_s}x_{i_s}^{(2)}\| + \widetilde{z}_1)^2 < t_n | D_n) \leq P(\|\beta_{i_s}x_{i_s}^{(2)}\| + \widetilde{z}_1 < t_n | D_n) \leq P(\Delta_n^* + \widetilde{z}_1 < t_n | D_n). \tag{34}$$

Recalling that $\widetilde{z}_1 \sim N(0, \sigma_n^2)$ and that $P(D_n^c) = o(1)$,

$$P(\Delta_n^* + \widetilde{z}_1 < t_n | D_n) \leq P(\Delta_n^* + \widetilde{z}_1 < t_n) + o(1). \tag{35}$$

Note that by the assumption of $(\frac{\Delta_n^*}{\sigma_n} - t_n) \to \infty$, $P(\Delta_n^* + \widetilde{z}_1 < t_n) = o(1)$. Combining this with (34)-(35) gives

$$P((\|\beta_{i_s}x_{i_s}^{(2)}\| + \widetilde{z}_1)^2 < t_n^2 | D_n) = o(1). \tag{36}$$

Inserting (36) into (33) gives (26). □

6.4 Proof of Lemma 6

For $1 \le i \le p$, introduce the random variable

$$Z_i = \sum_{j \neq i}^{p} \beta_j(x_i, x_j).$$

When $B_i = 0$, $\beta_i = 0$, and so $Z_i = \sum_{j=1}^{p} \beta_j(x_i, x_j)$. By the definition of C_{NS},

$$\max |C_{NS}\beta_S| = \max_{1 \le i \le p} \{(1 - B_i) \cdot |\sum_{j=1}^{p} \beta_j(x_i, x_j)|\} = \max_{1 \le i \le p} \{(1 - B_i)|Z_i|\}.$$

Also, recalling that the columns of matrix X are normalized such that $(x_i, x_i) = 1$, the diagonal coordinates of $(C_{SS} - I)$ are 0. Therefore,

$$\max |(C_{SS} - I)\beta_S| = \max_{1 \le i \le p} \{B_i \cdot |\sum_{j \neq i} \beta_j(x_i, x_j)|\} = \max_{1 \le i \le p} \{B_i \cdot |Z_i|\}.$$

Note that Z_i and B_i are independent and that $P(B_i = 0) = (1 - \varepsilon)$. It follows that

$$P(\max |C_{NS}\beta_S| \ge \delta) \le \sum_{i=1}^{p} P(B_i = 0) P(|Z_i| \ge \delta | B_i = 0) = (1 - \varepsilon) \sum_{i=1}^{p} P(|Z_i| \ge \delta),$$

and

$$P(\max |(C_{SS} - I)\beta_S| \ge \delta) \le \sum_{i=1}^{p} P(B_i = 1) P(|Z_i| \ge \delta | B_i = 1) = \varepsilon \sum_{i=1}^{p} P(|Z_i| \ge \delta).$$

Compare these with the lemma. It is sufficient to show

$$P(|Z_i| \ge \delta) \le e^{-\delta t}[e^{\varepsilon \bar{g}_i(t)} + e^{\varepsilon \bar{g}_i(-t)}]. \tag{37}$$

Now, by the definition of $g_{ij}(t)$, the moment generating function of Z_i satisfies that

$$E[e^{tZ_i}] = E[e^{t \sum_{j \neq i} \beta_j(x_i, x_j)}] = \Pi_{j \neq i}[1 + \varepsilon g_{ij}(t)].$$

Since $1 + x \le e^x$ for all x, $1 + \varepsilon g_{ij}(t) \le e^{\varepsilon g_{ij}(t)}$, so by the definition of $\bar{g}_i(t)$,

$$E[e^{tZ_i}] \le \Pi_{j \neq i} e^{\varepsilon g_{ij}(t)} = e^{\varepsilon \bar{g}_i(t)}.$$

It follows from Chebyshev's inequality that

$$P(Z_i \ge \delta) \le e^{-\delta t} E[e^{tZ_i}] \le e^{-\delta t} e^{\varepsilon \bar{g}_i(t)}. \tag{38}$$

Similarly,

$$P(Z_i < -\delta) \le e^{-\delta t} e^{\varepsilon \bar{g}_i(-t)} \tag{39}$$

Inserting (38)-(39) into (37) gives the claim. □

6.5 Proof of Corollary 3.1

Choose a constant q such that $q/2 - c_2 q > 1$ and let $t_n = q\log(p)/a_n$. By the definition of $A_n(a_n/2, \varepsilon_n, \bar{g})$, it is sufficient to show that for all $1 \le i \le p$,

$$e^{-a_n t_n/2} e^{\varepsilon_n \bar{g}_i(t_n)} = o(1/p), \qquad e^{-a_n t_n/2} e^{\varepsilon_n \bar{g}_i(-t_n)} = o(1/p).$$

The proofs are similar, so we only show the first one. Let u be a random variable such that $u \sim \pi_n$. Recall that the support of $|u|$ is contained in $[a_n, b_n]$. By the assumptions and the choice of t_n, for all fixed i and $j \neq i$, $|t_n u(x_i, x_j)| \le q\log(p)(b_n/a_n)|(x_i, x_j)| \le c_1 q$. Since $e^x - 1 \le x + e^x x^2/2$, it follows from Taylor expansion that

$$\varepsilon_n \bar{g}_i(t_n) = \varepsilon_n [e^{t_n u(x_i, x_j)} - 1] \le \varepsilon_n \sum_{j \neq i} E_{\pi_n}[t_n u(x_i, x_j) + \frac{e^{c_1 q}}{2} t_n^2 u^2 (x_i, x_j)^2].$$

By definitions of $m_n(X)$ and $v_n^2(X)$, $\varepsilon_n \sum_{j \neq i} E_\pi[t_n u(x_i, x_j)] = t_n \mu_n^{(1)} m_n(X)$, and $\varepsilon_n \sum_{j \neq i} E_{\pi_n}[t_n^2 u^2 (x_i, x_j)^2] = t_n^2 \mu_n^{(2)} v_n^2(X)$. It follows from (12) that

$$\varepsilon_n \bar{g}_i(t_n) \le q\log(p) \cdot [\frac{\mu_n^{(1)}}{a_n} m_n(X) + \frac{e^{c_1 q}}{2} \frac{\mu_n^{(2)}}{a_n^2} v_n^2(X) q\log(p)] \lesssim q c_2 \log(p).$$

Therefore,

$$e^{-a_n t_n/2} e^{\varepsilon_n \bar{g}_i(t_n)} \le e^{-[q/2 - c_2 q + o(1)]\log(p)},$$

and claim follows by the choice of q. □

6.6 Proof of Corollary 3.2

Choose a constant q such that $2 < q < \frac{c_3}{c_4 \delta}$. Let $t_n = a_n q \log(p)$, and u be a random variable such that $u \sim \Pi_n$. Similar to the proof of Lemma 3.1, we only show that

$$e^{-a_n t_n/2} e^{\varepsilon_n \bar{g}_i(t_n)} = o(1/p), \qquad \text{for all } 1 \le i \le p.$$

Fix $i \neq j$. When $(x_i, x_j) = 0$, $e^{t u(x_i, x_j)} - 1 = 0$. When $(x_i, x_j) \neq 0$, $e^{t_n u(x_i, x_j)} - 1 \le e^{t_n (b_n/a_n)\delta} \le e^{c_4 q \delta \log p}$. Also, $\varepsilon_n N_n^* \le e^{-[c_3 + o(1)]\log(p)}$. Therefore,

$$\varepsilon_n \bar{g}_i(t) \le \varepsilon_n N_n^* e^{c_4 q \delta \log(p)} \le e^{-[c_3 - c_4 q \delta + o(1)]\log p}.$$

By the choice of q, $c_3 - c_4 q \delta > 0$, so $\varepsilon_n \bar{g}_i(t) = o(1)$. It follows that

$$e^{-a_n t_n/2} e^{\varepsilon_n \bar{g}_i(t_n)} \le o(e^{-a_n t_n/2}) = o(e^{-q\log(p)/2}),$$

which gives the claim by $q > 2$. □

6.7 Proof of Theorem 9

Write

$$X = [x_1, \widetilde{X}], \qquad \beta = (\beta_1, \widetilde{\beta})^T.$$

Fix a constant $c_0 > 3$. Introduce the event

$$D_n(c_0) = \{1_S^T \widetilde{X}_S^T \widetilde{X}_S 1_S \le |S|[1 + \sqrt{\frac{|S|}{n}}(1 + \sqrt{2c_0 \log p})]^2, \text{for all } S\}.$$

The following lemma is proved in Section 6.7.1.

Lemma 12 *Fix $c_0 > 3$. As $p \to \infty$,*

$$P(D_n^c(c_0)) = o(1/p^2).$$

Since $d_n(\widehat{\beta}|X) \le p$ for any variable selection procedure $\widehat{\beta}$, Lemma 12 implies that the overall contribution of D_n^c to the Hamming distance $d_n^*(\widehat{\beta})$ is $o(1/p)$. In addition, write

$$d_n(\widehat{\beta}|X) = \sum_{j=1}^{p} E[1(\widehat{\beta}_j \neq \beta_j)].$$

By symmetry, it is sufficient to show that for any realization of $(X,\beta) \in D_n(c_0)$,

$$E[1(\widehat{\beta}_j \neq \beta_j)] \ge \begin{cases} L(n)p^{-\frac{(\vartheta+r)^2}{4r}}, & r \ge \vartheta, \\ p^{-\vartheta}, & 0 < r < \vartheta, \end{cases} \tag{40}$$

where $L(n)$ is a multi-log term that does not depend on (X,β).

We now show (40). Toward this end, we relate the estimation problem to the problem of testing the null hypothesis of $\beta_1 = 0$ versus the alternative hypothesis of $\beta_1 \neq 0$. Denote ϕ by the density of $N(0,1)$. Recall that $X = [x_1, \widetilde{X}]$ and $\beta = (\beta_1, \widetilde{\beta})^T$. The joint density associated with the null hypothesis is

$$f_0(y) = f_0(y;\varepsilon_n,\tau_n,n|X)\phi(y-\widetilde{X}\widetilde{\beta})d\widetilde{\beta} = \phi(y)\int e^{y^T\widetilde{X}\widetilde{\beta}-|\widetilde{X}\widetilde{\beta}|^2/2}d\widetilde{\beta},$$

and the joint density associated with the alternative hypothesis is

$$\begin{aligned} f_1(y) = f_1(y;\varepsilon_n,\tau_n,n|X) &= \int \phi(y-\tau_n x_1 - \widetilde{X}\widetilde{\beta})d\widetilde{\beta} \\ &= \phi(y-\tau_n x_1)\int e^{y^T\widetilde{X}\widetilde{\beta}-|\widetilde{X}\widetilde{\beta}|^2/2}e^{-\tau_n x_1^T\widetilde{X}\widetilde{\beta}}d\widetilde{\beta}. \end{aligned}$$

Since the prior probability that the null hypothesis is true is $(1-\varepsilon_n)$, the optimal test is the Neyman-Pearson test that rejects the null if and only if

$$\frac{f_1(y)}{f_0(y)} \ge \frac{(1-\varepsilon_n)}{\varepsilon_n}.$$

The optimal testing error is equal to

$$1 - \|(1-\varepsilon_n)f_0 - \varepsilon_n f_1\|_1.$$

Compared to (2), $\|\cdot\|_1$ stands for the L^1-distance between two functions, not the ℓ^1 norm of a vector.

We need to modify f_1 into a more tractable form, but with negligible difference in L^1-distance. Toward this end, let $N_n(\widetilde{\beta})$ be the number of nonzeros coordinates of $\widetilde{\beta}$. Introduce the event

$$B_n = \{|N_n(\widetilde{\beta}) - p\varepsilon_n| \le \frac{1}{2}p\varepsilon_n\}.$$

Let

$$a_n(y) = a_n(y;\varepsilon_n,\tau_n|X) = \frac{\int(e^{y^T\widetilde{X}\widetilde{\beta}-|\widetilde{X}\widetilde{\beta}|^2/2})(e^{-\tau_n x_1^T\widetilde{X}\widetilde{\beta}})\cdot 1_{\{B\}}d\widetilde{\beta}}{\int(e^{-y^T\widetilde{X}\widetilde{\beta}-|\widetilde{X}\widetilde{\beta}|^2/2})\cdot 1_{\{B\}}d\widetilde{\beta}}. \tag{41}$$

Note that the only difference between the numerator and the denominator is the term $e^{-\tau_n x_1^T \widetilde{X}\widetilde{\beta}}$ which ≈ 1 with high probability. Introduce

$$\widetilde{f}_1(y) = a_n(y)\phi(y-\tau_n x_1)\int e^{y^T\widetilde{X}\widetilde{\beta}-|\widetilde{X}\widetilde{\beta}|^2/2}d\widetilde{\beta}.$$

The following lemma is proved in Section 6.7.2.

Lemma 13 *As $p\to\infty$, there is a generic constant $c>0$ that does not depend on y such that $|a_n(y)-1|\le c\log(p)p^{(1-\vartheta)-\theta/2}$ and $\|f_1-\widetilde{f}_1\|_1 = o(1/p)$.*

We now ready to show the claim. Define $\Omega_n = \{y : a_n(y)\phi(y-\tau_n x_1)\ge\phi(y)\}$. Note that by the definitions of $f_0(y)$ and $\widetilde{f}_1(y)$, $y\in\Omega_n$ if and only if

$$\frac{\varepsilon_n\widetilde{f}_1(y)}{(1-\varepsilon_n)f_0(y)}\ge 1.$$

By Lemma 13,

$$|\int\widetilde{f}_1(y)dy-1|\le\|\widetilde{f}_1-f_1\|_1\le o(1/p).$$

It follows from elementary calculus that

$$1-\|(1-\varepsilon_n)f_0-\varepsilon_n\widetilde{f}_1\|_1 = \int_{\Omega_n}(1-\varepsilon_n)f_0(y)dy+\int_{\Omega_n^c}\varepsilon_n\widetilde{f}_1(y)dy+o(1/p).$$

Using Lemma 13 again, we can replace $\widetilde{f}_1$ by f_1 on the right hand side, so

$$1-\|(1-\varepsilon_n)f_0-\varepsilon_n\widetilde{f}_1\|_1 = \int_{\Omega_n}(1-\varepsilon_n)f_0(y)dy+\int_{\Omega_n^c}\varepsilon_n f_1(y)dy+o(1/p).$$

At the same time, let $\delta_p = c\log(p)p^{(1-\vartheta)-\theta/2}$ be as in Lemma 13, and let

$$t_0 = t_0(\vartheta,r) = \frac{\vartheta+r}{2\sqrt{r}}\sqrt{2\log p}.$$

be the unique solution of the equation $\phi(t) = \varepsilon_n\phi(t-\tau_n)$. It follows from Lemma 13 that,

$$\{\tau_n x^T y\ge t_0(1+\delta_p)\}\subset\Omega_n\subset\{\tau_n x_1^T y\ge t_0(1-\delta_p)\}.$$

As a result,

$$\int_{\Omega_n}f_0(y)dy\ge\int_{\tau_n x_1^T y\ge t_0(1+\delta_p)}f_0(y)\equiv P_0(\tau_n x_1^T Y\ge t_0(1+\delta_p)),$$

and

$$\int_{\Omega_n^c}f_1(y)dy\ge\int_{\tau_n x_1^T y\le t_0(1-\delta_p)}f_1(y)\equiv P_1(\tau_n x_1^T Y\le t_0(1-\delta_p)).$$

Note that under the null, $x_1^T Y = x_1^T\widetilde{X}\widetilde{\beta}+x_1^T z$. It is seen that given x_1, $x_1^T z\sim N(0,|x_1|^2)$, and $|x_1|^2 = 1+O(1/\sqrt{n})$. Also, it is seen that except for a probability of $o(1/p)$, $x_1^T\widetilde{X}\widetilde{\beta}$ is algebraically small. It follows that

$$P_0(\tau_n x_1^T Y\ge t_0(1+\delta_p))\lesssim\bar{\Phi}(t_0) = L(n)p^{-\frac{(\vartheta+r)^2}{4r}},$$

where $\bar{\Phi} = 1 - \Phi$ is the survival function of $N(0,1)$. Similarly, under the alternative,

$$x_1^T y = \tau_n(x_1, x_1) + x_1^T \widetilde{X}\widetilde{\beta} + x_1^T z,$$

where $(x_1, x_1) = 1 + O(1/\sqrt{n})$. So

$$\varepsilon_n P_1(\tau_n x_1^T y \le t_0(1-\delta_p)) \lesssim \Phi(t_0 - \tau_n) = \begin{cases} L(n)p^{-\frac{(\vartheta+r)^2}{4r}}, & r \ge \vartheta, \\ L(n)p^{-\vartheta}, & 0 < r < \vartheta, \end{cases}$$

Combine these gives the theorem. □

6.7.1 PROOF OF LEMMA 12

It is seen that

$$P(D_n^c(c_0)) \le \sum_{k=1}^{p} P\bigg(1_S^T X^T X 1_S \ge k[1 + \sqrt{\frac{k}{n}}(1 + \sqrt{2c_0 \log p})]^2, \text{for all } S \text{ with } |S| = k\bigg).$$

Fix $k \ge 1$. There are $\binom{p}{k}$ different S with $|S| = k$. It follows from Vershynin (2010, Lecture 9) that except a probability of $2\exp(-c_0 \log(p) \cdot k)$ that the largest eigenvalue of $X_S^T X_S$ is no greater than $[1 + \sqrt{\frac{k}{n}}(1 + \sqrt{2c_0 \log p})]^2$. So for any S with $|S| = k$, it follows from basic algebra that

$$P(1_S^T X^T X 1_S \ge k[1 + \sqrt{\frac{k}{n}}(1 + \sqrt{2c_0 \log p})]^2) \le 2\exp(-c_0 \log(p) \cdot k).$$

Combining these with $\binom{p}{k} \le p^k$ gives

$$P(D_n^c(c_0)) \le 2\sum_{k=1}^{p} \binom{p}{k} \exp(-c_0(\log p)k) \le 2\sum_{k=1}^{p} \exp(-(c_0 - 1)\log(p)k).$$

The claim follows by $c_0 > 3$. □

6.7.2 PROOF OF LEMMA 13

First, we claim that for any X in event $D_n(c_0)$,

$$|x_1^T \widetilde{X}\widetilde{\beta}| \le c \log(p)(N(\widetilde{\beta})/\sqrt{n}), \tag{42}$$

where $c > 0$ is a generic constant. Suppose $N_n(\widetilde{\beta}) = k$ and the nonzero coordinates of $\widetilde{\beta}$ are $i_1, i_2, \ldots, i_k$. Denote the $(k+1) \times (k+1)$ submatrix of $X^T X$ containing the 1^{st}, $(1+i_1)$-th, ..., and $(1+i_k)$-th rows and columns by U_{k+1}. Let ξ_1 be the $(k+1)$-vector with 1 on the first coordinate and 0 elsewhere, let ξ_2 be the $(k+1)$-vector with 0 on the first coordinate and 1 elsewhere. Then

$$x_1^T \widetilde{X}\widetilde{\beta} = \tau_n \xi_1^T U_{k+1} \xi_2 \equiv \tau_n \xi_1^T (U_{k+1} - I_{k+1})\xi_2.$$

Let $(U_{k+1} - I_{k+1}) = Q_{k+1}\Lambda_{k+1}Q_{k+1}^T$ be the orthogonal decomposition. By the definition of $D_n(c_0)$, all eigenvalues of $(U_{k+1} - I_{k+1})$ are no greater than $(1 + \sqrt{c\log(p)k/n})^2 - 1 \le \sqrt{c \log p}\sqrt{k/n}$ in absolute value. As a result, all diagonal coordinates of Λ_{k+1} are no greater than

$$\sqrt{c \log p}\sqrt{k/n}$$

in absolute value, and

$$\|\xi_1^T(U_{k+1}-I_{k+1})\xi_2\| \le \|\xi_1^T Q_{k+1}\Lambda_{k+1}\| \cdot \|Q_{k+1}\xi_2\| \le \sqrt{c\log p}\sqrt{k/n}\|\xi_1^T Q_{k+1}\| \cdot \|Q_{k+1}\xi_2\|.$$

The claim follows from $\|\xi_1^T Q_{k+1}\| = 1$ and $\|Q_{k+1}\xi_2\| = \sqrt{k}$.

We now show the lemma. Consider the first claim. Consider a realization of X in the event $D_n(c_0)$ and a realization of $\widetilde{\beta}$ in the event B_n. By the definitions of B_n, $N_n(\widetilde{\beta}) \le p\varepsilon_n + \frac{1}{2}p\varepsilon_n$. Recall that $p\varepsilon_n = p^{1-\vartheta}$, $n = p^\theta$. It follows that $\log(p)N(\widetilde{\beta})/\sqrt{n} \le c\log(p)p\varepsilon_n/\sqrt{n} = c\log(p)p^{1-\vartheta-\theta/2}$. Note that by the assumption of $(1-\vartheta) < \theta/2$, the exponent is negative. Combine this with (42),

$$|e^{-\tau_n x_1^T \widetilde{X}\widetilde{\beta}} - 1| \le c\log(p)(N(\widetilde{\beta})/\sqrt{n}), \tag{43}$$

Now, note that in the definition of $a_n(y)$ (i.e., (41)), the only difference between the integrand on the top and that on the bottom is the term $e^{-\tau_n x_1^T \widetilde{X}\widetilde{\beta}}$. Combine this with (43) gives the claim.

Consider the second claim. By the definitions of $\widetilde{f}_1(y)$ and $a_n(y)$,

$$\begin{aligned}\widetilde{f}_1(y) &= a_n(y)\phi(y-\tau_n x_1)\cdot\left[\int [e^{y^T\widetilde{X}\widetilde{\beta}-|\widetilde{X}\widetilde{\beta}|^2/2}1_{B_n}]d\widetilde{\beta} + \int [e^{y^T\widetilde{X}\widetilde{\beta}-|\widetilde{X}\widetilde{\beta}|^2/2}1_{B_n^c}]d\widetilde{\beta}\right]\\ &= \phi(y-\tau_n x_1)\cdot\left[\int [e^{y^T\widetilde{X}\widetilde{\beta}-|\widetilde{X}\widetilde{\beta}|^2/2}e^{-\tau_n x_1^T\widetilde{X}\widetilde{\beta}}1_{B_n^c}]d\widetilde{\beta} + a_n(y)\int [e^{y^T\widetilde{X}\widetilde{\beta}-|\widetilde{X}\widetilde{\beta}|^2/2}1_{B_n^c}]d\widetilde{\beta}\right].\end{aligned}$$

By the definition of $f_1(y)$,

$$f_1(y) = \phi(y-\tau_n x_1)\cdot\left[\int [e^{y^T\widetilde{X}\widetilde{\beta}-|\widetilde{X}\widetilde{\beta}|^2/2}e^{-\tau_n x_1^T\widetilde{X}\widetilde{\beta}}1_{B_n}]d\widetilde{\beta} + \int [e^{y^T\widetilde{X}\widetilde{\beta}-|\widetilde{X}\widetilde{\beta}|^2/2}e^{-\tau_n x_1^T\widetilde{X}\widetilde{\beta}}1_{B_n^c}]d\widetilde{\beta}\right].$$

Compare two equalities and recall that $a_n(y) \sim 1$ (Lemma 12),

$$\begin{aligned}\|f_1-\widetilde{f}_1\|_1 &\lesssim \int \phi(y-\tau_n x_1)[\int (e^{y^T\widetilde{X}\widetilde{\beta}-|\widetilde{X}\widetilde{\beta}|^2/2} + e^{y^T\widetilde{X}\widetilde{\beta}-|\widetilde{X}\widetilde{\beta}|^2/2}e^{-\tau_n x_1^T\widetilde{X}\widetilde{\beta}})1_{B_n^c}d\widetilde{\beta}]dy\\ &= \int\int \phi(y-\tau_n x_1-\widetilde{X}\widetilde{\beta})[e^{\tau_n x_1^T\widetilde{X}\widetilde{\beta}}+1]1_{B_n^c}d\widetilde{\beta}dy.\end{aligned} \tag{44}$$

Integrating over y, the last term is equal to $\int [1+e^{\tau_n x_1^T\widetilde{X}\widetilde{\beta}}]\cdot 1_{B_n^c}d\widetilde{\beta}$.

At the same time, by (42) and the definition of B_n^c,

$$\int [1+e^{\tau_n x_1^T\widetilde{X}\widetilde{\beta}}]\cdot 1_{B_n^c}d\widetilde{\beta} \le \sum_{\{k:|k-p\varepsilon_n|\ge\frac{1}{2}p\varepsilon_n\}} [1+e^{c\log(p)k/\sqrt{n}}]P(N(\widetilde{\beta})=k). \tag{45}$$

Recall that $p\varepsilon_n = p^{1-\vartheta}$, $n = p^\theta$, and $(1-\vartheta) < \theta/2$. Using Bennett's inequality for $P(N(\widetilde{\beta})=k)$ (e.g., Shorack and Wellner, 1986, Page 440), it follows from elementary calculus that

$$\sum_{\{k:|k-p\varepsilon_n|\ge\frac{1}{2}p\varepsilon_n\}} [1+e^{c\log(p)k/\sqrt{n}}]P(N(\widetilde{\beta})=k) = o(1/p). \tag{46}$$

Combining (44)–(46) gives the claim. □

Acknowledgments

We would like to thank David Donoho, Robert Tibshirani, and anonymous referees for helpful discussions. CG was supported in part by NSF grant DMS-0806009 and NIH grant R01NS047493, JJ was supported in part by NSF CAREER award DMS-0908613, LW was supported in part by NSF grant DMS-0806009, and ZY was supported in part by NSF grant SES-1061387 and NIH/NIDA grant R90 DA023420.

References

P. Bühlmann, M. Kalisch, and M. H. Maathuis. Variable selection in high-dimensional linear models: partially faithful distributions and the PC-simple algorith. *Biometrika*, 97:261–278, 2009.

T. Cai, L. Wang, and G. Xu. Shifting inequality and recovery of sparse signals. *IEEE Transactions on Signal Processing*, 59(3):1300–1308, 2010.

E. J. Candès and Y. Plan. Near-ideal model selection by ℓ^1 minimization. *The Annals of Statistics*, 37:2145–2177, 2009.

E. J. Candés and T. Tao. The Dantzig selector: statistical estimation when p is much larger than n. *The Annals of Statistics*, 35:2313–2351, 2007.

S. Chen, D. Donoho, and M. Saunders. Atomic decomposition by basis pursuit. *SIAM Journal on Scientific Computing*, 20(1):33–61, 1998.

D. Donoho. For most large underdetermined systems of equations, the minimal ℓ^1-norm near-solution approximates the sparsest near-solution. *Communications on Pure and Applied Mathematics*, 59(7):907–934, 2006.

D. Donoho and M. Elad. Optimally sparse representation in general (nonorthogonal) dictionaries via ℓ^1 minimization. *Proceedings of the National Academy of Sciences of the United States of America*, 100(5):2197–2202, 2003.

D. Donoho and X. Huo. Uncertainty principles and ideal atomic decomposition. *IEEE Transactions on Information Theory*, 47(7):2845–2862, 2001.

D. Donoho and J. Jin. Higher criticism for detecting sparse heterogeneous mixtures. *The Annals of Statistics*, 32(3):962–994, 2004.

B. Efron, R. Tibshirani, J. Storey, and V. Tusher. Empirical Bayes analysis of a microarray experiment. *Journal of the American Statistical Association*, 96:1151–1160, 2001.

B. Efron, T. Hastie, I. Johnstone, and R. Tibshirani. Least angle regression. *The Annals of Statistics*, 32(2):407–499, 2004.

J. Fan and J. Lv. Sure independence screening for ultrahigh dimensional feature space. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 70(5):849–911, 2008.

J.J. Fuchs. Recovery of exact sparse representations in the presence of noise. *IEEE Transactions on Information Theory*, 51(10):3601–3608, 2005.

P. Ji and J. Jin. UPS delivers optimal phase diagram in high dimensional variable selection. *The Annals of Statistics*, 40(1):73–103, 2012.

J. Jin. Proportion of nonzero normal means: oracle equivalence and uniformly consistent estimators. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 70(3):461–493, 2007.

K. Knight and W. J. Fu. Asymptotics for lasso-type estimators. *The Annals of Statistics*, 28:1356–1378, 2000.

N. Meinshausen and P. Bühlmann. High-dimensional graphs and variable selection with the lasso. *The Annals of Statistics*, 34(3):1436–1462, 2006.

N. Meinshausen and J. Rice. Estimating the proportion of false null hypotheses among a large number of independently tested hypotheses. *The Annals of Statistics*, 34(1):373–393, 2006.

P. Ravikumar. *Personal Communication*, 2007.

J. M. Robins, R. Scheines, P. Spirtes, and L. Wasserman. Uniform consistency in causal inference. *Biometrika*, 90(3):491–515, 2003.

G. R. Shorack and J. A. Wellner. *Empirical Processes with Applications to Statistics*. John Wiley & Sons, NY, 1986.

P. Spirtes, C. Glymour, and R. Scheines. *Causation, Prediction, and Search (Lecture Notes in Statistics)*. Springer-Verlag, NY, 1993.

T. Sun and C.-H. Zhang. Scaled sparse linear regression. 2011. Manuscript available at http://arxiv.org/abs/1104.4595.

R. Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 58(1):267–288, 1996.

J. Tropp. Greed is good: algorithic results for sparse approximation. *IEEE Transactions on Information Theory*, 50(10):2231–2242, 2004.

R. Vershynin. *Introduction to the Non-asymptotic Analysis of Random Matrices*. Lecture notes, Department of Mathematics, University of Michigan, 2010. Available electronically via www-personal.umich.edu/˜romanv/teaching/2006-07/280/course.html.

M. Wainwright. *Sharp Threshold for High-dimensional and Noisy Recovery of Sparsity*. Technical report, Department of Statistics, University of Berkeley, 2006.

L. Wasserman. *All of Nonparametric Statistics*. Springer Texts in Statistics. Springer, New York, 2006.

L. Wasserman and K. Roeder. High-dimensional variable selection. *The Annals of Statistics*, 37(5): 2178–2201, 2009.

P. Zhao and B. Yu. On model selection consistency of lasso. *Journal of Machine Learning Research*, 7:2541–2563, 2006.

H. Zou. The adaptive lasso and its oracle properties. *Journal of the American Statistical Association*, 101(476):1418–1429, 2006.

Journal of Machine Learning Research 13 (2012) 2145-2170 Submitted 11/11; Revised 6/12; Published 7/12

On the Necessity of Irrelevant Variables

David P. Helmbold DPH@SOE.UCSC.EDU
Department of Computer Science
University of California, Santa Cruz
Santa Cruz, CA 95064, USA

Philip M. Long PLONG@SV.NEC-LABS.COM
NEC Labs America
10080 N. Wolfe Rd, SW3-350
Cupertino, CA 95014, USA

Editor: Gábor Lugosi

Abstract

This work explores the effects of relevant and irrelevant boolean variables on the accuracy of classifiers. The analysis uses the assumption that the variables are conditionally independent given the class, and focuses on a natural family of learning algorithms for such sources when the relevant variables have a small advantage over random guessing. The main result is that algorithms relying predominately on irrelevant variables have error probabilities that quickly go to 0 in situations where algorithms that limit the use of irrelevant variables have errors bounded below by a positive constant. We also show that accurate learning is possible even when there are so few examples that one cannot determine with high confidence whether or not any individual variable is relevant.

Keywords: feature selection, generalization, learning theory

1. Introduction

When creating a classifier, a natural inclination is to only use variables that are obviously relevant since irrelevant variables typically decrease the accuracy of a classifier. On the other hand, this paper shows that the harm from irrelevant variables can be much less than the benefit from relevant variables and therefore it is possible to learn very accurate classifiers even when almost all of the variables are irrelevant. It can be advantageous to continue adding variables, even as their prospects for being relevant fade away. We show this with theoretical analysis and experiments using artificially generated data.

We provide an illustrative analysis that isolates the effects of relevant and irrelevant variables on a classifier's accuracy. We analyze the case in which variables complement one another, which we formalize using the common assumption of conditional independence given the class label. We focus on the situation where relatively few of the many variables are relevant, and the relevant variables are only weakly predictive.[1] Under these conditions, algorithms that cast a wide net can succeed while more selective algorithms fail.

We prove upper bounds on the error rate of a very simple learning algorithm that may include many irrelevant variables in its hypothesis. We also prove a contrasting lower bound on the error

1. Note that in many natural settings the individual variables are only weakly associated with the class label. This can happen when a lot of measurement error is present, as is seen in microarray data.

of every learning algorithm that uses mostly relevant variables. The combination of these results show that the simple algorithm's error rate approaches zero in situations where every algorithm that predicts with mostly relevant variables has an error rate greater than a positive constant.

Over the past decade or so, a number of empirical and theoretical findings have challenged the traditional rule of thumb described by Bishop (2006) as follows.

> One rough heuristic that is sometimes advocated is that the number of data points should be no less than some multiple (say 5 or 10) of the number of adaptive parameters in the model.

The Support Vector Machine literature (see Vapnik, 1998) views algorithms that compute apparently complicated functions of a given set of variables as linear classifiers applied to an expanded, even infinite, set of features. These empirically perform well on test data, and theoretical accounts have been given for this. Boosting and Bagging algorithms also generalize well, despite combining large numbers of simple classifiers, even if the number of such "base classifiers" is much more than the number of training examples (Quinlan, 1996; Breiman, 1998; Schapire et al., 1998). This is despite the fact that Friedman et al. (2000) showed the behavior of such classifiers is closely related to performing logistic regression on a potentially vast set of features (one for each possible decision tree, for example).

Similar effects are sometimes found even when the features added are restricted to the original "raw" variables. Figure 1, which is reproduced from Tibshirani et al. (2002), is one example. The curve labelled "te" is the test-set error, and this error is plotted as a function of the number of features selected by the Shrunken Centroids algorithm. The best accuracy is obtained using a classifier that depends on the expression level of well over 1000 genes, despite the fact that there are only a few dozen training examples.

It is impossible to tell if most of the variables used by the most accurate classifier in Figure 1 are irrelevant. However, we do know which variables are relevant and irrelevant in synthetic data (and can generate as many test examples as desired). Consider for the moment a simple algorithm applied to a simple source. Each of two classes is equally likely, and there are 1000 relevant boolean variables, 500 of which agree with the class label with probability $1/2+1/10$, and 500 which disagree with the class label with probability $1/2+1/10$. Another 99000 boolean variables are irrelevant. The algorithm is equally simple: it has a parameter β, and outputs the majority vote over those features (variables or their negations) that agree with the class label on a $1/2+\beta$ fraction of the training examples. Figure 2 plots three runs of this algorithm with 100 training examples, and 1000 test examples. Both the accuracy of the classifier and the fraction of relevant variables are plotted against the number of variables used in the model, for various values of β.[2] Each time, the best accuracy is achieved when an overwhelming majority of the variables used in the model are irrelevant, and those models with few ($< 25\%$) irrelevant variables perform far worse. Furthermore, the best accuracy is obtained with a model that uses many more variables than there are training examples. Also, accuracy over 90% is achieved even though there are few training examples and the correlation of the individual variables with the class label is weak. In fact, the number of examples is so small and the correlations are so weak that, for any individual feature, it is impossible to confidently tell whether or not the feature is relevant.

2. In the first graph, only the results in which fewer than 1000 features were chosen are shown, since including larger feature sets obscures the shape of the graph in the most interesting region, where relatively few features are chosen.

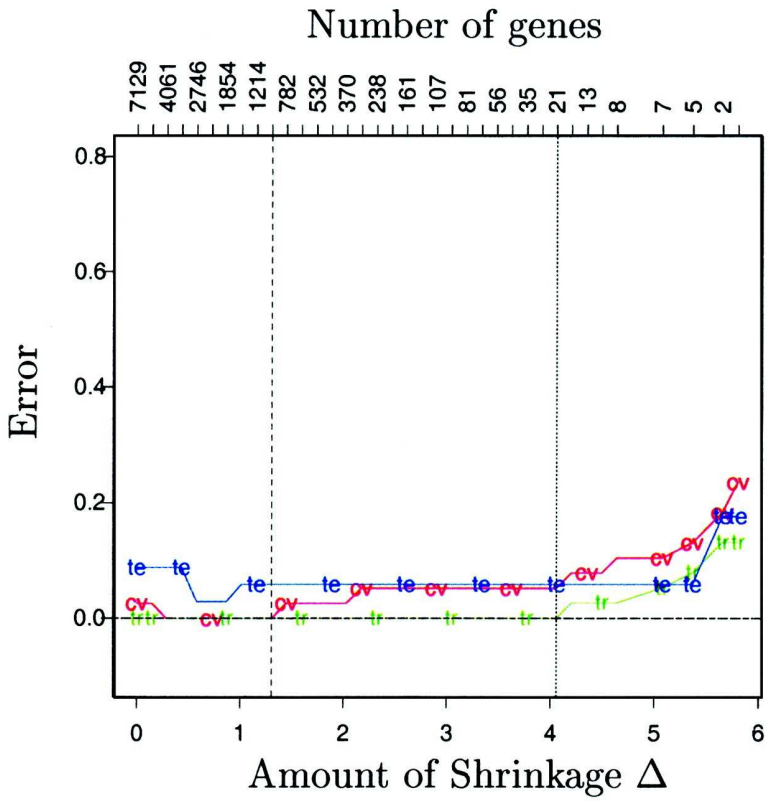


Figure 1: This graph is reproduced from Tibshirani et al. (2002). For a microarray data set, the training error, test error, and cross-validation error are plotted as a function both of the number of features (along the top) included in a linear model and a regularization parameter Δ (along the bottom).

Assume classifier f consists of a vote over n variables that are conditionally independent given the class label. Let k of the variables agree with the class label with probability $1/2+\gamma$, and the remaining $n-k$ variables agree with the label with probability $1/2$. Then the probability that f is incorrect is at most

$$\exp\left(\frac{-2\gamma^2 k^2}{n}\right) \tag{1}$$

(as shown in Section 3). The error bound decreases exponentially in the *square* of the number of relevant variables. The competing factor increases only *linearly* with the number of irrelevant variables. Thus, a very accurate classifier can be obtained with a feature set consisting predominantly of irrelevant variables.

In Section 4 we consider learning from training data where the variables are conditionally independent given the class label. Whereas Equation (1) bounded the error as a function of the number of variables n and relevant variables k in the *model*, we now use capital N and capital K for the total number of variables and number of relevant variables in the *data*. The $N-K$ irrelevant variables are independent of the label, agreeing with it with probability $1/2$. The K relevant variables either

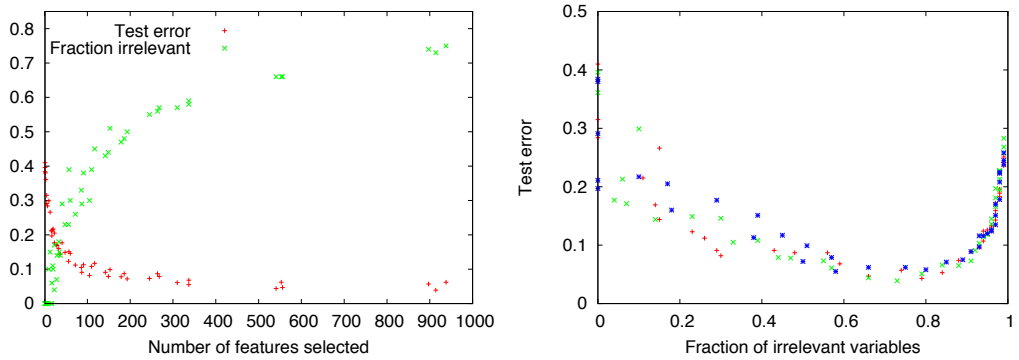


Figure 2: Left: Test error and fraction of irrelevant variables as a function of the number of features. Right: Scatter plot of test error rates (vertical) against fraction of irrelevant variables (horizontal).

agree with the label with probability $1/2+\gamma$ or with probability $1/2-\gamma$. We analyze an algorithm that chooses a value $\beta \geq 0$ and outputs a majority vote over all features that agree with the class label on at least $1/2+\beta$ of the training examples (as before, each feature is either a variable or its negation). Our Theorem 3 shows that if $\beta \leq \gamma$ and the algorithm is given m training examples, then the probability that it makes an incorrect prediction on an independent test example is at most

$$(1+o(1))\exp\left(-2\gamma^2 K\left(\frac{[1-8e^{-2(\gamma-\beta)^2 m}-\gamma)]_+^2}{1+8(N/K)e^{-2\beta^2 m}+\gamma}\right)\right),$$

where $[z]_+ \stackrel{\text{def}}{=} \max\{z,0\}$. (Throughout the paper, the "big Oh" and other asymptotic notation will be for the case where γ is small, $K\gamma$ is large, and N/K is large. Thus the edge of the relevant features and the fraction of features that are relevant both approach zero while the total number of relevant features increases. If K is not large relative to $1/\gamma^2$, even the Bayes optimal classifier is not accurate. No other assumptions about the relationship between the parameters are needed.)

When $\beta \leq \gamma/2$ and the number m of training examples satisfies $m \geq c/\gamma^2$ for an absolute constant c, we also show in Theorem 8 that the error probability is at most

$$(1+o(1))\exp\left(-\gamma^2 K^2/N\right). \tag{2}$$

If $N = o(\gamma^2 K^2)$, this error probability goes to zero. With only $\Theta(1/\gamma^2)$ examples, an algorithm cannot even tell with high confidence whether a relevant variable is positively or negatively associated with the class label, much less solve the more difficult problem of determining whether or not a variable is relevant. Indeed, this error bound is also achieved using $\beta = 0$, when, for each variable X_i, the algorithm includes either X_i or its negation in the vote.[3] Because bound (2) holds even when $\beta = 0$, it can be achieved by an algorithm that does not use knowledge of γ or K.

3. To be precise, the algorithm includes each variable or its negation when $\beta = 0$ and m is odd, and includes both the variable and its negation when m is even and the variable agrees with the class label exactly half the time. But, any time both a variable and its negation are included, their votes cancel. We will always use the smaller equivalent model obtained by removing such canceling votes.

Our upper bounds illustrate the potential rewards for algorithms that are "inclusive", using many of the available variables in their classifiers, even when this means that most variables in the model are irrelevant. We also prove a complementary lower bound that illustrates the potential cost when algorithms are "exclusive". We say that an algorithm is λ-exclusive if the expectation of the fraction of the variables used in its model that are relevant is at least λ. We show that any λ-exclusive policy has an error probability bounded below by $\lambda/4$ as K and N/K go to infinity and γ goes to 0 in such a way that the error rate obtained by the more "inclusive" setting $\beta = \gamma/2$ goes to 0. In particular, no λ-exclusive algorithm (where λ is a positive constant) can achieve a bound like (2).

Donoho and Jin (see Donoho and Jin, 2008; Jin, 2009) and Fan and Fan (2008), building on a line of research on multiple hypotheses testing (see Abramovich et al., 2006; Addario-Berry et al., 2010; Donoho and Jin, 2004, 2006; Meinshausen and Rice, 2006), performed analyses and simulations using sources with elements in common with the model studied here, including conditionally independent variables and a weak association between the variables and the class labels. Donoho and Jin also pointed out that their algorithm can produce accurate hypotheses while using many more irrelevant features than relevant ones. The main theoretical results proved in their papers describe conditions that imply that, if the relevant variables are too small a fraction of all the variables, and the number of examples is too small, then learning is impossible. The emphasis of our theoretical analysis is the opposite: algorithms can tolerate a large number of irrelevant variables, while using a small number of examples, and algorithms that avoid irrelevant variables, even to a limited extent, cannot learn as effectively as algorithms that cast a wider net. In particular, ours is the first analysis that we are aware of to have a result qualitatively like Theorem 13, which demonstrates the limitations of exclusive algorithms.

For the sources studied in this paper, there is a linear classifier that classifies most random examples correctly with a large margin, that is, most examples are not close to the decision boundary. The main motivation for our analysis was to understand the effects of relevant and irrelevant variables on generalization, but it is interesting to note that we get meaningful bounds in the extreme case that $m = \Theta(1/\gamma^2)$, whereas the margin-based bounds that we are aware of (such as Schapire et al. 1998, Koltchinskii and Panchenko 2002, Dasgupta and Long 2003 and Wang et al. 2008) are vacuous in this case. (Since these other bounds hold more generally, their overall strength is incomparable to our results.) Ng and Jordan (2001) showed that the Naive Bayes algorithm (which ignores class-conditional dependencies) converges relatively quickly, justifying its use when there are few examples. But their bound for Naive Bayes is also vacuous when $m = \Theta(1/\gamma^2)$. Bickel and Levina (2004) studied the case in which the class conditional distributions are Gaussians, and showed how an algorithm which does not model class conditional dependencies can perform nearly optimally in this case, especially when the number of variables is large. Bühlmann and Yu (2002) analyzed the variance-reduction benefits of Bagging with primary focus on the benefits of the smoother classifier that is obtained when ragged classifiers are averaged. As such it takes a different form than our analysis.

Our analysis demonstrates that certain effects are possible, but how important this is depends on how closely natural learning settings resemble our theoretical setting and the extent to which our analysis can be generalized. The conditional independence assumption is one way to express the intuitive notion that variables are not too redundant. A limit on the redundancy is needed for results like ours since, for example, a collection of $\Theta(k)$ perfectly correlated irrelevant variables would swamp the votes of the k relevant variables. On the other hand, many boosting algorithms minimize the potential for this kind of effect by choosing features in later iterations that make

errors on different examples then the previously chosen features. One relaxation of the conditional independence assumption is to allow each variable to conditionally depend on a limited number r of other variables, as is done in the formulation of the Lovasz Local Lemma (see Alon et al., 1992). As partial illustration of the robustness of the effects analyzed here, we generalize upper bound (1) to this case in Section 6.1. There we prove an error bound of $c(r+1)\exp\left(\frac{-2\gamma^2k^2}{n(r+1)}\right)$ when each variable depends on most r others. There are a number of ways that one could imagine relaxing the conditional independence assumption while still proving theorems of a similar flavor. Another obvious direction for generalization is to relax the strict categorization of variables into irrelevant and $(1/2+\gamma)$-relevant classes. We believe that many extensions of this work with different coverage and interpretability tradeoffs are possible. For example, our proof techniques easily give similar theorems when each relevant variable has a probability between $1/2+\gamma/2$ and $1/2+2\gamma$ of agreeing with the class label (as discussed in Section 6.2). Most of this paper uses the cleanest and simplest setting in order to focus attention on the main ideas.

We state some useful tail bounds in the next section, and Section 3 analyzes the error of simple voting classifiers. Section 4 gives bounds on the expected error of hypotheses learned from training data while Section 5 shows that, in certain situations, any exclusive algorithm must have high error while the error of some inclusive algorithms goes to 0. In Section 6.1 we bound the accuracy of voting classifiers under a weakened independence assumption and in Section 6.2 we consider relaxation of the assumption that all relevant variables have the same edge.

2. Tail Bounds

This section gathers together the several tail bounds that will be used in various places in the analysis. These bounds all assume that $U_1, U_2, \ldots, U_\ell$ are ℓ independent $\{0,1\}$-valued random variables and $U = \sum_{i=1}^{\ell} U_i$. We start with some upper bounds.

- The Hoeffding bound (see Pollard, 1984):

$$\mathbb{P}\left[\frac{1}{\ell}U - \mathbb{E}\left(\frac{1}{\ell}U\right) \geq \eta\right] \leq e^{-2\eta^2\ell}. \tag{3}$$

- The Chernoff bound see Angluin and Valiant, 1979; Motwani and Raghavan, 1995, and Appendix A.1. For any $\eta > 0$:

$$\mathbb{P}[U > (1+\eta)\mathbb{E}(U)] < \exp\left(-(1+\eta)\mathbb{E}(U)\ln\left(\frac{1+\eta}{e}\right)\right). \tag{4}$$

- For any $0 < \delta \leq 1$ (see Appendix A.2):

$$\mathbb{P}[U > 4\mathbb{E}(U) + 3\ln(1/\delta)] < \delta. \tag{5}$$

We also use the following lower bounds on the tails of distributions.

- If $\mathbb{P}[U_i = 1] = 1/2$ for all i, $\eta > 0$, and $\ell \geq 1/\eta^2$ then (see Appendix A.3):

$$\mathbb{P}\left[\frac{1}{\ell}U - \frac{1}{\ell}\mathbb{E}(U) \geq \eta\right] \geq \frac{1}{7\eta\sqrt{\ell}}\exp\left(-2\eta^2\ell\right) - \frac{1}{\sqrt{\ell}}. \tag{6}$$

- If $\mathbb{P}[U_i = 1] = 1/2$ for all i, then for all $0 \leq \eta \leq 1/8$ such that $\eta\ell$ is an integer[4] (see Appendix A.4):

$$\mathbb{P}\left[\frac{1}{\ell}U - \frac{1}{\ell}\mathbb{E}(U) \geq \eta\right] \geq \frac{1}{5}e^{-16\eta^2\ell}. \tag{7}$$

- A consequence of Slud's Inequality (1977) gives the following (see Appendix A.5). If $0 \leq \eta \leq 1/5$ and $\mathbb{P}[U_i = 1] = 1/2 + \eta$ for all i then:

$$\mathbb{P}\left[\frac{1}{\ell}U < 1/2\right] \geq \frac{1}{4}e^{-5\eta^2\ell}. \tag{8}$$

Note that the constants in the above bounds were chosen to be simple and illustrative, rather than the best possible.

3. The Accuracy of Models Containing Relevant and Irrelevant Variables

In this section we analyze the accuracy of the models (hypotheses) produced by the algorithms in Section 4. Each example is represented by a vector of N binary *variables* and a class designation. We use the following generative model:

- a random class designation from $\{0,1\}$ is chosen, with both classes equally likely, then
- each of K *relevant* variables are equal to the class designation with probability $1/2+\gamma$ (or with probability $1/2-\gamma$), and
- the remaining $N-K$ *irrelevant* variables are equal to the class label with probability $1/2$;
- all variables are conditionally independent given the class designation.

Which variables are relevant and whether each one is positively or negatively correlated with the class designations are chosen arbitrarily ahead of time.

A *feature* is either a variable or its complement. The $2(N-K)$ *irrelevant* features come from the irrelevant variables, the K *relevant* features agree with the class labels with probability $1/2+\gamma$, and the K *misleading* features agree with the class labels with probability $1/2-\gamma$.

We now consider models $\mathcal{M}$ predicting with a majority vote over a subset of the features. We use n for the total number of features in model $\mathcal{M}$, k for the number of relevant features, and ℓ for the number of misleading features (leaving $n-k-\ell$ irrelevant features). Since the votes of a variable and its negation "cancel out," we assume without loss of generality that models include at most one feature for each variable. Recall that $[z]_+ \stackrel{\text{def}}{=} \max\{z, 0\}$.

Theorem 1 *Let $\mathcal{M}$ be a majority vote of n features, k of which are relevant and ℓ of which are misleading (and $n-k-\ell$ are irrelevant). The probability that $\mathcal{M}$ predicts incorrectly is at most* $\exp\left(\frac{-2\gamma^2[k-\ell]_+^2}{n}\right)$.

4. For notational simplicity we omit the floors/ceilings implicit in the use of this bound.

Proof: If $\ell \geq k$ then the exponent is 0 and the bound trivially holds.

Suppose $k > \ell$. Model $\mathcal{M}$ predicts incorrectly only when at most half of its features are correct. The expected fraction of correct voters is $1/2 + \frac{\gamma(k-\ell)}{n}$, so, for $\mathcal{M}$'s prediction to be incorrect, the fraction of correct voters must be at least $\gamma(k-\ell)/n$ less than its expectation. Applying (3), this probability is at most

$$\exp\left(\frac{-2\gamma^2(k-\ell)^2}{n}\right).$$

□

The next corollary shows that even models where most of the features are irrelevant can be highly accurate.

Corollary 2 *If γ is a constant, $k - \ell = \omega(\sqrt{n})$ and $k = o(n)$, then the accuracy of the model approaches* 100% *while its fraction of irrelevant variables approaches* 1 *(as $n \to \infty$).*

For example, the conditions of Corollary 2 are satisfied when $\gamma = 1/4$, $k = 2n^{2/3}$ and $\ell = n^{2/3}$.

4. Learning

We now consider the problem of learning a model $\mathcal{M}$ from data. We assume that the algorithm receives m i.i.d. examples generated as described in Section 3. One test example is independently generated from the same distribution, and we evaluate the algorithm's *expected error*: the probability over training set and test example that its model makes an incorrect prediction on the test example (the "prediction model" of Haussler et al. 1994).

We define $\mathcal{M}_\beta$ to be the majority vote[5] of all features that equal the class label on at least $1/2+\beta$ of the training examples. To keep the analysis as clean as possible, our results in this section apply to algorithms that chose β as a function of the number of features N, the number of relevant features K, the edge of the relevant features γ, and training set size m, and then predict with $\mathcal{M}_\beta$. Note that this includes the algorithm that always choses $\beta = 0$ regardless of N, K, γ and m.

Recall that asymptotic notation will concern the case in which γ is small, $K\gamma$ is large, and N/K is large.

This section proves two theorems bounding the expected error rates of learned models. One can compare these bounds with a similar bound on the Bayes Optimal predictor that "knows" which features are relevant. This Bayes Optimal predictor for our generative model is a majority vote of the K relevant features, and has an error rate bounded by $e^{-2\gamma^2 K}$ (a bound as tight as the Hoeffding bound).

Theorem 3 *If $0 \leq \beta \leq \gamma$, then the expected error rate of $\mathcal{M}_\beta$ is at most*

$$(1+o(1))\exp\left(-2\gamma^2 K\left(\frac{[1-8e^{-2(\gamma-\beta)^2 m}-\gamma]_+^2}{1+8(N/K)e^{-2\beta^2 m}+\gamma}\right)\right).$$

Our proof of Theorem 3 starts with lemmas bounding the number of misleading, irrelevant, and relevant features in $\mathcal{M}_\beta$. These lemmas use a quantity $\delta > 0$ that will be determined later in the analysis.

5. If $\mathcal{M}_\beta$ is empty or the vote is tied then any default prediction, such as 1, will do.

Lemma 4 *With probability at least* $1-\delta$*, the number of misleading features in* $\mathcal{M}_\beta$ *is at most* $4Ke^{-2(\gamma+\beta)^2 m} + 3\ln(1/\delta)$.

Proof: For a particular misleading feature to be included in $\mathcal{M}_\beta$, Algorithm A must overestimate the probability that misleading feature equals the class label by at least $\beta+\gamma$. Applying (3), this happens with probability at most $e^{-2(\beta+\gamma)^2 m}$, so the expected number of misleading features in $\mathcal{M}_\beta$ is at most $Ke^{-2(\beta+\gamma)^2 m}$. Since each misleading feature is associated with a different independent variable, we can apply (5) with $\mathbb{E}(U) \leq Ke^{-2(\beta+\gamma)^2 m}$ to get the desired result. □

Lemma 5 *With probability at least* $1-2\delta$*, the number of irrelevant features in* $\mathcal{M}_\beta$ *is at most* $8Ne^{-2\beta^2 m} + 6\ln(1/\delta)$.

Proof: For a particular positive irrelevant feature to be included in $\mathcal{M}_\beta$, Algorithm A must overestimate the probability that the positive irrelevant feature equals the class label by β. Applying (3), this happens with probability at most $e^{-2\beta^2 m}$, so the expected number of irrelevant positive features in $\mathcal{M}_\beta$ is at most $(N-K)e^{-2\beta^2 m}$.

All of the events that variables agree with the label, for various variables, and various examples, are independent. So the events that various irrelevant variables are included in $\mathcal{M}_\beta$ are independent. Applying (5) with $\mathbb{E}(U) = (N-K)e^{-2\beta^2 m}$ gives that, with probability at least $1-\delta$, the number of irrelevant positive features in $\mathcal{M}_\beta$ is at most $4(N-K)e^{-2\beta^2 m}$.

A symmetric analysis establishes the same bound on the number of negative irrelevant features in $\mathcal{M}_\beta$. Adding these up completes the proof. □

Lemma 6 *With probability at least* $1-\delta$*, the number of relevant features in* $\mathcal{M}_\beta$ *is at least* $K - 4Ke^{-2(\gamma-\beta)^2 m} - 3\ln(1/\delta)$.

Proof: For a particular relevant feature to be excluded from $\mathcal{M}_\beta$, Algorithm A must underestimate the probability that the relevant feature equals the class label by at least $\gamma-\beta$. Applying (3), this happens with probability at most $e^{-2(\gamma-\beta)^2 m}$, so the expected number of relevant variables excluded from $\mathcal{M}_\beta$ is at most $Ke^{-2(\gamma-\beta)^2 m}$. Applying (5) as in the preceding two lemmas completes the proof. □

Lemma 7 *The probability that* $\mathcal{M}_\beta$ *makes an error is at most*

$$\exp\left(\frac{-2\gamma^2\left[K - 8Ke^{-2(\gamma-\beta)^2 m} - 6\ln(1/\delta)\right]_+^2}{K + 8Ne^{-2\beta^2 m} + 6\ln(1/\delta)}\right) + 4\delta.$$

for any $\delta > 0$ *and* $0 \leq \beta \leq \gamma$.

Proof: The bounds of Lemmas 4, 5, and 6 simultaneously hold with probability at least $1-4\delta$. Thus the error probability of $\mathcal{M}_\beta$ is at most 4δ plus the probability of error given that all three bounds hold. Plugging the three bounds into Theorem 1, (and over-estimating the number n of variables in the

model with K plus the bound of Lemma 5 on the number of irrelevant variables) gives a bound on $\mathcal{M}_\beta$'s error probability of

$$\exp\left(\frac{-2\gamma^2\left[(K-4Ke^{-2(\gamma-\beta)^2m}-3\ln(1/\delta))-(4Ke^{-2(\gamma+\beta)^2m}+3\ln(1/\delta))\right]_+^2}{K+8Ne^{-2\beta^2m}+6\ln(1/\delta)}\right) \tag{9}$$

when all three bounds hold. Under-approximating $(\gamma+\beta)^2$ with $(\gamma-\beta)^2$ and simplifying yields:

$$(9) \le \exp\left(\frac{-2\gamma^2\left[K-8Ke^{-2(\gamma-\beta)^2m}-6\ln(1/\delta)\right]_+^2}{K+8Ne^{-2\beta^2m}+6\ln(1/\delta)}\right).$$

Adding 4δ completes the proof. □

We are now ready to prove Theorem 3.

Proof (of Theorem 3): Using

$$\delta = \exp\left(-\frac{\gamma K}{6}\right)$$

in Lemma 7 bounds the probability that $\mathcal{M}_\beta$ makes a mistake by

$$\exp\left(\frac{-2\gamma^2\left[K-8Ke^{-2(\gamma-\beta)^2m}-\gamma K\right]_+^2}{K+8Ne^{-2\beta^2m}+\gamma K}\right)+4\exp\left(-\frac{\gamma K}{6}\right)$$
$$<\exp\left(\frac{-2\gamma^2K\left[1-8e^{-2(\gamma-\beta)^2m}-\gamma\right]_+^2}{1+\frac{8N}{K}e^{-2\beta^2m}+\gamma}\right)+4\exp\left(-\frac{\gamma K}{6}\right). \tag{10}$$

The first term is at least $e^{-2\gamma^2K}$, and

$$4\exp\left(-\frac{\gamma K}{6}\right) = o\left(e^{-2\gamma^2K}\right)$$

as $\gamma \to 0$ and $\gamma K \to \infty$, so (10) implies the bound

$$(1+o(1))\exp\left(\frac{-2\gamma^2K\left[1-8e^{-2(\gamma-\beta)^2m}-\gamma\right]_+^2}{1+\frac{8N}{K}e^{-2\beta^2m}+\gamma}\right)$$

as desired. □

The following theorem bounds the error in terms of just K, N, and γ when m is sufficiently large.

Theorem 8 *Suppose algorithm A produces models $\mathcal{M}_\beta$ where $0 \le \beta \le c\gamma$ for a constant $c \in [0,1)$.*

- *Then there is a constant b (depending only on c) such that whenever $m \ge b/\gamma^2$ the error of A's model is at most $(1+o(1))\exp\left(-\frac{\gamma^2K^2}{N}\right)$.*

- *If* $m = \omega(1/\gamma^2)$ *then the error of A's model is at most* $(1+o(1))\exp\left(\frac{-(2-o(1))\gamma^2K^2}{N}\right)$.

Proof Combining Lemmas 4 and 6 with the upper bound of N on the number of features in $\mathcal{M}_\beta$ as in Lemma 7's proof gives the following error bound on $\mathcal{M}_\beta$

$$\exp\left(\frac{-2\gamma^2\left[K-8Ke^{-2(\gamma-\beta)^2m}-6\ln(1/\delta)\right]_+^2}{N}\right)+2\delta$$

for any $\delta > 0$. Setting

$$\delta = \exp\left(-\frac{\gamma K}{6}\right)$$

and continuing as in the proof of Theorem 3 gives the bound

$$(1+o(1))\exp\left(\frac{-2\gamma^2K^2\left[1-2\left(4e^{-2(\gamma-\beta)^2m}+\gamma\right)\right]_+^2}{N}\right). \tag{11}$$

For the first part of the theorem, it suffices to show that the $[\cdots]_+^2$ term is at least $1/2$. Recalling that our analysis is for small γ, the term inside the $[\cdots]_+$ of (11) is at least

$$1-8e^{-2(1-c)^2\gamma^2m}-o(1).$$

When

$$m \geq \frac{\ln(32)}{2(1-c)^2\gamma^2}, \tag{12}$$

this term is at least $3/4-o(1)$, and thus its square is at least 1/2 for small enough γ, completing the proof of the first part of the theorem.

To see the second part of the theorem, since $m \in \omega(1/\gamma^2)$, the term of (11) inside the $[\cdots]_+$ is $1-o(1)$. ■

By examining inequality (12), we see that the constant b in Theorem 8 can be set to $\ln(32)/2(1-c)^2$.

Lemma 9 *The expected number of irrelevant variables in* $\mathcal{M}_\beta$ *is at least* $(N-K)e^{-16\beta^2m}$.

Proof Follows from inequality (7). ■

Corollary 10 *If K, N, and m are functions of* γ *such that*

$$\begin{aligned}
&\gamma \to 0,\\
&K^2/N \in \omega(\ln(1/\gamma)/\gamma^2),\\
&K = o(N) \text{ and}\\
&m = 2\ln(32)/\gamma^2
\end{aligned}$$

then if an algorithm outputs $\mathcal{M}_\beta$ *using a* β *in* $[0,\gamma/2]$*, it has an error that decreases super-polynomially (in* γ*), while the expected fraction of irrelevant variables in the model goes to* 1.

Note that Theorem 8 and Corollary 10 include non-trivial error bounds on the model $\mathcal{M}_0$ that votes all N variables (for odd sample size m).

5. Lower Bound

Here we show that any algorithm with an error guarantee like Theorem 8 must include many irrelevant features in its model. The preliminary version of this paper (Helmbold and Long, 2011) contains a related lower bound for algorithms that choose β as a function of N, K, m, and γ, and predict with $\mathcal{M}_\beta$. Here we present a more general lower bound that applies to algorithms outputting arbitrary hypotheses. This includes algorithms that use weighted voting (perhaps with L_1 regularization). In this section we

- set the number of features N, number of relevant features K, and sample size m as a functions of γ in such a way that Corollary 10 applies, and
- prove a constant lower bound for these combinations of values that holds for "exclusive" algorithms (defined below) when γ is small enough.

Thus, in this situation, "inclusive" algorithms relying on many irrelevant variables have error rates going to zero while every "exclusive" algorithm has an error rate bounded below by a constant.

The proofs in this section assume that all relevant variables are positively correlated with the class designation, so each relevant variable agrees with the class designation with probability $1/2+\gamma$. Although not essential for the results, this assumption simplifies the definitions and notation.[6] We also set $m = 2\ln(32)/\gamma^2$. This satisfies the assumption of Theorem 8 when $\beta \leq \gamma/2$ (see Inequality (12)).

Definition 11 *We say a classifier f includes a variable x_i if there is an input $(x_1,...,x_N)$ such that*

$$f(x_1,...,x_{i-1},x_i,x_{i+1},...,x_N) \neq f(x_1,...,x_{i-1},1-x_i,x_{i+1},...,x_N).$$

Let $V(f)$ be the set of variables included in f.

For a training set S, we will refer to the classifier output by algorithm A on S as $A(S)$. Let $\mathcal{R}$ be the set of relevant variables.

Definition 12 *We say that an algorithm A is λ-exclusive[7] if for every positive N, K, γ, and m, the expected fraction of the variables included in its hypothesis that are relevant is at least λ, that is, $\mathbb{E}\left(\frac{|V(A(S))\cap\mathcal{R}|}{|V(A(S))|}\right) \geq \lambda$.*

Our main lower bound theorem is the following.

6. The assumption that each relevant variable agrees with the class label with probability $1/2+\gamma$ gives a special case of the generative model described in Section 4, so the lower bounds proven here also apply to that more general setting.
7. The proceedings version of this paper (Helmbold and Long, 2011) used a different definition of λ-exclusive.

Theorem 13 *If*

$$K = \frac{1}{\gamma^2}\exp\left(\ln(1/\gamma)^{1/3}\right)$$
$$N = K\exp\left(\ln(1/\gamma)^{1/4}\right)$$
$$m = \frac{2\ln(32)}{\gamma^2}$$

then for any constant $\lambda > 0$ and any λ-exclusive algorithm A, the error rate of A is lower bounded by $\lambda/4 - o(1)$ as γ goes to 0.

Notice that this theorem provides a sharp contrast to Corollary 10. Corollary 10 shows that inclusive A using models $\mathcal{M}_\beta$ for any $0 \leq \beta \leq \gamma/2$ have error rates that goes to zero super-polynomially fast (in $1/\gamma$) under the assumptions of Theorem 13.

The values of K and N in Theorem 13 are chosen to make the proof convenient, but other values would work. For example, decreasing K and/or increasing N would make the lower bound part of Theorem 13 easier to prove. There is some slack to do so while continuing to ensure that the upper bound of Corollary 10 goes to 0.

As the correlation of variables with the label over the sample plays a central role in our analysis, we will use the following definition.

Definition 14 *If a variable agrees with the class label on $1/2+\eta$ of the training set then it has* (empirical) edge η.

The proof of Theorem 13 uses a critical value of β, namely $\beta^* = \gamma\ln(N/K)/10\ln(32)$, with the property that both:

$$\frac{\mathbb{E}\left(|\mathcal{M}_{\beta^*} \cap \mathcal{R}|\right)}{\mathbb{E}\left(|\mathcal{M}_{\beta^*}|\right)} \to 0, \tag{13}$$
$$\mathbb{E}\left(|\mathcal{M}_{\beta^*} \cap \mathcal{R}|\right) \in o(1/\gamma^2) \tag{14}$$

as $\gamma \to 0$.

Intuitively, (13) means that any algorithm that uses most of the variables having empirical edge at least β^* cannot be λ-exclusive. On the other hand, (14) implies that if the algorithm restricts itself to variables with empirical edges greater than β^* then it does not include enough relevant variables to be accurate. The proof must show that *arbitrary* algorithms frequently include either too many irrelevant variables to be λ-exclusive or too few relevant ones to be accurate. See Figure 3 for some useful facts about γ, m, and β^*.

To prove the lower bound, borrowing a technique from Ehrenfeucht et al. (1989), we will assume that the K relevant variables are randomly selected from the N variables, and lower bound the error with respect to this random choice, along with the training and test data. This will then imply that, for each algorithm, there will be a choice of the K relevant variables giving the same lower bound with respect only to the random choice of the training and test data. We will always use relevant variables that are positively associated with the class label, agreeing with it with probability $1/2+\gamma$.

Proof [of Theorem 13] Fix any learning algorithm A, and let $A(S)$ be the hypothesis produced by A from sample S. Let $n(S)$ be the number of variables included in $A(S)$ and let $\beta(S)$ be the $n(S)$'th largest empirical (w.r.t. S) edge of a variable.

$$b = 2\ln(32)$$
$$m = \frac{b}{\gamma^2} = \frac{2\ln(32)}{\gamma^2}$$
$$\beta^* = \frac{\gamma\ln(N/K)}{5b} = \frac{\gamma\ln(1/\gamma)^{1/4}}{10\ln(32)}$$

Figure 3: Some useful facts relating b, γ, m and β^* under the assumptions of Theorem 13.

Let q_γ be the probability that $\beta(S) \geq \beta^* = \gamma\ln(N/K)/10\ln(32)$. We will show in Section 5.2 that if A is λ-exclusive then $\lambda \leq q_\gamma + o(1)$ (as γ goes to 0). We will also show in Section 5.3 that the expected error of A is at least $q_\gamma/4 - o(1)$ as γ goes to 0. Therefore any λ-exclusive algorithm A has an expected error rate at least $\lambda/4 - o(1)$ as γ goes to 0. ■

Before attacking the two parts of the proof alluded to above, we need a subsection providing some basic results about relevant variables and optimal algorithms.

5.1 Relevant Variables and Good Hypotheses

This section proves some useful facts about relevant variables and good hypotheses. The first lemma is a lower bound on the accuracy of a model in terms of the number of relevant variables.

Lemma 15 *If $\gamma \in [0, 1/5]$ then any classifier using k relevant variables has an error probability at least $\frac{1}{4}e^{-5\gamma^2 k}$.*

Proof: The usual Naive Bayes calculation (see Duda et al., 2000) implies that the optimal classifier over a certain set V of variables is a majority vote over $V \cap \mathcal{R}$. Applying the lower tail bound (8) then completes the proof. □

Our next lemma shows that, given a sample, the probability that a variable is relevant (positively correlated with the class label) is monotonically increasing in its empirical edge.

Lemma 16 *For two variables x_i and x_j, and any training set S of m examples,*

- *$\mathbb{P}[x_i \text{ relevant} \mid S] > \mathbb{P}[x_j \text{ relevant} \mid S]$ if and only if the empirical edge of x_i in S is greater than the empirical edge of x_j in S, and*
- *$\mathbb{P}[x_i \text{ relevant} \mid S] = \mathbb{P}[x_j \text{ relevant} \mid S]$ if and only if the empirical edge of x_i in S is equal to the empirical edge of x_j in S.*

Proof Since the random choice of $\mathcal{R}$ does not effect that marginal distribution over the labels, we can generate S by picking the labels for all the examples first, then $\mathcal{R}$, and finally the values of the variables on all the examples. Thus if we can prove the lemma after conditioning on the values of the class labels, then scaling all of the probabilities by 2^{-m} would complete the proof. So, let us fix

the values of the class labels, and evaluate probabilities only with respect to the random choice of the relevant variables $\mathcal{R}$, and the values of the variables.

Let

$$\Delta = \mathbb{P}[x_i \in \mathcal{R} | S] - \mathbb{P}[x_j \in \mathcal{R} | S].$$

First, by subtracting off the probabilities that both variables are relevant, we have

$$\Delta = \mathbb{P}\left[x_i \in \mathcal{R}, x_j \notin \mathcal{R} \middle| S\right] - \mathbb{P}\left[x_i \notin \mathcal{R}, x_j \in \mathcal{R} \middle| S\right].$$

Let ONE be the event that exactly one of x_i or x_j is relevant. Then

$$\Delta = \left(\mathbb{P}\left[x_i \in \mathcal{R}, x_j \notin \mathcal{R} \middle| S, \text{ONE}\right] - \mathbb{P}\left[x_i \notin \mathcal{R}, x_j \in \mathcal{R} \middle| S, \text{ONE}\right]\right)\mathbb{P}[\text{ONE}].$$

So $\Delta > 0$ if and only if

$$\Delta' \stackrel{\text{def}}{=} \mathbb{P}\left[x_i \in \mathcal{R}, x_j \notin \mathcal{R} \middle| S, \text{ONE}\right] - \mathbb{P}\left[x_i \notin \mathcal{R}, x_j \in \mathcal{R} \middle| S, \text{ONE}\right] > 0$$

(and similarly for $\Delta = 0$ if and only if $\Delta' = 0$). If $\mathbb{Q}$ is the distribution obtained by conditioning on ONE, then

$$\Delta' = \mathbb{Q}\left[x_i \in \mathcal{R}, x_j \notin \mathcal{R} \middle| S\right] - \mathbb{Q}\left[x_i \notin \mathcal{R}, x_j \in \mathcal{R} \middle| S\right].$$

Let S_i be the values of variable i in S, and define S_j similarly for variable j. Let S' be the values of the other variables. Since we have already conditioned on the labels, after also conditioning on ONE (i.e., under the distribution $\mathbb{Q}$), the pair (S_i, S_j) is independent of S'. For each S_i we have $\mathbb{P}[S_i \mid x_i \notin \mathcal{R}] = \mathbb{Q}[S_i \mid x_i \notin \mathcal{R}]$. Furthermore, by symmetry,

$$\mathbb{Q}\left[x_i \in \mathcal{R}, x_j \notin \mathcal{R} \middle| S'\right] = \mathbb{Q}\left[x_i \notin \mathcal{R}, x_j \in \mathcal{R} \middle| S'\right] = \frac{1}{2}.$$

Thus, by using Bayes' Rule on each term, we have

$$\begin{aligned}
\Delta' &= \mathbb{Q}\left[x_i \in \mathcal{R}, x_j \notin \mathcal{R} \middle| S_i, S_j, S'\right] - \mathbb{Q}\left[x_i \notin \mathcal{R}, x_j \in \mathcal{R} \middle| S_i, S_j, S'\right] \\
&= \frac{\mathbb{Q}\left[S_i, S_j \middle| x_i \in \mathcal{R}, x_j \notin \mathcal{R}, S'\right] - \mathbb{Q}\left[S_i, S_j \middle| x_i \notin \mathcal{R}, x_j \in \mathcal{R}, S'\right]}{2\,\mathbb{Q}[S_i, S_j | S']} \\
&= \frac{(1/2+\gamma)^{m_i}(1/2-\gamma)^{m-m_i} - (1/2+\gamma)^{m_j}(1/2-\gamma)^{m-m_j}}{2^{m+1}\,\mathbb{Q}[S_i, S_j]},
\end{aligned}$$

where m_i and m_j are the numbers of times that variables x_i and x_j agree with the label in sample S. The proof concludes by observing that Δ' is positive exactly when $m_i > m_j$ and zero exactly when $m_i = m_j$. ■

Because, in this lower bound proof, relevant variables are always positively associated with the class label, we will use a variant of $\mathcal{M}_\beta$ which only considers positive features.

Definition 17 *Let $\mathcal{V}_\beta$ be a vote over the variables with empirical edge at least β.*

When there is no chance of confusion, we will refer to the set of variables in $\mathcal{V}_\beta$ also as $\mathcal{V}_\beta$ (rather than $V(\mathcal{V}_\beta)$).

We now establish lower bounds on the probability of variables being included in $\mathcal{V}_\beta$ (here β can be a function of γ, but does not depend on the particular sample S).

Lemma 18 *If $\gamma \leq 1/8$ and $\beta \geq 0$ then the probability that a given variable has empirical edge at least β is at least*

$$\frac{1}{5}\exp\left(-16\beta^2 m\right).$$

If in addition $m \geq 1/\beta^2$, then the probability that a given variable has empirical edge at least β is at least

$$\frac{1}{7\beta\sqrt{m}}\exp\left(-2\beta^2 m\right) - \frac{1}{\sqrt{m}}.$$

Proof: Since relevant variables agree with the class label with probability $1/2+\gamma$, the probability that a relevant variable has empirical edge at least β is lower bounded by the probability that an irrelevant variable has empirical edge at least β. An irrelevant variable has empirical edge at least β only when it agrees with the class on $1/2+\beta$ of the sample. Applying Bound (7), this happens with probability at least $\frac{1}{5}\exp\left(-16\beta^2 m\right)$. The second part uses Bound (6) instead of (7). □

We now upper bound the probability of a relevant variable being included in $\mathcal{V}_\beta$, again for β that does not depend on S.

Lemma 19 *If $\beta \geq \gamma$, the probability that a given relevant variable has empirical edge at least β is at most $e^{-2(\beta-\gamma)^2 m}$.*

Proof: Use (3) to bound the probability that a relevant feature agrees with the class label $\beta-\gamma$ more often than its expected fraction of times. □

5.2 Bounding λ-Exclusiveness

Recall that $n(S)$ is the number of variables used by $A(S)$, and $\beta(S)$ is the edge of the variable whose rank, when the variables are ordered by their empirical edges, is $n(S)$. We will show that: if $A(S)$ is λ-exclusive, then there is reasonable probability that $\beta(S)$ is at least the critical value $\beta^* = \gamma\ln(N/K)/5b$. Specifically, if A is λ-exclusive, then, for any small enough γ, we have $\mathbb{P}[\beta(S) \geq \beta^*] > \lambda/2$.

Suppose, given the training set S, the variables are sorted in decreasing order of empirical edge (breaking ties arbitrarily, say using the variable index). Let $\mathcal{V}_{S,k}$ consist of the first k variables in this sorted order, the "top k" variables.

Since for each sample S and each variable x_i, the probability $\mathbb{P}\left[x_i \text{ relevant}\middle|S\right]$ decreases as the empirical edge of x_i decreases (Lemma 16), the expectation $\mathbb{E}\left(\frac{|\mathcal{V}_{S,k}\cap\mathcal{R}|}{|\mathcal{V}_{S,k}|}\,\middle|\,S\right)$ is non-increasing with k.

Furthermore, Lemma 16 also implies that for each sample S, we have

$$\mathbb{E}\left(\frac{|V(A(S))\cap\mathcal{R}|}{|V(A(S))|}\,\middle|\,S\right) \leq \mathbb{E}\left(\frac{|\mathcal{V}_{S,n(S)}\cap\mathcal{R}|}{|\mathcal{V}_{S,n(S)}|}\,\middle|\,S\right).$$

Therefore, by averaging over samples, for each γ we have

$$\mathbb{E}\left(\frac{|V(A(S))\cap\mathcal{R}|}{|V(A(S))|}\right) \leq \mathbb{E}\left(\frac{|\mathcal{V}_{S,n(S)}\cap\mathcal{R}|}{|\mathcal{V}_{S,n(S)}|}\right). \tag{15}$$

Note that the numerators in the expectations are never greater than the denominators. We will next give upper bounds on $|\mathcal{V}_{\beta^*} \cap \mathcal{R}|$ and lower bounds on $|\mathcal{V}_{\beta^*}|$ that each hold with probability $1-\gamma$.

The next step is a high-confidence upper bound on $|\mathcal{V}_{\beta^*} \cap \mathcal{R}|$. From Lemma 19, the probability that a particular relevant variable is in $\mathcal{V}_{\beta^*}$ is at most (recall that $m = b/\gamma^2$)

$$\begin{aligned} e^{-2(\beta^*-\gamma)^2 m} &= e^{-2b(\beta^*/\gamma-1)^2} \\ &= \exp\left(-2b\left(\frac{\ln(1/\gamma)^{1/4}}{5b} - 1\right)^2\right) \\ &= \exp\left(\frac{-2\ln(1/\gamma)^{1/2}}{25b} + \frac{4\ln(1/\gamma)^{1/4}}{5} - 2b\right) \\ &< \exp\left(\frac{-2\ln(1/\gamma)^{1/2}}{25b} + \frac{4\ln(1/\gamma)^{1/4}}{5}\right). \end{aligned}$$

Let $p_{\text{rel}} = \exp\left(\frac{-2\ln(1/\gamma)^{1/2}}{25b} + \frac{4\ln(1/\gamma)^{1/4}}{5}\right)$ be this upper bound, and note that p_{rel} drops to 0 as γ goes to 0, but a rate slower than γ^ε for any ε. The number of relevant variables in $\mathcal{V}_{\beta^*}$ has a binomial distribution with parameters K and p where $p < p_{\text{rel}}$. The standard deviation of this distribution is

$$\sigma = \sqrt{Kp(1-p)} < \sqrt{Kp} < \frac{\exp\left(\frac{\ln(1/\gamma)^{1/3}}{2}\right)\sqrt{p_{\text{rel}}}}{\gamma}. \tag{16}$$

Using the Chebyshev bound,

$$\mathbb{P}[|X - \mathbb{E}(X)| > a\sigma] \leq \frac{1}{a^2} \tag{17}$$

with $a = 1/\sqrt{\gamma}$ gives that

$$\begin{aligned} \mathbb{P}\left[|\mathcal{V}_{\beta^*} \cap \mathcal{R}| - Kp > \frac{\sigma}{\sqrt{\gamma}}\right] &\leq \gamma, \\ \mathbb{P}\left[|\mathcal{V}_{\beta^*} \cap \mathcal{R}| > Kp_{\text{rel}} + \frac{\sigma}{\sqrt{\gamma}}\right] &\leq \gamma. \end{aligned} \tag{18}$$

Since $\sigma < \sqrt{Kp} < \sqrt{Kp_{\text{rel}}}$ by (16), we have $\sigma\sqrt{Kp_{\text{rel}}} < Kp_{\text{rel}}$. Substituting the values of K and p_{rel} into the square-root yields

$$\begin{aligned} Kp_{\text{rel}} &> \sigma \times \frac{\exp\left(\frac{\ln(1/\gamma)^{1/3}}{2}\right)\exp\left(\frac{-\ln(1/\gamma)^{1/2}}{25b} + \frac{2\ln(1/\gamma)^{1/4}}{5}\right)}{\gamma} \\ &> \sigma/\sqrt{\gamma}, \end{aligned}$$

for small enough γ. Combining with (18), we get that

$$\mathbb{P}\left[|\mathcal{V}_{\beta^*} \cap \mathcal{R}| > 2Kp_{\text{rel}}\right] \leq \gamma \tag{19}$$

holds for small enough γ.

Using similar reasoning, we now obtain a lower bound on the expected number of variables in $\mathcal{V}_{\beta^*}$. Lemma 18 shows that, for each variable, the probability of the variable having empirical edge β^* is at least

$$\begin{aligned}\frac{1}{7\beta^*\sqrt{m}}\exp\left(-2\beta^{*2}m\right)-\frac{1}{\sqrt{m}} &= \frac{5\sqrt{b}}{7\ln(1/\gamma)^{1/4}}\exp\left(-2\frac{\ln(1/\gamma)^{1/2}}{25b}\right)-\frac{\gamma}{\sqrt{b}}\\ &> \frac{\sqrt{b}}{2\ln(1/\gamma)^{1/4}}\exp\left(-2\frac{\ln(1/\gamma)^{1/2}}{25b}\right)\end{aligned}$$

for sufficiently small γ. Since the empirical edges of different variables are independent, the probability that at least n variables have empirical edge at least β^* is lower bounded by the probability of at least n successes from the binomial distribution with parameters N and p_{irrel} where

$$p_{\text{irrel}} = \frac{\sqrt{b}}{2\ln(1/\gamma)^{1/4}}\exp\left(-2\frac{\ln(1/\gamma)^{1/2}}{25b}\right).$$

If, now, we define σ to be the standard deviation of this binomial distribution, then, like before, $\sigma=\sqrt{Np_{\text{irrel}}(1-p_{\text{irrel}})}<\sqrt{Np_{\text{irrel}}}$, and

$$\begin{aligned}Np_{\text{irrel}}/2 &> \sigma\sqrt{Np_{\text{irrel}}}/2\\ &= \frac{\sigma}{2}\times\frac{\exp((1/2)(\ln(1/\gamma)^{1/4}+\ln(1/\gamma)^{1/3}))}{\gamma}\times\frac{b^{1/4}}{\sqrt{2}\ln(1/\gamma)^{1/8}}\exp\left(-\frac{\ln(1/\gamma)^{1/2}}{25b}\right),\end{aligned}$$

so that, for small enough γ, $Np_{\text{irrel}}/2>\sigma/\sqrt{\gamma}$. Therefore applying the Chebyshev bound (17) with $a=1/\sqrt{\gamma}$ gives (for sufficiently small γ)

$$\mathbb{P}\left[|\mathcal{V}_{\beta^*}|<\frac{Np_{\text{irrel}}}{2}\right]\le\mathbb{P}\left[|\mathcal{V}_{\beta^*}|<Np_{\text{irrel}}-\sigma/\sqrt{\gamma}\right]<\gamma. \tag{20}$$

Recall that

$$q_\gamma=\mathbb{P}[\beta(S)\ge\beta^*]=\mathbb{P}\left[n(S)\le|\mathcal{V}_{\beta^*}|\right].$$

If A is λ-exclusive then, using (15), we have

$$\begin{aligned}\lambda\le\mathbb{E}\left(\frac{|V(A(S))\cap\mathcal{R}|)}{|V(A(S))|}\right)&\le\mathbb{E}\left(\frac{|\mathcal{V}_{S,n(S)}\cap\mathcal{R}|}{|\mathcal{V}_{S,n(S)}|}\right)\\ &\le(1-q_\gamma)\mathbb{E}\left(\frac{|\mathcal{V}_{S,n(S)}\cap\mathcal{R}|}{|\mathcal{V}_{S,n(S)}|}\;\middle|\;|\mathcal{V}_{\beta^*}|<n(S)\right)+q_\gamma\\ &\le(1-q_\gamma)\mathbb{E}\left(\frac{|\mathcal{V}_{\beta^*}\cap\mathcal{R}|}{|\mathcal{V}_{\beta^*}|}\;\middle|\;|\mathcal{V}_{\beta^*}|<n(S)\right)+q_\gamma\\ &\le(1-q_\gamma)\left(\frac{2Kp_{\text{rel}}}{Np_{\text{irrel}}/2}+2\gamma\right)+q_\gamma\end{aligned}$$

where we use the upper and lower bounds from Equations (19) and (20) that each hold with probability $1-\gamma$. Note that the ratio

$$\frac{2Kp_{\text{rel}}}{Np_{\text{irrel}}/2} \leq \frac{2\frac{e^{\ln(1/\gamma)^{1/3}}}{\gamma^2}\exp\left(\frac{-2\ln(1/\gamma)^{1/2}}{25b}+\frac{4\ln(1/\gamma)^{1/4}}{5}\right)}{\frac{e^{\ln(1/\gamma)^{1/3}}e^{\ln(1/\gamma)^{1/4}}}{4\gamma^2}\frac{\sqrt{b}}{\ln(1/\gamma)^{1/4}}\exp\left(-2\frac{\ln(1/\gamma)^{1/2}}{25b}\right)}$$

$$= \frac{8\ln(1/\gamma)^{1/4}\exp\left(\frac{-2\ln(1/\gamma)^{1/2}}{25b}+\frac{4\ln(1/\gamma)^{1/4}}{5}\right)}{\sqrt{b}\,e^{\ln(1/\gamma)^{1/4}}\exp\left(-2\frac{\ln(1/\gamma)^{1/2}}{25b}\right)}$$

$$= \frac{8\ln(1/\gamma)^{1/4}\exp\left(\frac{-\ln(1/\gamma)^{1/4}}{5}\right)}{\sqrt{b}}$$

which goes to 0 as γ goes to 0. Therefore,

$$\lambda \leq \mathbb{E}\left(\frac{|V(A(S))\cap\mathcal{R}|)}{|V(A(S))|}\right) \leq q_\gamma + o(1)$$

which implies that,

$$q_\gamma = \mathbb{P}[\beta(S) \geq \beta^*] \geq \lambda - o(1)$$

as γ goes to 0.

5.3 Large Error

Call a variable *good* if it is relevant and its empirical edge is at least β^* in the sample. Let p be the probability that a relevant variable is good. Thus the number of good variables is binomially distributed with parameters K and p. We have that the expected number of good variables is pK and the variance is $Kp(1-p) < Kp$. By Chebyshev's inequality, we have

$$\mathbb{P}\left[\#\text{ good vars} \geq Kp + a\sqrt{Kp}\right] \leq \mathbb{P}\left[\#\text{ good vars} \geq Kp + a\sqrt{Kp(1-p)}\right] \leq \frac{1}{a^2}, \qquad (21)$$

and setting $a = \sqrt{Kp}$, this gives

$$\mathbb{P}[\#\text{ good vars} \geq 2Kp] \leq \frac{1}{Kp}.$$

By Lemma 19, $Kp \leq Ke^{-2(\beta^*-\gamma)^2 m} = Ke^{-2b\left(\ln(1/\gamma)^{1/4}/5b-1\right)^2}$, so

$$\ln(Kp) \leq \ln K - 2b\left(\frac{\ln(1/\gamma)^{1/2}}{25b^2} - \frac{2\ln(1/\gamma)^{1/4}}{5b} + 1\right),$$

$$\ln(Kp) \leq 2\ln(1/\gamma) + \ln(1/\gamma)^{1/3} - 2b\left(\frac{\ln(1/\gamma)^{1/2}}{25b^2} - \frac{2\ln(1/\gamma)^{1/4}}{5b} + 1\right).$$

So for small enough γ,

$$\ln(Kp) \leq 2\ln(1/\gamma) - \frac{\ln(1/\gamma)^{1/2}}{25b}$$

and thus $Kp \in o(1/\gamma^2)$.

So if $Kp > 1/\gamma$, then with probability at least $1-\gamma$, there are less than $2Kp \in o(1/\gamma^2)$ good variables. On the other hand, if $Kp < 1/\gamma$, then, setting $a = \sqrt{1/\gamma}$ in bound (21) gives that the probability that there are more than $2/\gamma$ good variables is at most γ. So in either case the probability that there are more than $\frac{2}{\gamma^2}\exp\left(-\ln(1/\gamma)^{1/2}/25b\right)$ good variables is at most γ (for small enough γ).

So if $\mathbb{P}[\beta(S) \geq \beta^*] \geq q_\gamma$, then with probability at least $q_\gamma - \gamma$ algorithm A is using a hypothesis with at most $\frac{2}{\gamma^2}\exp\left(-\ln(1/\gamma)^{1/2}/25b\right)$ relevant variables. Applying Lemma 15 yields the following lower bound on the probability of error:

$$(q_\gamma - \gamma)\frac{1}{4}\exp\left(-10\exp\left(-\ln(1/\gamma)^{1/2}/25b\right)\right). \tag{22}$$

Since the limit of (22) for small γ is $q_\gamma/4$, this completes the proof of Theorem 13.

6. Relaxations of Some Assumptions

To keep the analysis clean, and facilitate the interpretation of the results, we have analyzed an idealized model. In this section, we briefly consider the consequences of some relaxations of our assumptions.

6.1 Conditionally Dependent Variables

Theorem 1 can be generalized to the case in which there is limited dependence among the variables, after conditioning on the class designation, in a variety of ways. For example, suppose that there is a degree-r graph G whose nodes are variables, and such that, conditioned on the label, each variable is independent of all variables not connected to it by an edge in G. Assume that k variables agree with the label with probability $1/2+\gamma$, and the $n-k$ agree with the label with probability $1/2$. Let us say that a source like this *has r-local dependence*. Then applying a Chernoff-Hoeffding bound for such sets of random variables due to Pemmaraju (2001), if $r \leq n/2$, one gets a bound of $c(r+1)\exp\left(\frac{-2\gamma^2k^2}{n(r+1)}\right)$ the probability of error.

6.2 Variables with Different Strengths

We have previously assumed that all relevant variables are equally strongly associated with the class label. Our analysis is easily generalized to the situation when the strengths of associations fall in an interval $[\gamma_{\min}, \gamma_{\max}]$. Thus relevant variables agree with the class label with probability at least $1/2+\gamma_{\min}$ and misleading variables agree with the class label with probability at least $1/2-\gamma_{\max}$. Although a sophisticated analysis would take each variable's degree of association into account, it is possible to leverage our previous analysis with a simpler approach. Using the $1/2+\gamma_{\min}$ and $1/2-\gamma_{\max}$ underestimates on the probability that relevant variables and misleading variables agree with the class label leads to an analog of Theorem 1. This analog says that models voting n variables, k of which are relevant and ℓ of which are misleading, have error probabilities bounded by

$$\exp\left(\frac{-2[\gamma_{\min}k - \gamma_{\max}\ell]_+^2}{n}\right).$$

We can also use the upper and lower bounds on association to get high-confidence bounds (like those of Lemmas 4 and 6) on the numbers of relevant and misleading features in models $\mathcal{M}_\beta$. This leads to an analog of Theorem 3 bounding the expected error rate of $\mathcal{M}_\beta$ by

$$(1+o(1))\exp\left(\frac{-2\gamma_{\min}^2 K\left[1-4(1+\gamma_{\max}/\gamma_{\min})e^{-2(\gamma_{\min}-\beta)^2 m}-\gamma_{\min}\right]_+^2}{1+\frac{8N}{K}e^{-2\beta^2 m}+\gamma_{\min}}\right)$$

when $0 \leq \beta \leq \gamma$ and $\gamma_{\max} \in o(1)$. Note that γ in Theorem 3 is replaced by $\gamma_{\min}$ here, and $\gamma_{\max}$ only appears in the $4(1+\gamma_{\max}/\gamma_{\min})$ factor (which replaces an "8" in the original theorem).

Continuing to mimic our previous analysis gives analogs to Theorem 8 and Corollary 10. These analogs imply that if $\gamma_{\max}/\gamma_{\min}$ is bounded then algorithms using small β perform well in the same limiting situations used in Section 5 to bound the effectiveness of exclusive algorithms.

A more sophisticated analysis keeping better track of the degree of association between relevant variables and the class label may produce better bounds. In addition, if the variables have varying strengths then it makes sense to consider classifiers that assign different voting weights to the variables based on their estimated strength of association with the class label. An analysis that takes account of these issues is a potentially interesting subject for further research.

7. Conclusions

We analyzed learning when there are few examples, a small fraction of the variables are relevant, and the relevant variables are only weakly correlated with the class label. In this situation, algorithms that produce hypotheses consisting predominately of irrelevant variables can be highly accurate (with error rates going to 0). Furthermore, this inclusion of many irrelevant variables is essential. Any algorithm limiting the expected fraction of irrelevant variables in its hypotheses has an error rate bounded below by a constant. This is in stark contrast with many feature selection heuristics that limit the number of features to a small multiple of the number of examples, or that limit the classifier to use variables that pass stringent statistical tests of association with the class label.

These results have two implications on the practice of machine learning. First, they show that the engineering practice of producing models that include enormous numbers of variables is sometimes justified. Second, they run counter to the intuitively appealing view that accurate class prediction "validates" the variables used by the predictor.

Acknowledgments

We thank Aravind Srinivasan for his help.

Appendix A. Proofs of Tail Bounds

This appendix has the proofs of the tail bounds used in the paper.

A.1 Proof of (4)

Equation 4.1 from Motwani and Raghavan (1995) is

$$\mathbb{P}[U > (1+\eta)\mathbb{E}(U)] < \left(\frac{e^{\eta}}{(1+\eta)^{1+\eta}}\right)^{\mathbb{E}(U)}$$

and holds for independent 0-1 valued U_i's each with (possibly different) probabilities $p_i = P(U_i = 1)$ where $0 < p_i < 1$ and $\eta > 0$. Taking the logarithm of the RHS, we get

$$\begin{aligned} \ln(\text{RHS}) &= \mathbb{E}(U)\,(\eta - (1+\eta)\ln(1+\eta)) \\ &< \mathbb{E}(U)\,(\eta + 1 - (1+\eta)\ln(1+\eta)) \\ &= -\mathbb{E}(U)(\eta+1)(\ln(1+\eta)-1), \end{aligned}$$

which implies (4).

A.2 Proof of (5)

Using (4) with $\eta = 3 + 3\ln(1/\delta)/\mathbb{E}(U)$ gives

$$\begin{aligned} \mathbb{P}[U > 4\mathbb{E}(U) + 3\ln(1/\delta)] &< \exp\left(-(4\mathbb{E}(U) + 3\ln\delta)\ln\left(\frac{4+3\ln(1/\delta)/\mathbb{E}(U)}{e}\right)\right) \\ &< \exp\left(-(3\ln(1/\delta)\ln\left(\frac{4}{e}\right)\right) \\ &< \exp\left(-\ln(1/\delta)\right) = \delta \end{aligned}$$

using the fact that $\ln(4/e) \approx 0.38 > 1/3$.

A.3 Proof of (6)

The following is a straightforward consequence of the Berry-Esseen inequality.

Lemma 20 (see DasGupta, 2008, Theorem 11.1) *Under the assumptions of Section 2 with each $\mathbb{P}[U_i = 1] = 1/2$, let:*

$$\begin{aligned} &T_i = 2(U_i - 1/2), \\ &T = \sqrt{\frac{1}{\ell}}\sum_{i=1}^{\ell} T_i, \text{ and} \\ &Z \text{ be a standard normal random variable.} \end{aligned}$$

Then for all η, we have $\left|\mathbb{P}[T > \eta] - \mathbb{P}[Z > \eta]\right| \le \dfrac{1}{\sqrt{\ell}}$.

Lemma 21 (Feller, 1968, Chapter VII, section 1) *If Z is a standard normal random variable and $x > 0$, then*

$$\frac{1}{\sqrt{2\pi}}\left(\frac{1}{x} - \frac{1}{x^3}\right)e^{-x^2/2} < \mathbb{P}[Z > x] < \frac{1}{\sqrt{2\pi}}\left(\frac{1}{x}\right)e^{-x^2/2}.$$

Now, to prove (6), let $M = \frac{1}{\ell}\sum_{i=1}^{\ell}(U_i - \frac{1}{2})$ and let Z be a standard normal random variable. Then Lemma 20 implies that, for all κ

$$\left|\mathbb{P}\left[2\sqrt{\ell}M > \kappa\right] - \mathbb{P}[Z > \kappa]\right| \leq \frac{1}{\sqrt{\ell}}.$$

Using $\kappa = 2\eta\sqrt{\ell}$,

$$\mathbb{P}[M > \eta] \geq \mathbb{P}\left[Z > 2\eta\sqrt{\ell}\right] - \frac{1}{\sqrt{\ell}}. \tag{23}$$

Applying Lemma 21, we get

$$\mathbb{P}\left[Z > 2\eta\sqrt{\ell}\right] \;\geq\; \frac{1}{\sqrt{2\pi}}\left(\frac{1}{2\eta\sqrt{\ell}} - \left(\frac{1}{2\eta\sqrt{\ell}}\right)^3\right)e^{-2\eta^2\ell}.$$

Since $\ell \geq 1/\eta^2$, we get

$$\begin{aligned}\mathbb{P}\left[Z > 2\eta\sqrt{\ell}\right] &\geq \frac{1}{\sqrt{2\pi}}\left(\frac{1}{2} - \frac{1}{8}\right)\frac{1}{\eta\sqrt{\ell}}e^{-2\eta^2\ell} \\ &\geq \frac{1}{7\eta\sqrt{\ell}}e^{-2\eta^2\ell}.\end{aligned}$$

Combining with (23) completes the proof of (6). □

A.4 Proof of (7)

We follow the proof of Proposition 7.3.2 in Matoušek and Vondrak (2011, Page 46).

Lemma 22 *For n even, let $U_1, \ldots, U_n$ be i.i.d. RVs with $\mathbb{P}[U_1 = 0] = \mathbb{P}[U_1 = 1] = 1/2$ and $U = \sum_{i=1}^{n}$. Then for integer $t \in [0, \frac{n}{8}]$,*

$$\mathbb{P}\left[U \geq \frac{n}{2} + t\right] \geq \frac{1}{5}e^{-16t^2/n}.$$

Proof Let integer $m = n/2$.

$$\begin{aligned}\mathbb{P}[U \geq m + t] &= 2^{-2m}\sum_{j=t}^{m}\binom{2m}{m+j} \\ &\geq 2^{-2m}\sum_{j=t}^{2t-1}\binom{2m}{m+j} \\ &= 2^{-2m}\sum_{j=t}^{2t-1}\binom{2m}{m}\cdot\frac{m}{m+j}\cdot\frac{m-1}{m+j-1}\cdots\frac{m-j+1}{m+1} \\ &\geq \frac{1}{2\sqrt{m}}\sum_{j=t}^{2t-1}\prod_{i=1}^{j}\left(1 - \frac{j}{m+1}\right) \qquad \text{using } \tbinom{2m}{m} \geq 2^{2m}/2\sqrt{m} \\ &\geq \frac{t}{2\sqrt{m}}\left(1 - \frac{2t}{m}\right)^{2t} \\ &\geq \frac{t}{2\sqrt{m}}e^{-8t^2/m} \qquad \text{since } 1 - x \geq e^{-2x} \text{ for } 0 \leq x \leq 1/2.\end{aligned}$$

For $t \geq \frac{1}{2}\sqrt{m}$, the last expression is at least $\frac{1}{4}e^{-16t^2/n}$.

Note that $\mathbb{P}[U = m] = 2^{-2m}\binom{2m}{m} \leq 1/\sqrt{\pi m}$. Thus for $0 \leq t < \frac{1}{2}\sqrt{m}$, we have

$$\begin{aligned}
\mathbb{P}[U \geq m+t] &\geq \frac{1}{2} - t\,\mathbb{P}[U = m] \\
&\geq \frac{1}{2} - \frac{1}{2}\sqrt{m}\frac{1}{\sqrt{\pi m}} \\
&\geq \frac{1}{2} - \frac{1}{2\sqrt{\pi}} \approx 0.218 \geq \frac{1}{5} \geq \frac{1}{5}e^{-16^2/n}.
\end{aligned}$$

Thus the bound $\frac{1}{5}e^{-16t^2/n}$ holds for all $0 \leq t \leq m/4$. ■

A.5 Proof of (8)

The proof of (8) follows the proof of Lemma 5.1 in Anthony and Bartlett (1999). It uses the next two lemmas.

Lemma 23 (Slud's Inequality 1977) *Let B be a binomial (ℓ, p) random variable with $p \leq 1/2$. Then for $\ell(1-p) \geq j \geq \ell p$,*

$$\mathbb{P}[B \geq j] \geq \mathbb{P}\left[Z \geq \frac{j - \ell p}{\sqrt{\ell p(1-p)}}\right]$$

where Z is a standard normal random variable.

Lemma 24 (see Anthony and Bartlett, 1999, Appendix 1) *If Z is a standard normal and $x > 0$ then*

$$\mathbb{P}[Z \geq x] \geq \frac{1}{2}\left(1 - \sqrt{1 - e^{-x^2}}\right).$$

Recall that in (8) U the sum of the ℓ i.i.d. boolean random variables, each of which is 1 with probability $\frac{1}{2} + \eta$. Let B be a random variable with the binomial $(\ell, \frac{1}{2} - \eta)$ distribution.

$$\begin{aligned}
\mathbb{P}\left[\frac{1}{\ell}U < 1/2\right] &= \mathbb{P}[B \geq \ell/2] \\
&\geq \mathbb{P}\left[N \geq \frac{\ell/2 - \ell(1/2-\eta)}{\sqrt{\ell(1/2+\eta)(1/2-\eta)}}\right] && \text{Slud's Inequality} \\
&= \mathbb{P}\left[N \geq \frac{2\eta\sqrt{\ell}}{\sqrt{(1-4\eta^2)}}\right] \\
&\geq \frac{1}{2}\left(1 - \sqrt{1 - \exp\left(-\frac{4\eta^2\ell}{1-4\eta^2}\right)}\right) \\
&\geq \frac{1}{4}\exp\left(-\frac{4\eta^2\ell}{1-4\eta^2}\right) && \text{since } 1 - \sqrt{1-x} > x/2 \\
&\geq \frac{1}{4}\exp\left(-5\eta^2\ell\right) && \text{when } \eta \leq 1/5
\end{aligned}$$

completing the proof of (8).

References

F. Abramovich, Y. Benjamini, D. Donoho, and I. M. Johnstone. Adapting to unknown sparsity by controlling the false discovery rate. *Annals of Statistics*, 34:584–653, 2006.

L. Addario-Berry, N. Broutin, L. Devroye, and G. Lugosi. On combinatorial testing problems. *Annals of Statistics*, 38(5):3063–3092, 2010.

N. Alon, J. H. Spencer, and P. Erdös. *The Probabilistic Method*. Wiley, New York, 1992.

D. Angluin and L. Valiant. Fast probabilistic algorithms for Hamiltonion circuits and matchings. *J. Comp. Sys. Sci.*, 18(2):155–193, 1979.

M. Anthony and P. L. Bartlett. *Neural Network Learning: Theoretical Foundations*. Cambridge University Press, Cambridge, 1999.

P. Bickel and E. Levina. Some theory of Fisher's linear discriminant function, 'Naive Bayes', and some alternatives when there are many more variables than observations. *Bernoulli*, 10(6):989–1010, 2004.

C. M. Bishop. *Pattern Recognition and Machine Learning*. Springer, Berlin, 2006.

L. Breiman. Arcing classifiers. *Annals of Statistics*, 26(3):801–824, 1998.

P. Bühlmann and B. Yu. Analyzing bagging. *Annals of Statistics*, 30:927–961, 2002.

A. DasGupta. *Asymptotic Theory of Statistics and Probability*. Springer, Berlin, 2008.

S. Dasgupta and P. M. Long. Boosting with diverse base classifiers. In *COLT*, pages 273–287, Washington, D.C., 2003.

D. Donoho and J. Jin. Higher criticism for detecting sparse heterogeneous mixtures. *Annals of Statistics*, 32:952–994, 2004.

D. Donoho and J. Jin. Asymptotic minimaxity of false discovery rate thresholding for sparse exponential data. *Annals of Statistics*, 34, 2006.

D. Donoho and J. Jin. Higher criticism thresholding: optimal feature selection when useful features and rare and weak. *PNAS*, 105(39):14790–14795, 2008.

R. O. Duda, P. E. Hart, and D. G. Stork. *Pattern Classification (2nd ed.)*. Wiley, New York, 2000.

A. Ehrenfeucht, D. Haussler, M. Kearns, and L. G. Valiant. A general lower bound on the number of examples needed for learning. *Information and Computation*, 82(3):247–251, 1989.

J. Fan and Y. Fan. High dimensional classification using features annealed independence rules. *Annals of Statistics*, 36(6):2605–2637, 2008.

W. Feller. *An Introduction to Probability Theory and Its Applications*. Wiley, New York, 1968.

J. Friedman, T. Hastie, and R. Tibshirani. Additive logistic regression: A statistical view of boosting. *Annals of Statistics*, 38(2):337–407, 2000.

D. Haussler, N. Littlestone, and M. K. Warmuth. Predicting $\{0,1\}$-functions on randomly drawn points. *Information and Computation*, 115(2):129–161, 1994.

D. P. Helmbold and P. M. Long. On the necessity of irrelevant variables. In *ICML*, pages 281–288, Bellevue, WA, 2011.

J. Jin. Impossibility of successful classication when useful features are rare and weak. *PNAS*, 106 (22):8859–8864, 2009.

V. Koltchinskii and D. Panchenko. Empirical margin distributions and bounding the generalization error of combined classifiers. *Annals of Statistics*, 30(1), 2002.

J. Matoušek and J. Vondrak. The probabilistic method, 2011. Lecture notes.

N. Meinshausen and J. Rice. Estimating the proportion of false null hypotheses among a large number of independently tested hypotheses. *Annals of Statistics*, 34(1):373–393, 2006.

R. Motwani and P. Raghavan. *Randomized Algorithms*. Cambridge University Press, Cambridge, 1995.

A. Y. Ng and M. I. Jordan. On discriminative vs. generative classifiers: A comparison of logistic regression and naive bayes. In *NIPS*, pages 841–848, Vancouver, B.C., 2001.

S. Pemmaraju. Equitable coloring extends Chernoff-Hoeffding bounds. In *RANDOM*, pages 285–296, Berkeley, CA, 2001.

D. Pollard. *Convergence of Stochastic Processes*. Springer Verlag, Berlin, 1984.

J. Quinlan. Bagging, boosting and C4.5. In *AAAI*, pages 725–730, Portland, OR, 1996.

R. E. Schapire, Y. Freund, P. L. Bartlett, and W. S. Lee. Boosting the margin: A new explanation for the effectiveness of voting methods. *Annals of Statistics*, 26(5):1651–1686, 1998.

E. Slud. Distribution inequalities for the binomial law. *Annals of Probability*, 5:404–412, 1977.

R. Tibshirani, T. Hastie, B. Narasimhan, and G. Chu. Diagnosis of multiple cancer types by shrunken centroids of gene expression. *PNAS*, 99(10):6567–72, 2002.

V. N. Vapnik. *Statistical Learning Theory*. Wiley, New York, 1998.

L. Wang, M. Sugiyama, C. Yang, Z. Zhou, and J. Feng. On the margin explanation of boosting algorithms. In *COLT*, pages 479–490, Helsinki, Finland, 2008.

Journal of Machine Learning Research 13 (2012) 2171-2175 Submitted 7/11; Revised 4/12; Published 7/12

DEAP: Evolutionary Algorithms Made Easy

Félix-Antoine Fortin FELIX-ANTOINE.FORTIN.1@ULAVAL.CA
François-Michel De Rainville FRANCOIS-MICHEL.DE-RAINVILLE.1@ULAVAL.CA
Marc-André Gardner MARC-ANDRE.GARDNER.1@ULAVAL.CA
Marc Parizeau MARC.PARIZEAU@GEL.ULAVAL.CA
Christian Gagné CHRISTIAN.GAGNE@GEL.ULAVAL.CA
Laboratoire de vision et systèmes numériques
Département de génie électrique et de génie informatique
Université Laval
Québec (Québec), Canada G1V 0A6

Editor: Mikio Braun

Abstract

DEAP is a novel evolutionary computation framework for rapid prototyping and testing of ideas. Its design departs from most other existing frameworks in that it seeks to make algorithms explicit and data structures transparent, as opposed to the more common black-box frameworks. Freely available with extensive documentation at `http://deap.gel.ulaval.ca`, DEAP is an open source project under an LGPL license.

Keywords: distributed evolutionary algorithms, software tools

1. Introduction

Evolutionary Computation (EC) is a sophisticated field with very diverse techniques and mechanisms, where even well designed frameworks can become quite complicated under the hood. They thus strive to hide the implementation details as much as possible, by providing large libraries of high-level functionalities, often in many different flavours. This is the black-box software model (Roberts and Johnson, 1997). The more elaborate these boxes become, the more obscure they are, and the less likely the commoner is to ever take a peek under the hood to consider making changes. But using EC to solve real-world problems most often requires the customization of algorithms.

The DEAP (Distributed Evolutionary Algorithms in Python) framework is built over the Python programming language that provides the essential glue for assembling sophisticated EC systems. Its aim is to provide practical tools for rapid prototyping of custom evolutionary algorithms, where every step of the process is as explicit (pseudocode like) and easy to read and understand as possible. It also places a high value on both code compactness and code clarity.

2. Core Architecture

DEAP's core is composed of two simple structures: a creator and a toolbox. The *creator* module is a meta-factory that allows the run-time creation of classes via both inheritance and composition. Attributes, both data and functions, can be dynamically added to existing classes in order to cre-

ate new types empowered with user-specific EC functionalities. Practically speaking, this allows the creation of genotypes and populations from any data structure such as lists, sets, dictionaries, trees, etc. This creator concept is key to facilitating the implementation of any type of EA, including genetic algorithms (Mitchell, 1998), genetic programming (Banzhaf et al., 1998), evolution strategies (Beyer and Schwefel, 2002), covariance matrix adaptation evolution strategy (Hansen and Ostermeier, 2001), particle swarm optimization (Kennedy and Eberhart, 2001), and many more.

The *toolbox* is a container for the tools (operators) that the user wants to use in his EA. The toolbox is manually populated by the user with selected tools. For instance, if the user needs a crossover in his algorithm, but has access to several crossover types, he will choose the one best suited for his current problem, for example a uniform crossover "cxUniform", and register it into the toolbox using a generic "mate" alias. This way, he is able to build algorithms that are decoupled from operator sets. If he later decides that some other crossover is better suited, his algorithm will remain unchanged, he will only need to update the corresponding alias in the toolbox.

The core functionalities of DEAP are levered by several peripheral modules. The *algorithms* module contains four classical EC algorithms: generational, $(\mu\ ,\ \lambda)$, $(\mu+\lambda)$, and ask-and-tell (Collette et al., 2010). These serve as a starting point for users to build their own custom EAs meeting their specific needs. The *tools* module provides basic EC operators such as initializations, mutations, crossovers, and selections. These operators can be directly added to a toolbox in order to be used in algorithms. This module also contains a number of components that gather useful information regarding the evolution: fitness statistics, genealogy, hall-of-fame for the best individuals encountered so far, and checkpointing mechanisms to allow evolution restart. A *base* module contains some data structures frequently used in EC, but not implemented in standard Python (e.g., generic fitness object). The last module named *dtm*, for Distributed Task Manager, offers distributed substitutes for common Python functions such as `apply` and `map`. DTM allows distribution of specific parts of users' algorithms by taking care of spawning and distributing sub-tasks across a cluster of computers. It even balances the workload among workers to optimize the distribution efficiency.

To illustrate DEAP usage, we now present an example for **multi-objective feature selection**. The individual is represented as a bit-string where each bit corresponds to a feature that can be selected or not. The objective is to maximize the number of well-classified test cases and to minimize the number of features used. Figure 1 shows how this problem can be solved using DEAP.

On line 2, the relevant DEAP modules are first imported. On line 3, a multi-objective fitness class `FitnessMulti` is created. The first argument of the `creator.create` method defines the name of the derived class, while the second argument specifies the inherited base class (in this case `base.Fitness`). The third argument adds a new class attribute called `weights`, initialized with a tuple that specifies a two-objective fitness, of which the first must be maximized (1.0), and the second must be minimized (-1.0). Next, an `Individual` class is derived from the Python list and composed with our newly created `FitnessMulti` object. After defining a proper evaluation function that returns the fitness values (classification rate, number of selected features) (lines 6 and 7), a toolbox object is created on line 9, and populated on lines 10 through 16 with aliases in order to initialize individuals and population, and specify the variation operators (`mate`, `mutate`, and `select`) and the fitness evaluation function (`evaluate`) used by the evolutionary loop. The toolbox `register` method accepts a variable number of arguments. The first is the alias name, and the second the function that we want to associate with this alias. All other arguments are passed to this function when the alias is called. For instance, a call to `toolbox.bit` will in fact call `random.randint` with arguments 0 and 1, and thus generate a random bit. On line 11, to initialize an individual assuming a 80 features selec-

```
1  import knn, random
2  from deap import creator, base, tools, algorithms
3  creator.create("FitnessMulti", base.Fitness, weights=(1.0, -1.0))
4  creator.create("Individual", list, fitness=creator.FitnessMulti)
5
6  def evalFitness(individual):
7      return knn.classification_rate(features=individual), sum(individual)
8
9  toolbox = base.Toolbox()
10 toolbox.register("bit", random.randint, 0, 1)
11 toolbox.register("individual", tools.initRepeat, creator.Individual, toolbox.bit, n=80)
12 toolbox.register("population", tools.initRepeat, list, toolbox.individual, n=100)
13 toolbox.register("evaluate", evalFitness)
14 toolbox.register("mate", tools.cxUniform, indpb=0.1)
15 toolbox.register("mutate", tools.mutFlipBit, indpb=0.05)
16 toolbox.register("select", tools.selNSGA2)
17
18 population = toolbox.population()
19 fits = toolbox.map(toolbox.evaluate, population)
20 for fit, ind in zip(fits, population):
21     ind.fitness.values = fit
22
23 for gen in range(50):
24     offspring = algorithms.varOr(population, toolbox, lambda_=100, cxpb=0.5, mutpb=0.1)
25     fits = toolbox.map(toolbox.evaluate, offspring)
26     for fit, ind in zip(fits, offspring):
27         ind.fitness.values = fit
28     population = toolbox.select(offspring + population, k=100)
```

Figure 1: Multi-objective feature selection example with NSGA-II (Deb et al., 2002).

tion problem, this bit function is simply called $n = 80$ times repeatedly using the `tools.initRepeat` method that accepts three arguments: a container, a function, and the number of times to repeat initialization. Similarly, a `population` alias is defined to allow population initialization, in this case with $n = 100$ individuals. Line 18 then proceeds with the allocation of a population of individuals that have their fitness evaluated on line 19 by mapping the evaluation function to every element of the population container. Lines 20 and 21 replace the individuals fitness with their newly computed values. Finally, lines 23 through 28 show the evolutionary loop that uses the $\mu + \lambda$ strategy, where $\mu = 100$ parents (current population) are mixed with $\lambda = 100$ offspring (`lambda_` line 24) for the selection process (NSGA-II) to produce the next generation of $k = 100$ parents (line 28). The `varOr` algorithm loops until it has generated λ offspring using either crossover (`mate` alias) with probability `cxpb`, mutation (`mutate` alias) with probability `mutpb`, or reproduction.

Efficient **distribution** of specific parts of user algorithms is made possible by the `dtm` module. For instance, in the previous examples, the only required changes in order to distribute the fitness evaluation task on a cluster of computers are to import the `dtm` module and to replace the toolbox default `map` function by its distributed `dtm` equivalent:

```
from deap import dtm
toolbox.register("map", dtm.map)
```

The map function calls (lines 19 and 25 in Figure 1) spawn sub-tasks of the evaluation function. The task manager distributes these sub-tasks across the worker nodes, and automatically balances the workload evenly among them.

Framework	Type	Configuration	Algorithm	Example	Total
ECJ	202	35	65	26	328
EO	43	n/a	67	68	178
Open BEAGLE	256	41	116	64	477
Pyevolve	42	n/a	336	25	378
inspyred	23	n/a	143	24	190
DEAP	n/a	n/a	n/a	59	59

Table 1: Comparison of the number of lines needed to define different components of a OneMax example with some major frameworks, as counted by cloc (`http://cloc.sf.net`).

Table 1 presents a comparison of the number of lines required by some of the most popular frameworks to define different components of a OneMax example (without distribution). Columns respectively indicate how many lines are required to define the *bit-string* type, to configure the operators, to implement the generational algorithm and to execute the example. DEAP is the only framework that allows the complete definition of the EA in less than one hundred lines of code. Even though some frameworks may allow shorter expressions of standardized solutions, any needed customization of type or algorithm can also force the user to dig deep into the framework to modify perhaps hundreds of lines. We therefore assert that DEAP is superior to previous frameworks for rapid prototyping of new algorithms and definition of custom types.

3. Conclusion

Current major EC frameworks generally do a good job of offering generic tools to solve hard problems using EAs. However, their implementation intricacies can also make them difficult to extend for the commoner. Even experts can become overwhelmed when trying to implement specific features. This paper introduced the DEAP framework that combines the flexibility and power of the Python programming language with a clean and lean core of transparent EC components that both facilitate rapid prototyping and testing of new EA ideas, and encourage creativeness through simplicity and explicit algorithms. The DEAP core is implemented in 6 program files (Python modules) with only 1653 lines of code. To these core lines, the DTM module adds another 2073 lines for enabling the easy distribution of computationally intensive algorithm parts. The framework also includes more than 35 application examples that add another 2448 lines of code, leading to an average ratio of core lines to example lines of 1.5, compared with 2.4 for inspyred, 3.4 for ECJ, 5.4 for Pyevolve, 5.5 for EO, and 16.3 for OpenBEAGLE. In other words, DEAP implements the 15 examples of OpenBEAGLE plus 20 more with 10 times less lines of code.

Acknowledgments

This work has been supported by the Fonds de recherche du Québec - Nature et technologies (FQRNT) and by the Natural Sciences and Engineering Research Council of Canada (NSERC).

References

W. Banzhaf, P. Nordin, R. E. Keller, and F. D. Francone. *Genetic Programming: An Introduction*. Morgan Kaufmann, 1998.

H.-G. Beyer and H.-P. Schwefel. Evolution strategies: A comprehensive introduction. *Natural Computing*, 1:3–52, 2002.

Y. Collette, N. Hansen, G. Pujol, D. Salazar Aponte, and R. Le Riche. On object-oriented programming of optimizers – Examples in Scilab. In P. Breitkopf and R. F. Coelho, editors, *Multidisciplinary Design Optimization in Computational Mechanics*, chapter 14, pages 527–565. Wiley, 2010.

K. Deb, A. Pratab, S. Agarwal, and T. Meyarivan. A fast elitist non-dominated sorting genetic algorithm for multi-objective optimization: NSGA-II. *IEEE Transactions on Evolutionary Computation*, 6:849–858, April 2002.

N. Hansen and A. Ostermeier. Completely derandomized self-adaptation in evolution strategies. *Evolutionary Computation*, 9(2):159–195, 2001.

J. Kennedy and R. C. Eberhart. *Swarm Intelligence*. Morgan Kaufmann, 2001.

M. Mitchell. *An Introduction to Genetic Algorithms*. MIT Press, 1998.

D. Roberts and R. E. Johnson. Evolving frameworks: A pattern language for developing object-oriented frameworks. In *Pattern Languages of Program Design 3*. Addison Wesley, 1997.

Journal of Machine Learning Research 13 (2012) 2177-2204 Submitted 2/11; Revised 9/11; Published 7/12

An Introduction to Artificial Prediction Markets for Classification

Adrian Barbu ABARBU@FSU.EDU
Department of Statistics
Florida State University
Tallahassee, FL 32306, USA

Nathan Lay NLAY@FSU.EDU
Department of Scientific Computing
Florida State University
Tallahassee, FL 32306, USA

Editor: Shie Mannor

Abstract

Prediction markets are used in real life to predict outcomes of interest such as presidential elections. This paper presents a mathematical theory of artificial prediction markets for supervised learning of conditional probability estimators. The artificial prediction market is a novel method for fusing the prediction information of features or trained classifiers, where the fusion result is the contract price on the possible outcomes. The market can be trained online by updating the participants' budgets using training examples. Inspired by the real prediction markets, the equations that govern the market are derived from simple and reasonable assumptions. Efficient numerical algorithms are presented for solving these equations. The obtained artificial prediction market is shown to be a maximum likelihood estimator. It generalizes linear aggregation, existent in boosting and random forest, as well as logistic regression and some kernel methods. Furthermore, the market mechanism allows the aggregation of specialized classifiers that participate only on specific instances. Experimental comparisons show that the artificial prediction markets often outperform random forest and implicit online learning on synthetic data and real UCI data sets. Moreover, an extensive evaluation for pelvic and abdominal lymph node detection in CT data shows that the prediction market improves adaboost's detection rate from 79.6% to 81.2% at 3 false positives/volume.

Keywords: online learning, ensemble methods, supervised learning, random forest, implicit online learning

1. Introduction

Prediction markets, also known as information markets, are forums that trade contracts that yield payments dependent on the outcome of future events of interest. They have been used in the US Department of Defense (Polk et al., 2003), health care (Polgreen et al., 2006), to predict presidential elections (Wolfers and Zitzewitz, 2004) and in large corporations to make informed decisions (Cowgill et al., 2008). The prices of the contracts traded in these markets are good approximations for the probability of the outcome of interest (Manski, 2006; Gjerstad and Hall, 2005). prediction markets are capable of fusing the information that the market participants possess through the contract price. For more details, see Arrow et al. (2008).

In this paper we introduce a mathematical theory for simulating prediction markets numerically for the purpose of supervised learning of probability estimators. We derive the mathematical equa-

tions that govern the market and show how can they be solved numerically or in some cases even analytically. An important part of the prediction market is the contract price, which will be shown to be an estimator of the class-conditional probability given the evidence presented through a feature vector $\mathbf{x}$. It is the result of the fusion of the information possessed by the market participants.

The obtained artificial prediction market turns out to have good modeling power. It will be shown in Section 3.1 that it generalizes linear aggregation of classifiers, the basis of boosting (Friedman et al., 2000; Schapire, 2003) and random forest (Breiman, 2001). It turns out that to obtain linear aggregation, each market participant purchases contracts for the class it predicts, regardless of the market price for that contract. Furthermore, in Sections 3.2 and 3.3 will be presented special *betting functions* that make the prediction market equivalent to a logistic regression and a kernel-based classifier respectively.

We introduce a new type of classifier that is *specialized* in modeling certain regions of the feature space. Such classifiers have good accuracy in their region of specialization and are not used in predicting outcomes for observations outside this region. This means that for each observation, a different subset of classifiers will be aggregated to obtain the estimated probability, making the whole approach become a sort of *ad-hoc* aggregation. This is contrast to the general trend in boosting where the same classifiers are aggregated for all observations.

We give examples of generic specialized classifiers as the leaves of random trees from a random forest. Experimental validation on thousands of synthetic data sets with Bayes errors ranging from 0 (very easy) to 0.5 (very difficult) as well as on real UCI data show that the prediction market using the specialized classifiers outperforms the random forest in prediction and in estimating the true underlying probability.

Moreover, we present experimental comparisons on many UCI data sets of the artificial prediction market with the recently introduced implicit online learning (Kulis and Bartlett, 2010) and observe that the market significantly outperforms the implicit online learning on some of the data sets and is never outperformed by it.

2. The Artificial Prediction Market for Classification

This work simulates the *Iowa electronic market* (Wolfers and Zitzewitz, 2004), which is a real prediction market that can be found online at `http://www.biz.uiowa.edu/iem/`.

2.1 The Iowa Electronic Market

The *Iowa electronic market* (Wolfers and Zitzewitz, 2004) is a forum where contracts for future outcomes of interest (e.g., presidential elections) are traded.

Contracts are sold for each of the possible outcomes of the event of interest. The contract price fluctuates based on supply and demand. In the Iowa electronic market, a winning contract (that predicted the correct outcome) pays \$1 after the outcome is known. Therefore, the contract price will always be between 0 and 1.

Our market will simulate this behavior, with contracts for all the possible outcomes, paying 1 if that outcome is realized.

2.2 Setup of the Artificial Prediction Market

If the possible classes (outcomes) are $1,...,K$, we assume there exist contracts for each class, whose prices form a K-dimensional vector $\mathbf{c} = (c_1,...,c_K) \in \Delta \subset [0,1]^K$, where Δ is the probability simplex $\Delta = \{\mathbf{c} \in [0,1]^K, \sum_{k=1}^K c_k = 1\}$.

Let $\Omega \subset \mathbb{R}^F$ be the instance or feature space containing all the available information that can be used in making outcome predictions $p(Y = k|\mathbf{x}), \mathbf{x} \in \Omega$.

The market consists of a number of market participants $(\beta_m, \phi_m(\mathbf{x},\mathbf{c})), m = 1,...,M$.

A *market participant* is a pair $(\beta, \phi(\mathbf{x},\mathbf{c}))$ of a *budget* β and a *betting function* $\phi(\mathbf{x},\mathbf{c}) : \Omega \times \Delta \to [0,1]^K, \phi(\mathbf{x},\mathbf{c}) = \left(\phi^1(\mathbf{x},\mathbf{c}),...,\phi^K(\mathbf{x},\mathbf{c})\right)$. The budget β represents the weight or importance of the participant in the market. The betting function tells what percentage of its budget this participant will allocate to purchase contracts for each class, based on the instance $\mathbf{x} \in \Omega$ and the market price $\mathbf{c}$. As the market price $\mathbf{c}$ is not known in advance, the betting function describes what the participant plans to do for each possible price $\mathbf{c}$. The betting functions could be based on trained classifiers $h(\mathbf{x}) : \Omega \to \Delta, h(\mathbf{x}) = (h^1(\mathbf{x}),...,h^K(\mathbf{x})), \sum_{k=1}^K h^k(\mathbf{x}) = 1$, but they can also be related to the feature space in other ways. We will show that logistic regression and kernel methods can also be represented using the artificial prediction market and specific types of betting functions. In order to bet at most the budget β, the betting functions must satisfy $\sum_{k=1}^K \phi^k(\mathbf{x},\mathbf{c})) \leq 1$.

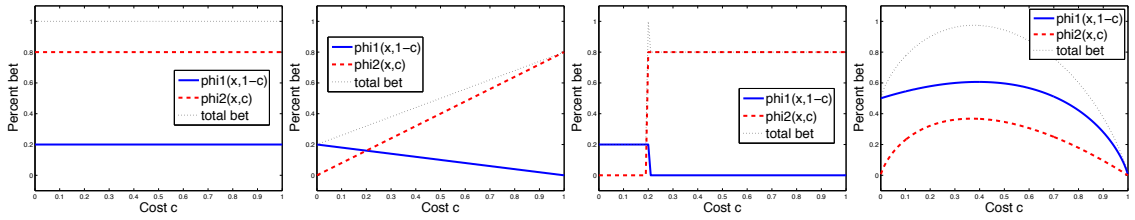


Figure 1: Betting function examples: a) Constant, b) Linear, c) Aggressive, d) Logistic. Shown are $\phi^1(\mathbf{x}, 1-c)$ (red), $\phi^2(\mathbf{x},c)$ (blue), and the total amount bet $\phi^1(\mathbf{x}, 1-c) + \phi^2(\mathbf{x},c)$ (black dotted). For a) through c), the classifier probability is $h^2(\mathbf{x}) = 0.2$.

Examples of betting functions include the following, also shown in Figure 1:

- Constant betting functions
$$\phi^k(\mathbf{x},\mathbf{c}) = \phi^k(\mathbf{x})$$
for example based on trained classifiers $\phi^k(\mathbf{x},\mathbf{c}) = \eta h^k(\mathbf{x})$, where $\eta \in (0,1]$ is constant.

- Linear betting functions
$$\phi^k(\mathbf{x},\mathbf{c}) = (1 - c_k)h^k(\mathbf{x}). \tag{1}$$

- Aggressive betting functions
$$\phi^k(\mathbf{x},\mathbf{c}) = h^k(\mathbf{x}) \begin{cases} 1 & \text{if } c_k \leq h^k(\mathbf{x}) \\ 0 & \text{if } c_k > h^k(\mathbf{x}) + \varepsilon \\ \frac{h^k(\mathbf{x}) + \varepsilon - c_k}{\varepsilon} & \text{otherwise} \end{cases}. \tag{2}$$

- Logistic betting functions:

$$\phi_m^1(\mathbf{x}, 1-c) = (1-c)(x_m^+ - \ln(1-c)/B),$$
$$\phi_m^2(\mathbf{x}, c) = c(-x_m^- - \ln c/B)$$

where $x^+ = xI(x>0), x^- = xI(x<0)$ and $B = \sum_m \beta_m$.

The betting functions play a similar role to the *potential functions* from maximum entropy models (Berger et al., 1996; Ratnaparkhi et al., 1996; Zhu et al., 1998), in that they make a conversion from the feature output (or classifier output for some markets) to a common unit of measure (energy for the maximum entropy models and money for the market).

The contract price does not fluctuate in our setup, instead it is governed by Equation (4). This equation guarantees that at this price, the total amount obtained from selling contracts to the participants is equal to the total amount won by the winning contracts, independent of the outcome.

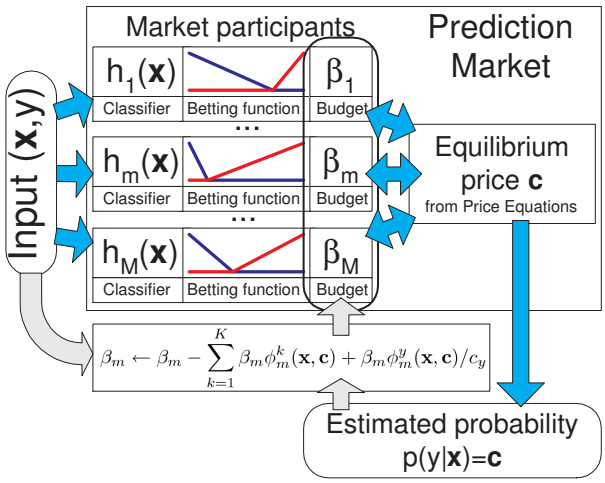


Figure 2: Online learning and aggregation using the artificial prediction market. Given feature vector $\mathbf{x}$, a set of market participants will establish the market equilibrium price $\mathbf{c}$, which is an estimator of $P(Y=k|\mathbf{x})$. The equilibrium price is governed by the Price Equations (4). Online training on an example $(\mathbf{x}, y)$ is achieved through **Budget Update** $(\mathbf{x}, y, \mathbf{c})$ shown with gray arrows.

Algorithm 1 Budget Update $(\mathbf{x}, y, \mathbf{c})$

Input: Training example $(\mathbf{x}, y)$, price $\mathbf{c}$

for $m = 1$ **to** M **do**

Update participant m's budget as

$$\beta_m \leftarrow \beta_m - \sum_{k=1}^{K} \beta_m \phi_m^k(\mathbf{x}, \mathbf{c}) + \frac{\beta_m}{c_y} \phi_m^y(\mathbf{x}, \mathbf{c}) \tag{3}$$

end for

2.3 Training the Artificial Prediction Market

Training the market involves initializing all participants with the same budget β_0 and presenting to the market a set of training examples $(\mathbf{x}_i, y_i), i = 1, ..., N$. For each example $(\mathbf{x}_i, y_i)$ the participants purchase contracts for the different classes based on the market price $\mathbf{c}$ (which is not known yet) and their budgets β_m are updated based on the contracts purchased and the true outcome y_i. After all training examples have been presented, the participants will have budgets that depend on how well they predicted the correct class y for each training example $\mathbf{x}$. This procedure is illustrated in Figure 2.

Algorithm 2 Prediction Market Training

Input: Training examples $(\mathbf{x_i}, y_i), i = 1, ..., N$
Initialize all budgets $\beta_m = \beta_0, m = 1, ..., M$.
for each training example $(\mathbf{x}_i, y_i)$ **do**
  Compute equilibrium price $\mathbf{c}_i$ using Equation 4
  Run **Budget Update** $(\mathbf{x}_i, y_i, \mathbf{c}_i)$
end for

The budget update procedure subtracts from the budget of each participant the amounts it bets for each class, then rewards each participant based on how many contracts it purchased for the correct class.

Participant m purchased $\beta_m \phi_m^k(\mathbf{x}, \mathbf{c})$ worth of contracts for class k, at price c_k. Thus the number of contracts purchased for class k is $\beta_m \phi_m^k(\mathbf{x}, \mathbf{c})/c_k$. Totally, participant m's budget is decreased by the amount $\sum_{k=1}^K \beta_m \phi_m^k(\mathbf{x}, \mathbf{c})$ invested in contracts. Since participant m bought $\beta_m \phi_m^y(\mathbf{x}, \mathbf{c})/c_y$ contracts for the correct class y, he is rewarded the amount $\beta_m \phi_m^y(\mathbf{x}, \mathbf{c})/c_y$.

2.4 The Market Price Equations

Since we are simulating a real market, we assume that the total amount of money collectively owned by the participants is conserved after each training example is presented. Thus the sum of all participants' budgets $\sum_{m=1}^M \beta_m$ should always be $M\beta_0$, the amount given at the beginning. Since any of the outcomes is theoretically possible for each instance, we have the following constraint:

Assumption 1 *The total budget $\sum_{m=1}^M \beta_m$ must be conserved independent of the outcome y.*

This condition transforms into a set of equations that constrain the market price, which we call the price equations. The market price $\mathbf{c}$ also obeys $\sum_{k=1}^K c_k = 1$.

Let $B(\mathbf{x}, \mathbf{c}) = \sum_{m=1}^M \sum_{k=1}^K \beta_m \phi_m^k(\mathbf{x}, \mathbf{c})$ be the total bet for observation $\mathbf{x}$ at price $\mathbf{c}$. We have

Theorem 1 Price Equations. *The total budget $\sum_{m=1}^M \beta_m$ is conserved after the* **Budget Update**$(\mathbf{x}, y, \mathbf{c})$*, independent of the outcome y, if and only if $c_k > 0, k = 1, ..., K$ and*

$$\sum_{m=1}^{M} \beta_m \phi_m^k(\mathbf{x}, \mathbf{c}) = c_k B(\mathbf{x}, \mathbf{c}), \quad \forall k = 1, ..., K. \tag{4}$$

The proof is given in the Appendix.

2.5 Price Uniqueness

The price equations together with the equation $\sum_{k=1}^{K} c_k = 1$ are enough to uniquely determine the market price $\mathbf{c}$, under mild assumptions on the betting functions $\phi^k(\mathbf{x}, \mathbf{c})$.

Observe that if $c_k = 0$ for some k, then the contract costs 0 and pays 1, so there is everything to win. In this case, one should have $\phi^k(\mathbf{x}, \mathbf{c}) > 0$.

This suggests a class of betting functions $\phi^k(\mathbf{x}, c_k)$ depending only on the price c_k that are continuous and monotonically non-increasing in c_k. If all $\phi_m^k(\mathbf{x}, c_k), m = 1, ..., M$ are continuous and monotonically non-increasing in c_k with $\phi_m^k(\mathbf{x}, 0) > 0$ then $f_k(c_k) = \frac{1}{c_k} \sum_{m=1}^{M} \beta_m \phi_m^k(\mathbf{x}, c_k)$ is continuous and strictly decreasing in c_k as long as $f_k(c_k) > 0$.

To obtain conditions for price uniqueness, we use the following function

$$f_k(c_k) = \frac{1}{c_k} \sum_{m=1}^{M} \beta_m \phi_m^k(\mathbf{x}, c_k), k = 1, ..., K.$$

Remark 2 *If all $f_k(c_k)$ are continuous and strictly decreasing in c_k as long as $f_k(c_k) > 0$, then for every $n > 0$, $n \geq n_k = f_k(1)$ there is a unique $c_k = c_k(n)$ that satisfies $f_k(c_k) = n$.*

The proof is given in the Appendix.

To guarantee price uniqueness, we need at least one market participant to satisfy the following

Assumption 2 *The total bet of participant $(\beta_m, \phi_m(\mathbf{x}, \mathbf{c}))$ is positive inside the simplex Δ, that is,*

$$\sum_{j=1}^{K} \phi_m^j(\mathbf{x}, c_j) > 0, \ \forall c \in (0,1)^K, \sum_{j=1}^{K} c_j = 1. \tag{5}$$

Then we have the following result, also proved in the Appendix.

Theorem 3 *Assume all betting functions $\phi_m^k(\mathbf{x}, c_k), m = 1, ..., M, k = 1, ..., K$ are continuous, with $\phi^k(\mathbf{x}, 0) > 0$ and $\phi_m^k(\mathbf{x}, c)/c$ is strictly decreasing in c as long as $\phi_m^k(\mathbf{x}, c) > 0$. If the betting function $\phi_m(\mathbf{x}, \mathbf{c})$ of least one participant with $\beta_m > 0$ satisfies Assumption 2, then for the* **Budget Update**$(\mathbf{x}, y, \mathbf{c})$ *there is a unique price $\mathbf{c} = (c_1, ..., c_K) \in (0,1)^K \cap \Delta$ such that the total budget $\sum_{m=1}^{M} \beta_m$ is conserved.*

Observe that all four betting functions defined in Section 2.2 (constant, linear, aggressive and logistic) satisfy the conditions of Theorem 3, so there is a unique price that conserves the budget.

2.6 Solving the Market Price Equations

In practice, a double bisection algorithm could be used to find the equilibrium price, computing each $c_k(n)$ by the bisection method, and employing another bisection algorithm to find n such that the price condition $\sum_{k=1}^{K} c_k(n) = 1$ holds. Observe that the n satisfying $\sum_{k=1}^{K} c_k(n) = 1$ can be bounded from above by

$$n = n \sum_{k=1}^{K} c_k(n) = \sum_{k=1}^{K} c_k(n) f_k(c_k(n)) = \sum_{k=1}^{K} \sum_{m=1}^{M} \beta_m \phi_m^k(\mathbf{x}, \mathbf{c}) \leq \sum_{m=1}^{M} \beta_m$$

because for each m, $\sum_{k=1}^{K} \phi_m^k(\mathbf{x}, \mathbf{c}) \leq 1$.

A potentially faster alternative to the double bisection method is the Mann Iteration (Mann, 1953) described in Algorithm 3. The price equations can be viewed as fixed point equation $F(\mathbf{c}) = \mathbf{c}$, where $F(\mathbf{c}) = \frac{1}{n}(f_1(\mathbf{c}), ..., f_K(\mathbf{c}))$ with $f_k(\mathbf{c}) = \sum_{m=1}^{m} \beta_m \phi_m^k(\mathbf{x}, c_k)$. The Mann iteration is a fixed point algorithm, which makes weighted update steps

$$\mathbf{c}^{t+1} = (1 - \frac{1}{t})\mathbf{c}^t + \frac{1}{t}F(\mathbf{c}^t).$$

The Mann iteration is guaranteed to converge for contractions or pseudo-contractions. However, we observed experimentally that it usually converges in only a few (up to 10) steps, making it about 100-1000 times faster than the double bisection algorithm. If, after a small number of steps, the Mann iteration has not converged, the double bisection algorithm is used on that instance to compute the equilibrium price. However, this happens on less than 0.1% of the instances.

Algorithm 3 Market Price by Mann Iteration

Initialize $i = 1,\ c_k = \frac{1}{K}, k = 1, ..., K$
repeat
  $f_k = \sum_m \beta_m \phi_m^k(x, \mathbf{c})$
  $n = \sum_k f_k$
  if $n \neq 0$ **then**
    $f_k \leftarrow \frac{f_k}{n}$
    $r_k = f_k - c_k$
    $c_k \leftarrow \frac{(i-1)c_k + f_k}{i}$
  end if
  $i \leftarrow i + 1$
until $\sum_k |r_k| \leq \varepsilon$ or $n = 0$ or $i > i_{\max}$

2.7 Two-class Formulation

For the two-class problem, that is, $K = 2$, the budget equation can be simplified by writing $\mathbf{c} = (1 - c, c)$ and obtaining the *two-class market price equation*

$$(1 - c) \sum_{m=1}^{M} \beta_m \phi_m^2(\mathbf{x}, c) - c \sum_{m=1}^{M} \beta_m \phi_m^1(\mathbf{x}, 1 - c) = 0. \quad (6)$$

This can be solved numerically directly in c using the bisection method. Again, the solution is unique if $\phi_m^k(\mathbf{x}, c_k), m = 1, ..., M, k = 1, 2$ are continuous, monotonically non-increasing and obey condition (5). Moreover, the solution is guaranteed to exist if there exist m, m' with $\beta_m > 0, \beta_{m'} > 0$ and such that $\phi_m^2(\mathbf{x}, 0) > 0, \phi_{m'}^1(\mathbf{x}, 1) > 0$.

3. Relation to Existing Supervised Learning Methods

There is a large degree of flexibility in choosing the betting functions $\phi_m(\mathbf{x}, \mathbf{c})$. Different betting functions give different ways to fuse the market participants. In what follows we prove that by choosing specific betting functions, the artificial prediction market behaves like a linear aggregator or logistic regressor, or that it can be used as a kernel-based classifier.

3.1 Constant Betting and Linear Aggregation

For markets with constant betting functions, $\phi_m^k(\mathbf{x},\mathbf{c}) = \phi_m^k(\mathbf{x})$ the market price has a simple analytic formula, proved in the Appendix.

Theorem 4 Constant Betting. *If all betting function are constant $\phi_m^k(\mathbf{x},\mathbf{c}) = \phi_m^k(\mathbf{x})$, then the equilibrium price is*

$$\mathbf{c} = \frac{\sum_{m=1}^M \beta_m \phi_m(\mathbf{x})}{\sum_{m=1}^M \sum_{k=1}^K \beta_m \phi_m^k(\mathbf{x})}. \tag{7}$$

Furthermore, if the betting functions are based on classifiers $\phi_m^k(\mathbf{x},\mathbf{c}) = \eta h_m^k(\mathbf{x})$ then the equilibrium price is obtained by linear aggregation

$$\mathbf{c} = \frac{\sum_{m=1}^M \beta_m h_m(\mathbf{x})}{\sum_{m=1}^M \beta_m} = \sum_m \alpha_m h_m(\mathbf{x}).$$

This way the artificial prediction market can model linear aggregation of classifiers. Methods such as Adaboost (Freund and Schapire, 1996; Friedman et al., 2000; Schapire, 2003) and Random Forest (Breiman, 2001) also aggregate their constituents using linear aggregation. However, there is more to Adaboost and Random Forest than linear aggregation, since it is very important how to construct the constituents that are aggregated.

In particular, the random forest (Breiman, 2001) can be viewed as an artificial prediction market with constant betting (linear aggregation) where all participants are random trees with the same budget $\beta_m = 1, m = 1,...,M$.

We also obtain an analytic form of the budget update:

$$\beta_m \leftarrow \beta_m - \beta_m \sum_{k=1}^K \phi_m^k(\mathbf{x}) + \beta_m \frac{\phi_m^y(\mathbf{x}) \sum_{j=1}^M \sum_{k=1}^K \beta_j \phi_j^k(\mathbf{x})}{\sum_{j=1}^M \beta_j \phi_j^y(\mathbf{x})}$$

which for classifier based betting functions $\phi_m^k(\mathbf{x},\mathbf{c}) = \eta h_m^k(\mathbf{x})$ becomes:

$$\beta_m \leftarrow \beta_m(1-\eta) + \eta\beta_m \frac{h_m^y(\mathbf{x}) \sum_{j=1}^M \beta_j}{\sum_{j=1}^M \beta_j h_j^y(\mathbf{x})}.$$

This is a novel online update rule for linear aggregation.

3.2 Prediction Markets for Logistic Regression

A variant of logistic regression can also be modeled using prediction markets, with the following betting functions

$$\phi_m^1(\mathbf{x}, 1-c) = (1-c)(x_m^+ - \frac{1}{B}\ln(1-c)),$$
$$\phi_m^2(\mathbf{x}, c) = c(-x_m^- - \frac{1}{B}\ln c)$$

where $x^+ = xI(x > 0), x^- = xI(x < 0)$ and $B = \sum_m \beta_m$. The two class equation (6) becomes: $\sum_{m=1}^M \beta_m c(1-c)(x_m - \ln(1-c)/B + \ln c/B) = 0$ so $\ln \frac{1-c}{c} = \sum_{m=1}^M \beta_m x_m$, which gives the logistic regression model

$$\hat{p}(Y = 1|\mathbf{x}) = c = \frac{1}{1+\exp(\sum_{m=1}^M \beta_m \mathbf{x}_m)}.$$

The budget update equation $\beta_m \leftarrow \beta_m - \eta\beta_m[(1-c)x_m^+ + cx_m^- - H(c)/B] + \eta\beta_m u_y(c)$ is obtained, where $u_1(c) = x_m^+ - \ln(1-c)/B, u_2(c) = -x_m^- - \ln(c)/B$.

Writing $\mathbf{x}\beta = \sum_{m=1}^M \beta_m \mathbf{x}_m$, the budget update can be rearranged to

$$\beta_m \leftarrow \beta_m - \eta\beta_m \left(x_m - \frac{\mathbf{x}\beta}{B}\right)\left(y - \frac{1}{1+\exp(\mathbf{x}\beta)}\right). \tag{8}$$

This equation resembles the standard per-observation update equation for online logistic regression:

$$\beta_m \leftarrow \beta_m - \eta x_m \left(y - \frac{1}{1+\exp(\mathbf{x}\beta)}\right), \tag{9}$$

with two differences. The term $\mathbf{x}\beta/B$ ensures the budgets always sum to B while the factor β_m makes sure that $\beta_m \geq 0$.

The update from Equation (8), like Equation (9) tries to increase $|\mathbf{x}\beta|$, but it does that subject to constraints that $\beta_m \geq 0, m = 1,...,M$ and $\sum_{m=1}^M \beta_m = B$. Observe also that multiplying β by a constant does not change the decision line of the logistic regression.

3.3 Relation to Kernel Methods

Here we construct a market participant from each training example $(x_n, y_n), n = 1,...N$, thus the number of participants M is the number N of training examples. We construct a participant from training example (x_m, y_m) by defining the following betting functions in terms of $u_m(\mathbf{x}) = \frac{\mathbf{x}_m^T\mathbf{x}}{\|\mathbf{x}_m\|\|\mathbf{x}\|}$:

$$\begin{aligned} \phi_m^{y_m}(\mathbf{x}) = u_m(\mathbf{x})^+ &= \begin{cases} u_m(\mathbf{x}) \text{ if } u_m(\mathbf{x}) \geq 0 \\ 0 \text{ else} \end{cases}, \\ \phi_m^{2-y_m}(\mathbf{x}) = -u_m(\mathbf{x})^- &= \begin{cases} 0 \text{ if } u_m(\mathbf{x}) \geq 0 \\ -u_m(\mathbf{x}) \text{ else} \end{cases}. \end{aligned} \tag{10}$$

Observe that these betting functions do not depend on the contract price c, so it is a constant market but not one based on classifiers. The two-class price equation gives

$$c = \frac{\sum_m \beta_m \phi_m^2(\mathbf{x})}{\sum_m \beta_m(\phi_m^1(\mathbf{x}) + \phi_m^2(\mathbf{x}))} = \frac{\sum_m \beta_m[y_m u_m(\mathbf{x}) - u_m(\mathbf{x})^-]}{\sum_m \beta_m |u_m(\mathbf{x})|}$$

since it can be verified that $\phi_m^2(\mathbf{x}) = y_m u_m(\mathbf{x}) - u_m(\mathbf{x})^-$ and $\phi_m^1(\mathbf{x}) + \phi_m^2(\mathbf{x}) = |u_m(\mathbf{x})|$.

The decision rule $c > 0.5$ becomes $\sum_m \beta_m \phi_m^2(\mathbf{x}) > \sum_m \beta_m \phi_m^1(\mathbf{x})$ or $\sum_m \beta_m(\phi_m^2(\mathbf{x}) - \phi_m^1(\mathbf{x})) > 0$. Since $\phi_m^2(\mathbf{x}) - \phi_m^1(\mathbf{x}) = (2y_m - 2)u_m(\mathbf{x}) = (2y_m - 2)\frac{\mathbf{x}_m^T\mathbf{x}}{\|\mathbf{x}_m\|\|\mathbf{x}\|}$ (since in our setup $y_m \in \{1,2\}$), we obtain the SVM type of decision rule with $\alpha_m = \beta_m/\|\mathbf{x}_m\|$:

$$h(\mathbf{x}) = \text{sgn}(\sum_{m=1}^M \alpha_m(2y_m - 3)\mathbf{x}_m^T\mathbf{x}).$$

The budget update becomes in this case:

$$\beta_m \leftarrow \beta_m - \eta\beta_m|u_m(\mathbf{x})| + \eta\beta_m \frac{\phi_m^y(\mathbf{x})}{c_y}.$$

The same reasoning carries out for $u_m(\mathbf{x}) = K(\mathbf{x}_m, \mathbf{x})$ with the RBF kernel $K(\mathbf{x}_m, \mathbf{x}) = \exp(-\|\mathbf{x}_m - \mathbf{x}\|^2/\sigma^2)$. In Figure 3, left, is shown an example of the decision boundary of a market trained online with an RBF kernel with $\sigma = 0.2$ on 1000 examples uniformly sampled in the $[-1,1]^2$ interval. In Figure 3, right is shown the estimated probability $\hat{p}(y = 1|\mathbf{x})$.

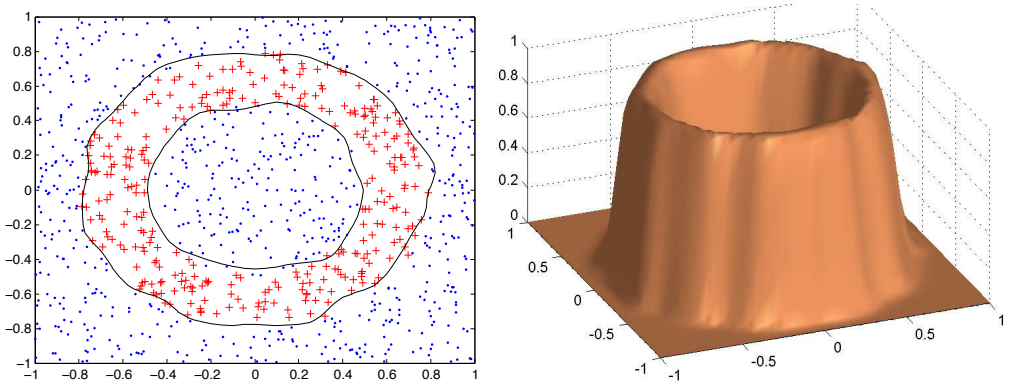


Figure 3: Left: 1000 training examples and learned decision boundary (right) for an RBF kernel-based market from Equation (10) with $\sigma = 0.1$. Right: estimated probability function.

This example shows that the artificial prediction market is an online method with enough modeling power to represent complex decision boundaries such as those given by RBF kernels through the betting functions of the participants. It will be shown in Theorem 5 that the constant market maximizes the likelihood, so it is not clear yet what can be done to obtain a small number of support vectors as in the online kernel-based methods (Bordes et al., 2005; Cauwenberghs and Poggio, 2001; Kivinen et al., 2004).

4. Prediction Markets and Maximum Likelihood

This section discusses what type of optimization is performed during the budget update from Equation (3). Specifically, we prove that the artificial prediction markets perform maximum likelihood learning of the parameters by a version of gradient ascent.

Consider the reparametrization $\gamma = (\gamma_1, ..., \gamma_M) = (\sqrt{\beta_1}, ..., \sqrt{\beta_M})$. The market price $\mathbf{c}(\mathbf{x}) = (c_1(\mathbf{x}), ..., c_K(\mathbf{x})$ is an estimate of the class probability $p(y = k|\mathbf{x})$ for each instance $\mathbf{x} \in \Omega$. Thus a set of training observations $(\mathbf{x}_i, y_i), i = 1, ..., N$, since $\hat{p}(y = y_i|\mathbf{x}_i) = c_{y_i}(\mathbf{x}_i)$, the (normalized) log-likelihood function is

$$L(\gamma) = \frac{1}{N}\sum_{i=1}^{N} \ln \hat{p}(y = y_i|\mathbf{x}_i) = \frac{1}{N}\sum_{i=1}^{N} \ln c_{y_i}(\mathbf{x}_i). \tag{11}$$

We will again use the total amount bet $B(\mathbf{x}, \mathbf{c}) = \sum_{m=1}^{M}\sum_{k=1}^{K} \beta_m \phi_m^k(\mathbf{x}, \mathbf{c})$ for observation $\mathbf{x}$ at market price $\mathbf{c}$.

We will first focus on the constant market $\phi_m^k(\mathbf{x}, \mathbf{c}) = \phi_m^k(\mathbf{x})$, in which case $B(\mathbf{x}, \mathbf{c}) = B(\mathbf{x}) = \sum_{m=1}^{M}\sum_{k=1}^{K} \beta_m \phi_m^k(\mathbf{x})$. We introduce a batch update on all the training examples $(\mathbf{x}_i, y_i), i = 1, ..., N$:

$$\beta_m \leftarrow \beta_m + \beta_m \frac{\eta}{N}\sum_{i=1}^{N} \frac{1}{B(\mathbf{x}_i)} \left(\frac{\phi_m^{y_i}(\mathbf{x}_i)}{c_{y_i}(\mathbf{x}_i)} - \sum_{k=1}^{K} \phi_m^k(\mathbf{x}_i) \right). \tag{12}$$

Equation (12) can be viewed as presenting all observations $(\mathbf{x}_i, y_i)$ to the market simultaneously instead of sequentially. The following statement is proved in the Appendix

Theorem 5 ML for constant market. *The update* (12) *for the constant market maximizes the likelihood* (11) *by gradient ascent on* γ *subject to the constraint* $\sum_{m=1}^{M} \gamma_m^2 = 1$. *The incremental update*

$$\beta_m \leftarrow \beta_m + \beta_m \frac{\eta}{B(\mathbf{x}_i)} \left(\frac{\phi_m^{y_i}(\mathbf{x}_i)}{c_{y_i}(\mathbf{x}_i)} - \sum_{k=1}^{K} \phi_m^k(\mathbf{x}_i) \right) \tag{13}$$

maximizes the likelihood (11) *by constrained stochastic gradient ascent.*

In the general case of non-constant betting functions, the log-likelihood is

$$L(\gamma) = \sum_{i=1}^{N} \log c_{y_i}(\mathbf{x}_i) = \sum_{i=1}^{N} \log \sum_{m=1}^{M} \gamma_m^2 \phi_m^{y_i}(\mathbf{x}_i, \mathbf{c}(\mathbf{x}_i)) - \sum_{i=1}^{N} \log \sum_{k=1}^{K} \sum_{m=1}^{M} \gamma_m^2 \phi_m^k(\mathbf{x}_i, \mathbf{c}(\mathbf{x}_i)). \tag{14}$$

If we ignore the dependence of $\phi_m^k(\mathbf{x}_i, \mathbf{c}(\mathbf{x}_i))$ on γ in (14), and approximate the gradient as:

$$\frac{\partial L(\gamma)}{\partial \gamma_j} \approx \sum_{i=1}^{N} \left(\frac{\gamma_j \phi_j^{y_i}(\mathbf{x}_i, \mathbf{c}(\mathbf{x}_i))}{\sum_{m=1}^{M} \gamma_m^2 \phi_m^{y_i}(\mathbf{x}_i, \mathbf{c}(\mathbf{x}_i))} - \frac{\gamma_j \sum_{k=1}^{K} \phi_j^k(\mathbf{x}_i, \mathbf{c}(\mathbf{x}_i))}{\sum_{k=1}^{K} \sum_{m=1}^{M} \gamma_m^2 \phi_m^k(\mathbf{x}_i, \mathbf{c}(\mathbf{x}_i))} \right)$$

then the proof of Theorem 5 follows through and we obtain the following market update

$$\beta_m \leftarrow \beta_m + \beta_m \frac{\eta}{B(\mathbf{x}, \mathbf{c})} \left[\frac{\phi_m^y(\mathbf{x}, \mathbf{c})}{c_y} - \sum_{k=1}^{K} \phi_m^k(\mathbf{x}, \mathbf{c}) \right], \; m = 1, ..., M. \tag{15}$$

This way we obtain only an approximate statement in the general case

Remark 6 Maximum Likelihood. *The prediction market update* (15) *finds an approximate maximum of the likelihood* (11) *subject to the constraint* $\sum_{m=1}^{M} \gamma_m^2 = 1$ *by an approximate constrained stochastic gradient ascent.*

Observe that the updates from (13) and (15) differ from the update (3) by using an adaptive step size $\eta / B(\mathbf{x}, \mathbf{c})$ instead of the fixed step size 1.

It is easy to check that maximizing the likelihood is equivalent to minimizing an approximation of the expected KL divergence to the true distribution

$$E_\Omega[KL(p(y|\mathbf{x}), c_y(\mathbf{x}))] = \int_\Omega p(\mathbf{x}) \int_Y p(y|\mathbf{x}) \log \frac{p(y|\mathbf{x})}{c_y(\mathbf{x})} dy d\mathbf{x}$$

obtained using the training set as Monte Carlo samples from $p(\mathbf{x}, y)$.

In many cases the number of negative examples is much larger than the positive examples, and is desired to maximize a weighted log-likelihood

$$L(\gamma) = \frac{1}{N} \sum_{i=1}^{N} w(\mathbf{x}_i) \ln c_{y_i}(\mathbf{x}_i).$$

This can be achieved (exactly for constant betting and approximately in general) using the weighted update rule

$$\beta_m \leftarrow \beta_m + \eta w(\mathbf{x}) \frac{\beta_m}{B(\mathbf{x},\mathbf{c})} \left[\frac{\phi_m^y(\mathbf{x},\mathbf{c})}{c_y} - \sum_{k=1}^{K} \phi_m^k(\mathbf{x},\mathbf{c}) \right], \ m = 1,...,M. \tag{16}$$

The parameter η and the number of training epochs can be used to control how close the budgets β are to the ML optimum, and this way avoid overfitting the training data.

An important issue for the real prediction markets is the *efficient market hypothesis*, which states that the market price fuses in an optimal way the information available to the market participants (Fama, 1970; Basu, 1977; Malkiel, 2003). From Theorem 5 we can draw the following conclusions for the artificial prediction market with constant betting:

1. In general, an untrained market (in which the budgets have not been updated based on training data) will not satisfy the efficient market hypothesis.

2. The market trained with a large amount of representative training data and small η satisfies the efficient market hypothesis.

5. Specialized Classifiers

The prediction market is capable of fusing the information available to the market participants, which can be trained classifiers. These classifiers are usually suboptimal, due to computational or complexity constraints, to the way they are trained, or other reasons.

In boosting, all selected classifiers are aggregated for each instance $\mathbf{x} \in \Omega$. This can be detrimental since some classifiers could perform poorly on subregions of the instance space Ω, degrading the performance of the boosted classifier. In many situations there exist simple rules that hold on subsets of Ω but not on the entire Ω. Classifiers trained on such subsets $D_i \subset \Omega$, would have small misclassification error on D_i but unpredictable behavior outside of D_i. The artificial prediction market can aggregate such classifiers, transformed into participants that don't bet anything outside of their domain of expertise $D_i \subset \Omega$. This way, for different instances $\mathbf{x} \in \Omega$, different subsets of participants will contribute to the resulting probability estimate. We call these *specialized classifiers* since they only give their opinion through betting on observations that fall inside their domain of specialization.

Thus a specialized classifier with a domain D would have a betting function of the form:

$$\phi^k(\mathbf{x},\mathbf{c}) = \begin{cases} \varphi^k(\mathbf{x},\mathbf{c}) \text{ if } \mathbf{x} \in D \\ 0 \text{ else} \end{cases}. \tag{17}$$

This idea is illustrated on the following simple 2D example of a triangular region, shown in Figure 4, with positive examples inside the triangle and negatives outside. An accurate classifier for that region can be constructed using six market participants, one for each half-plane determined by each side of the triangle.

Three of these classifiers correspond to the three half planes that are outside the triangle. These participants have 100% accuracy in predicting the observations, all negatives, that fall in their half planes and don't bet anything outside of their half planes. The other three classifiers are not very good, and will have smaller budgets. On an observation that lies outside of the triangle, one or two of the high-budget classifiers will bet a large amount on the correct prediction and will drive the

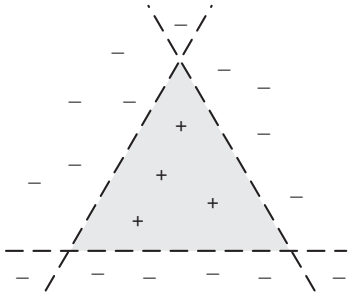


Figure 4: A perfect classifier can be constructed for the triangular region above from a market of six specialized classifiers that only bid on a half-plane determined by one side of the triangle. Three of these specialized classifiers have 100% accuracy while the other three have low accuracy. Nevertheless, the market is capable of obtaining 100% overall accuracy.

output probability. When an observation falls inside the triangle, only the small-budget classifiers will participate but will be in agreement and still output the correct probability. Evaluating this market on 1000 positives and 1000 negatives showed that the market obtained a prediction accuracy of 100%.

There are many ways to construct specialized classifiers, depending on the problem setup. In natural language processing for example, a specialized classifier could be based on grammar rules, which work very well in many cases, but not always.

We propose two generic sets of specialized classifiers. The first set are the leaves of the random trees of a random forest while the second set are the leaves of the decision trees trained by adaboost. Each leaf f is a rule that defines a domain $D_f = \{\mathbf{x} \in \Omega, f(\mathbf{x}) = 1\}$ of the instances that obey that rule. The betting function of this specialized classifier is given in Equation (17) where $\varphi_f^k(\mathbf{x}, \mathbf{c})$ is based on the associated classifier $h_f^k(\mathbf{x}) = n_{fk}/n_f$, obtaining constant, linear and aggressive versions. Here n_{fk} is the number of training instances of class k that obey rule f and $n_f = \sum_k n_{fk}$. By the way the random trees are trained, usually $n_f = n_{fk}$ for some k.

In Friedman and Popescu (2008) these rules were combined using a linear aggregation method similar to boosting. One could also use other nodes of the random tree, not necessarily the leaves, for the same purpose.

It can be verified using Equation (7) that constant specialized betting is the linear aggregation of the participants that are currently betting. This is different than the linear aggregation of all the classifiers.

6. Related Work

This work borrows prediction market ideas from Economics and brings them to Machine Learning for supervised aggregation of classifiers or features in general.

Related work in Economics. Recent work in Economics (Manski, 2006; Perols et al., 2009; Plott et al., 2003) investigates the information fusion of the prediction markets. However, none of these works aims at using the prediction markets as a tool for learning class probability estimators in a supervised manner.

Some works (Perols et al., 2009; Plott et al., 2003) focus on parimutuel betting mechanisms for combining classifiers. In parimutuel betting contracts are sold for all possible outcomes (classes) and the entire budget (minus fees) is divided between the participants that purchased contracts for the winning outcome. Parimutuel betting has a different way of fusing information than the Iowa prediction market.

The information based decision fusion (Perols et al., 2009) is a first version of an artificial prediction market. It aggregates classifiers through the parimutuel betting mechanism, using a loop that updates the odds for each outcome and takes updated bets until convergence. This insures a stronger information fusion than without updating the odds. Our work is different in many ways. First our work uses the Iowa electronic market instead of parimutuel betting with odds-updating. Using the Iowa model allowed us to obtain a closed form equation for the market price in some important cases. It also allowed us to relate the market to some existing learning methods. Second, our work presents a multi-class formulation of the prediction markets as opposed to a two-class approach presented in Perols et al. (2009). Third, the analytical market price formulation allowed us to prove that the constant market performs maximum likelihood learning. Finally, our work evaluates the prediction market not only in terms of classification accuracy but also in the accuracy of predicting the exact class conditional probability given the evidence.

Related work in Machine Learning. Implicit online learning (Kulis and Bartlett, 2010) presents a generic online learning method that balances between a "conservativeness" term that discourages large changes in the model and a "correctness" term that tries to adapt to the new observation. Instead of using a linear approximation as other online methods do, this approach solves an implicit equation for finding the new model. In this regard, the prediction market also solves an implicit equation at each step for finding the new model, but does not balance two criteria like the implicit online learning method. Instead it performs maximum likelihood estimation, which is consistent and asymptotically optimal. In experiments, we observed that the prediction market obtains significantly smaller misclassification errors on many data sets compared to implicit online learning.

Specialization can be viewed as a type of reject rule (Chow, 1970; Tortorella, 2004). However, instead of having a reject rule for the aggregated classifier, each market participant has his own reject rule to decide on what observations to contribute to the aggregation. ROC-based reject rules (Tortorella, 2004) could be found for each market participant and used for defining its domain of specialization. Moreover, the market can give an overall reject rule on hopeless instances that fall outside the specialization domain of all participants. No participant will bet for such an instance and this can be detected as an overall rejection of that instance.

If the overall reject option is not desired, one could avoid having instances for which no classifiers bet by including in the market a set of participants that are all the leaves of a number of random trees. This way, by the design of the random trees, it is guaranteed that each instance will fall into at least one leaf, that is, participant, hence the instance will not be rejected.

A simplified specialization approach is taken in delegated classifiers (Ferri et al., 2004). A first classifier would decide on the relatively easy instances and would delegate more difficult examples to a second classifier. This approach can be seen as a market with two participants that are not overlapping. The specialization domain of the second participant is defined by the first participant. The market takes a more generic approach where each classifier decides independently on which instances to bet.

The same type of leaves of random trees (i.e., rules) were used by Friedman and Popescu (2008) for linear aggregation. However, our work presents a more generic aggregation method through the

prediction market, with linear aggregation as a particular case, and we view the rules as one sort of specialized classifiers that only bid in a subdomain of the feature space.

Our earlier work (Lay and Barbu, 2010) focused only on aggregation of classifiers and did not discuss the connection between the artificial prediction markets and logistic regression, kernel methods and maximum likelihood learning. Moreover, it did not include an experimental comparison with implicit online learning and adaboost.

Two other prediction market mechanisms have been recently proposed in the literature. The first one (Chen and Vaughan, 2010; Chen et al., 2011) has the participants entering the market sequentially. Each participant is paid by an entity called the market maker according to a predefined scoring rule. The second prediction market mechanism is the machine learning market (Storkey, 2011; Storkey et al., 2012), dealing with all participants simultaneously. Each market participant purchases contracts for the possible outcomes to maximize its own utility function. The equilibrium price of the contracts is computed by an optimization procedure. Different utility functions result in different forms of the equilibrium price, such as the mean, median, or geometric mean of the participants' beliefs.

7. Experimental Validation

In this section we present experimental comparisons of the performance of different artificial prediction markets with random forest, adaboost and implicit online learning (Kulis and Bartlett, 2010).

Four artificial prediction markets are evaluated in this section. These markets have the same classifiers, namely the leaves of the trained random trees, but differ either in the betting functions or in the way the budgets are trained as follows:

1. The first market has constant betting and equal budgets for all participants. We proved in Section 3.1 that this is a random forest (Breiman, 2001).

2. The second market has constant betting based on specialized classifiers (the leaves of the random trees), with the budgets initialized with the same values like the market 1 above, but trained using the update equation (13). Thus after training it will be different from market 1.

3. The third market has linear betting functions (1), for which the market price can be computed analytically only for binary classification. The market is initialized with equal budgets and trained using Equation (15).

4. The fourth market has aggressive betting (2) with $\varepsilon = 0.01$ and the market price computed using the Mann iteration Algorithm 3. The market is initialized with equal budgets and trained using Equation (15). The value $\varepsilon = 0.01$ was chosen for simplicity; a better choice would be to obtain it by cross-validation.

For each data set, 50 random trees are trained on bootstrap samples of the training data. These trained random trees are used to construct the random forest and the other three markets described above. This way only the aggregation capabilities of the different markets are compared.

The budgets in the markets 2-4 described above are trained on the same training data using the update equation (15) which simplifies to (13) for the constant market.

A C++ implementation of these markets can be found at the following address: `http://stat.fsu.edu/~abarbu/Research/PredMarket.zip`.

7.1 Case Study

We first investigate the behavior of three markets on a data set in terms of training and test error as well as loss function. For that, we chose the `satimage` data set from the UCI repository (Blake and Merz, 1998) since it has a supplied test set. The `satimage` data set has a training set of size 4435 and a test set of size 2000.

The markets investigated are the constant market with both incremental and batch updates, given in Equations (13) and (12) respectively, the linear and aggressive markets with incremental updates given in (15). Observe that the η in Equation (13) is not divided by N (the number of observations) while the η in (12) is divided by N. Thus to obtain the same behavior the η in (13) should be the η from (12) divided by N. We used $\eta = 100/N$ for the incremental update and $\eta = 100$ for the batch update unless otherwise specified.

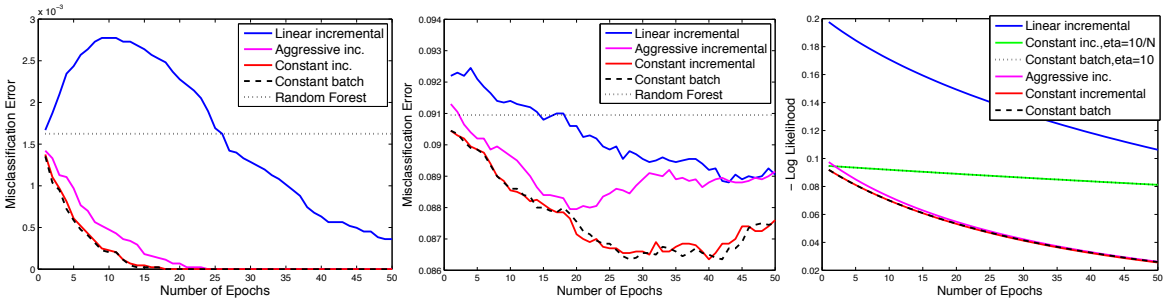


Figure 5: Experiments on the satimage data set for the incremental and batch market updates. Left: The training error vs. number of epochs. Middle: The test error vs. number of epochs. Right: The negative log-likelihood function vs. number of training epochs. The learning rates are $\eta = 100/N$ for the incremental update and $\eta = 100$ for the batch update unless otherwise specified.

In Figure 5 are plotted the misclassification errors on the training and test sets and the negative log-likelihood function vs. the number of training epochs, averaged over 10 runs. From Figure 5 one could see that the incremental and batch updates perform similarly in terms of the likelihood function, training and test errors. However, the incremental update is preferred since it is requires less memory and can handle an arbitrarily large amount of training data. The aggressive and constant markets achieve similar values of the negative log likelihood and similar training errors, but the aggressive market seems to overfit more since the test error is larger than the constant incremental (p-value< 0.05). The linear market has worse values of the log-likelihood, training and test errors (p-value< 0.05).

7.2 Evaluation of the Probability Estimation and Classification Accuracy on Synthetic Data

We perform a series of experiments on synthetic data sets to evaluate the market's ability to predict class conditional probabilities $P(Y|\mathbf{x})$. The experiments are performed on 5000 binary data sets with 50 levels of Bayes error

$$E = \int \min\{p(\mathbf{x}, Y = 0), p(\mathbf{x}, Y = 1)\} d\mathbf{x},$$

ranging from 0.01 to 0.5 with equal increments. For each data set, the two classes have equal frequency. Both $p(\mathbf{x}|Y=k), k=0,1$ are normal distributions $\mathcal{N}(\mu_k, \sigma^2 I)$, with $\mu_0 = 0, \sigma^2 = 1$ and μ_1 chosen in some random direction at such a distance to obtain the desired Bayes error.

For each of the 50 Bayes error levels, 100 data sets of size 200 were generated using the bisection method to find an appropriate μ_1 in a random direction. Training of the participant budgets is done with $\eta = 0.1$.

For each observation $\mathbf{x}$, the class conditional probability can be computed analytically using the Bayes rule

$$p^*(Y=1|\mathbf{x}) = \frac{p(\mathbf{x}|Y=1)p(Y=1)}{p(\mathbf{x},Y=0)+p(\mathbf{x},Y=1)}.$$

An estimation $\hat{p}(y=1|\mathbf{x})$ obtained with one of the markets is compared to the true probability

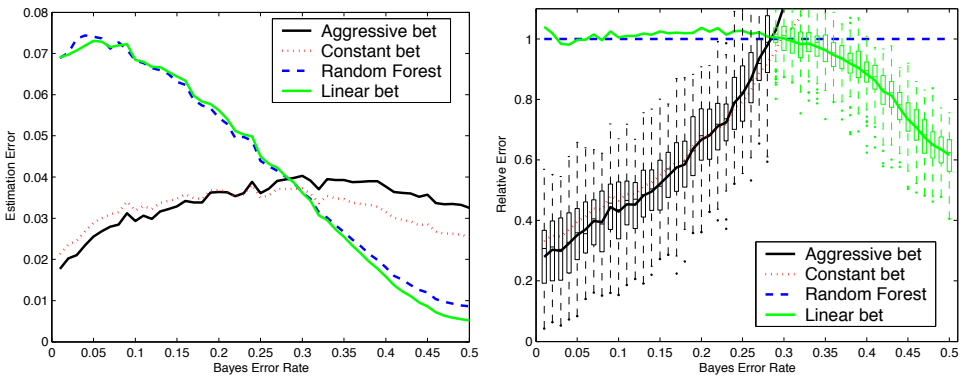


Figure 6: Left: Class probability estimation error vs problem difficulty for 5000 100D problems. Right: Probability estimation errors relative to random forest. The aggressive and linear betting are shown with box plots.

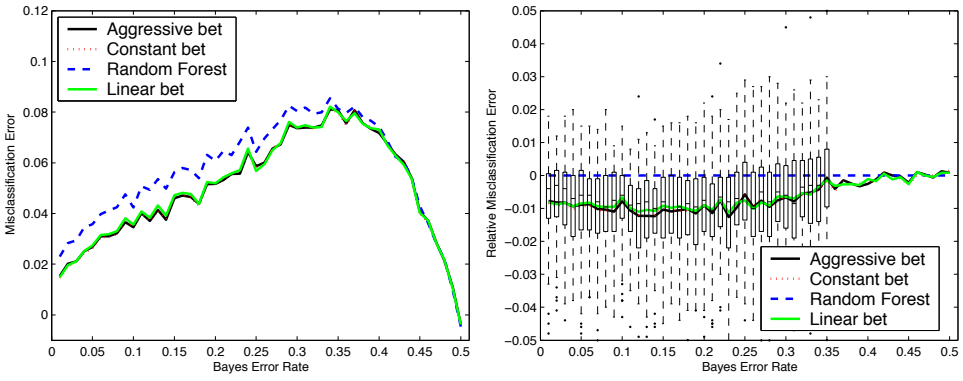


Figure 7: Left: Misclassification error minus Bayes error vs problem difficulty for 5000 100D problems. Right: Misclassification errors relative to random forest. The aggressive betting is shown with box plots.

$p^*(Y=1|\mathbf{x})$ using the L_2 norm

$$E(\hat{p},p^*)=\int(\hat{p}(y=1|\mathbf{x})-p^*(y=1|\mathbf{x}))^2p(\mathbf{x})d\mathbf{x}$$

where $p(\mathbf{x})=p(\mathbf{x},Y=0)+p(\mathbf{x},Y=1)$.

In practice, this error is approximated using a sample of size 1000. The errors of the probability estimates obtained by the four markets are shown in Figure 6 for a 100D problem setup. Also shown on the right are the errors relative to the random forest, obtained by dividing each error to the corresponding random forest error. As one could see, the aggressive and constant betting markets obtain significantly better (p-value < 0.01) probability estimators than the random forest, for Bayes errors up to 0.28. On the other hand, the linear betting market obtains probability estimators significantly better (p-value < 0.01) than the random forest for Bayes error from 0.34 to 0.5.

We also evaluated the misclassification errors of the four markets in predicting the correct class, for the same 5000 data sets. The difference between these misclassification errors and the Bayes error are shown in Figure 7, left. The difference between these misclassification errors and the random forest error are shown in Figure 7, right. We see that all markets with trained participants predict significantly better (p-value < 0.01) than random forest for Bayes errors up to 0.3, and behave similar to random forest for the remaining data sets.

7.3 Comparison with Random Forest on UCI Data Sets

In this section we conduct an evaluation on 31 data sets from the UCI machine learning repository (Blake and Merz, 1998). The optimal number of training epochs and η are meta-parameters that need to be chosen appropriately for each data set. We observed experimentally that η can take any value up to a maximum that depends on the data set. In these experiments we took $\eta = 10/N_{train}$. The best number of epochs was chosen by ten fold cross-validation.

In order to compare with the results in Breiman (2001), the training and test sets were randomly subsampled from the available data, with 90% for training and 10% for testing. The exceptions are the `satimage`, `zipcode`, `hill-valley` and `poker`data sets with test sets of size $2000, 2007, 606, 10^6$ respectively. All results were averaged over 100 runs.

We present two random forest results. In the column named RFB are presented the random forest results from Breiman (2001) where each tree node is split based on a random feature. In the column named RF we present the results of our own RF implementation with splits based on random features. The leaf nodes of the random trees from our RF implementation are used as specialized participants for all the markets evaluated.

The CB, LB and AB columns are the performances of the constant, linear and respectively aggressive markets on these data sets.

Significant mean differences ($\alpha < 0.01$) from RFB are shown with $+,-$ for when RFB is worse respectively better. Significant paired t-tests (Demšar, 2006) ($\alpha < 0.01$) that compare the markets with our RF implementation are shown with $\bullet,\dagger$ for when RF is worse respectively better.

The constant, linear and aggressive markets significantly outperformed our RF implementation on 22, 19 respectively 22 data sets out of the 31 evaluated. They were not significantly outperformed by our RF implementation on any of the 31 data sets.

Compared to the RF results from Breiman (2001) (RFB), CB, LB and AB significantly outperformed RFB on 6,5,6 data sets respectively, and were not significantly outperformed on any data set.

Data	N_{train}	N_{test}	F	K	RFB	RF	CB	LB	AB
breast-cancer	683	–	9	2	2.7	2.5	2.4	2.4	2.4
sonar	208	–	60	2	18.0	16.6	14.1 •+	14.2 •+	14.1 •+
vowel	990	–	10	11	3.3	2.9	2.6 •+	2.7 +	2.6 •+
ecoli	336	–	7	8	13.0	12.9	12.9	12.8	12.9
german	1000	–	24	2	26.2	25.5	24.9 •+	25.1	24.9 •+
glass	214	–	9	6	21.2	23.5	22.2 •	22.4	22.2 •
image	2310	–	19	7	2.7	2.7	2.5 •	2.5 •	2.5 •
ionosphere	351	–	34	2	7.5	7.4	6.7 •	6.9 •	6.7 •
letter-recognition	20000	–	16	26	4.7	4.2 +	4.2 •+	4.2 •+	4.2 •+
liver-disorders	345	–	6	2	24.7	26.5	26.3	26.2	26.2
pima-diabetes	768	–	8	2	24.3	24.1	23.8	23.7	23.8
satimage	4435	2000	36	6	10.5	10.1 +	10.0 •+	10.1 •+	10.0 •+
vehicle	846	–	18	4	26.4	26.3	26.1	26.2	26.1
voting-records	232	–	16	2	4.6	5.3	4.2 •	4.2 •	4.2 •
zipcode	7291	2007	256	10	7.8	7.7	7.6 •+	7.7 •+	7.6 •+
abalone	4177	–	8	3	–	45.5	45.4	45.4	45.4
balance-scale	625	–	4	3	–	15.4	15.4	15.4	15.4
car	1728	–	6	4	–	2.8	2.0 •	2.2 •	2.0 •
connect-4	67557	–	42	3	–	19.6	19.3 •	19.4 •	19.5 •
cylinder-bands	277	–	33	2	–	22.7	20.9 •	21.1 •	20.9 •
hill-valley	606	606	100	2	–	46.9	45.8 •	46.3 •	45.8 •
isolet	1559	–	617	26	–	17.0	15.7 •	15.8 •	15.7 •
king-rook-vs-king	28056	–	6	18	–	15.6	15.4 •	15.4 •	15.4 •
king-rk-vs-k-pawn	3196	–	36	2	–	2.0	1.5 •	1.6 •	1.5 •
madelon	2000	–	500	2	–	46.1	45.2 •	45.3 •	45.2 •
magic	19020	–	10	2	–	12.0	11.9 •	11.9 •	11.9 •
musk	6598	–	166	2	–	3.7	3.5 •	3.6 •	3.5 •
poker	25010	10^6	10	10	–	43.2	43.1 •	43.1 •	43.1 •
SAheart	462	–	9	2	–	30.8	30.8	30.7	30.8
splice-junction	3190	–	59	3	–	18.9	17.7 •	18.2 •	17.7 •
yeast	1484	–	8	10	–	38.3	38.1	38.0	38.1

Table 1: The misclassification errors for 31 data sets from the UC Irvine Repository are shown in percent (%).. The markets evaluated are our implementation of random forest (RF), and markets with Constant (CB), Linear (LB) and respectively Aggressive (AB) Betting. RFB contains the random forest results from Breiman (2001).

7.4 Comparison with Implicit Online Learning on UCI Data Sets

We implemented the implicit online learning (Kulis and Bartlett, 2010) algorithm for classification with linear aggregation. The objective of implicit online learning is to minimize the loss $\ell(\beta)$ in a *conservative* way. The *conservativeness* of the update is determined by a Bregman divergence

$$D(\beta, \beta^t) = \phi(\beta) - \phi(\beta^t) - \langle \nabla\phi(\beta^t), \beta - \beta^t \rangle$$

where $\phi(\beta)$ are real-valued strictly convex functions. Rather than minimize the loss function itself, the function

$$f_t(\beta) = D(\beta, \beta^t) + \eta_t \ell(\beta)$$

is minimized instead. Here η_t is the learning rate. The Bregman divergence ensures that the optimal β is not *too far* from β^t. The algorithm for implicit online learning is as follows

$$\tilde{\beta}^{t+1} = \operatorname*{argmin}_{\beta \in \mathbb{R}^M} f_t(\beta)$$
$$\beta^{t+1} = \operatorname*{argmin}_{\beta \in S} D(\beta, \tilde{\beta}^{t+1}).$$

The first step solves the unconstrained version of the problem while the second step finds the *nearest* feasible solution to the unconstrained minimizer subject to the Bregman divergence.

For our problem we use

$$\ell(\beta) = -\log(c_y(\beta))$$

where $c_y(\beta)$ is the constant market equilibrium price for ground truth label y. We chose the squared Euclidean distance $D(\beta, \beta^t) = \|\beta - \beta^t\|_2^2$ as our Bregman divergence and learning rate $\eta_t = 1/\sqrt{t}$. To ensure that $c = \sum_{m=1}^M h_m \beta_m = H\beta$ is a valid probability vector, the feasible solution set is therefore $S = \{\beta \in [0,1]^M : \sum_{m=1}^M \beta_m = 1\}$. This gives the following update scheme

$$\tilde{\beta}^{t+1} = \beta^t + \eta_t \frac{1}{p} (H^y)^T$$
$$\beta^{t+1} = \operatorname*{argmin}_{\beta \in S} \left\{ \|\beta - \tilde{\beta}^{t+1}\|_2^2 \right\}$$

where $H^y = \left(h_1^y, h_2^y, \ldots, h_M^y\right)$ is the vector of classifier outputs for the true label y, $q = H^y \beta^t$, $r = H^y (H^y)^T$ and $p = \frac{1}{2}\left(q + \sqrt{q^2 + 4\eta_t r}\right)$.

The results presented in Table 2 are obtained by 10 fold cross-validation. The cross-validation errors were averaged over 10 different permutations of the data in the cross-validation folds.

The results from CB online and implicit online are obtained in one epoch. The results from the CB offline and implicit offline columns are obtained in an off-line fashion using an appropriate number of epochs (up to 10) to obtain the smallest cross-validated error on a random permutation of the data that is different from the 10 permutations used to obtain the results.

The comparisons are done with paired t-tests and shown with $*$ and ‡ when the constant betting market is significantly ($\alpha < 0.01$) better or worse than the corresponding implicit online learning. We also performed a comparison with our RF implementation, and significant differences are shown with • and †.

Compared to RF, implicit online learning won 5-0, CB online won in 9-1 and CB offline won 12-0.

Compared to implicit online, which performed identical with implicit offline, both CB online and CB offline won 9-0.

Data Set	N_{train}	N_{test}	F	K	RF	Implicit Online	CB Online	Implicit Offline	CB Offline
breast-cancer	683	–	9	2	3.1	3.1	3	3.1	3
sonar	208	–	60	2	15.1	15.2	15.3	15.1	14.6
vowel	990	–	10	11	3.2	3.2	3.2	3.2	2.9 •∗
ecoli	336	–	7	8	13.7	13.7	13.6	13.7	13.6
german	1000	–	24	2	23.6	23.5	23.5	23.5	23.4
glass	214	–	9	6	21.4	21.4	21.3	21.4	21
image	2310	–	19	7	1.9	1.9	1.9	1.9	1.8 •
ionosphere	351	–	34	2	6.4	6.5	6.5	6.5	6.5
letter-recognition	20000	–	16	26	3.3	3.3	3.3 •∗	3.3	3.3
liver-disorders	345	–	6	2	26.4	26.4	26.4	26.4	26.4
pima-diabetes	768	–	8	2	23.2	23.2	23.2	23.2	23.2
satimage	4435	2000	36	6	8.8	8.8	8.8	8.8	8.7 •
vehicle	846	–	18	4	24.8	24.7	24.9	24.7	24.9
voting-records	232	–	16	2	3.5	3.5	3.5	3.5	3.5
zipcode	7291	2007	256	10	6.1	6.1	6.2	6.1	6.2
abalone	4177	–	8	3	45.5	45.5	45.6 †	45.5	45.5
balance-scale	625	–	4	3	17.7	17.7	17.7	17.7	17.7
car	1728	–	6	4	2.3	2.3	1.8 •∗	2.3	1.1 •∗
connect-4	67557	–	42	3	19.9	19.9 •	19.5 •∗	19.9 •	18.2 •∗
cylinder-bands	277	–	33	2	21.4	21.3	21.2	21.3	20.8 •
hill-valley	606	606	100	2	43.8	43.7	43.7	43.7	43.7
isolet	1559	–	617	26	6.9	6.9	6.9	6.9	6.9
king-rk-vs-king	28056	–	6	18	21.6	21.6 •	19.6 •∗	21.5 •	15.7 •∗
king-rk-vs-k-pawn	3196	–	36	2	1	1	0.7 •∗	1	0.5 •∗
magic	19020	–	10	2	11.9	11.9 •	11.8 •∗	11.9 •	11.7 •∗
madelon	2000	–	500	2	26.8	26.5 •	25.6 •∗	26.4 •	21.6 •∗
musk	6598	–	166	2	1.7	1.7 •	1.6 •∗	1.7 •	1 •∗
splice-junction-gene	3190	–	59	3	4.3	4.3	4.2 •∗	4.3	4.1 •∗
SAheart	462	–	9	2	31.5	31.5	31.6	31.5	31.6
yeast	1484	–	8	10	37.3	37.3	37.3	37.3	37.3

Table 2: Comparison with Implicit Online Learning and random forest using 10-fold cross-validation.

7.5 Comparison with Adaboost for Lymph Node Detection

Finally, we compared the linear aggregation capability of the artificial prediction market with adaboost for a lymph node detection problem. The system is setup as described in Barbu et al. (2012), namely a set of lymph node candidate positions (x,y,z) are obtained using a trained detector. Each candidate is segmented using gradient descent optimization and about 17000 features are extracted from the segmentation result. Using these features, adaboost constructed 32 weak classifiers. Each weak classifier is associated with one feature, splits the feature range into 64 bins and returns a predefined value (1 or -1), for each bin.

Thus, one can consider there are $M = 32 \times 64 = 2048$ specialized participants, each betting for one class (1 or -1) for any observation that falls in its domain. The participants are given budgets $\beta_{ij}, i = 1,..,32, j = 1,..,64$ where i is the feature index and j is the bin index. The participant budgets $\beta_{ij}, j = 1,...,64$ corresponding to the same feature i are initialized the same value β_i, namely the adaboost coefficient. For each bin, the return class 1 or -1 is the outcome for which the participant will bet its budget.

The constant betting market of the 2048 participants is initialized with these budgets and trained with the same training examples that were used to train the adaboost classifier.

The obtained constant market probability for an observation $\mathbf{x} = (x_1,...,x_{32})$ is based on the bin indexes $\mathbf{b} = (b_1(x_1),...,b_{32}(x_{32}))$:

$$p(y = 1|\mathbf{b}) = \frac{\sum_{i=1}^{32} \beta_{i,b_i} h_i(b_i)}{\sum_{i=1}^{32} \beta_{i,b_i}}.$$

An important issue is that the number N_{pos} of positive examples is much smaller than the number N_{neg} of negatives. Similar to adaboost, the sum of the weights of the positive examples should be the same as the sum of weights of the negatives. To accomplish this in the market, we use the weighted update rule Equation (16), with $w_{pos} = \frac{1}{N_{pos}}$ for each positive example and $w_{neg} = \frac{1}{N_{neg}}$ for each negative.

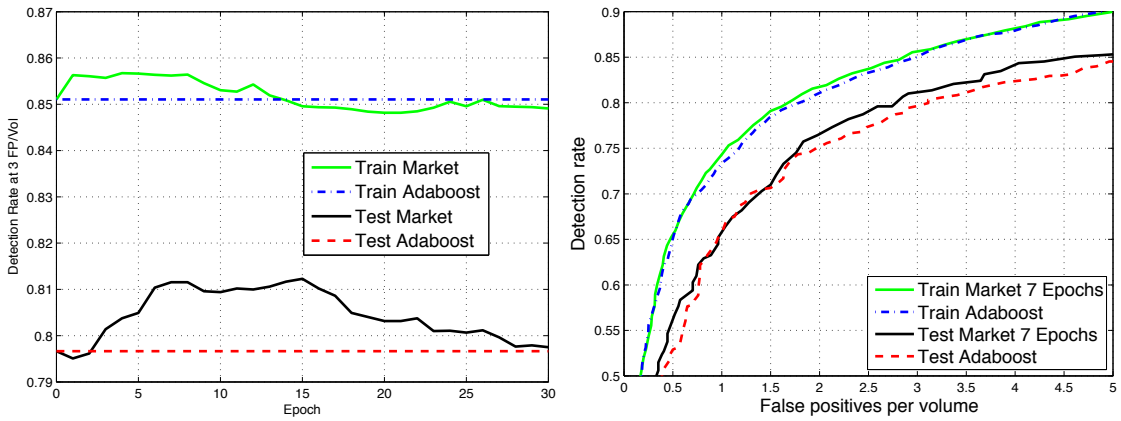


Figure 8: Left: Detection rate at 3 FP/vol vs. number of training epochs for a lymph node detection problem. Right: ROC curves for adaboost and the constant betting market with participants as the 2048 adaboost weak classifier bins. The results are obtained with six-fold cross-validation.

The adaboost classifier and the constant market were evaluated for a lymph node detection application on a data set containing 54 CT scans of the pelvic and abdominal region, with a total of 569 lymph nodes, with six-fold cross-validation. The evaluation criterion is the same for all methods, as specified in Barbu et al. (2012). A lymph node detection is considered correct if its center is inside a manual solid lymph node segmentation and is incorrect if it not inside any lymph node segmentation (solid or non-solid).

In Figure 8, left, is shown the training and testing detection rate at 3 false positives per volume (a clinically acceptable false positive rate) vs the number of training epochs. We see the detection rate increases to about 81% for epochs 6 to 16 epochs and then gradually decreases. In Figure 8, right, are shown the training and test ROC curves of adaboost and the constant market trained with 7 epochs. In this case the detection rate at 3 false positives per volume improved from 79.6% for adaboost to 81.2% for the constant market. The p-value for this difference was 0.0276 based on paired t-test.

8. Conclusion and Future Work

This paper presents a theory for artificial prediction markets for the purpose of supervised learning of class conditional probability estimators. The artificial prediction market is a novel online learning algorithm that can be easily implemented for two class and multi class applications. Linear aggregation, logistic regression as well as certain kernel methods can be viewed as particular instances of the artificial prediction markets. Inspired from real life, specialized classifiers that only bet on subsets of the instance space Ω were introduced. Experimental comparisons on real and synthetic data show that the prediction market usually outperforms random forest, adaboost and implicit online learning in prediction accuracy.

The artificial prediction market shows the following promising features:

1. It can be *updated online* with minimal computational cost when a new observation $(\mathbf{x}, y)$ is presented.

2. It has a simple form of the update iteration that can be easily implemented.

3. For multi-class classification it can *fuse information* from all types of binary or multi-class classifiers: for example, trained *one-vs-all, many-vs-many*, multi-class decision tree, etc.

4. It can obtain meaningful probability estimates when only a subset of the market participants are involved for a particular instance $\mathbf{x} \in X$. This feature is useful for learning on manifolds (Belkin and Niyogi, 2004; Elgammal and Lee, 2004; Saul and Roweis, 2003), where the location on the manifold decides which market participants should be involved. For example, in face detection, different face part classifiers (eyes, mouth, ears, nose, hair, etc) can be involved in the market, depending on the orientation of the head hypothesis being evaluated.

5. Because of their betting functions, the *specialized* market participants can decide for which instances they bet and how much. This is another way to combine classifiers, different from the boosting approach where all classifiers participate in estimating the class probability for each observation.

We are currently extending the artificial prediction market framework to regression and density estimation. These extensions involve contracts for uncountably many outcomes but the update and the market price equations extend naturally.

Future work includes finding explicit bounds for the generalization error based on the number of training examples. Another item of future work is finding other generic types specialized participants that are not leaves of random or adaboost trees. For example, by clustering the instances $\mathbf{x} \in \Omega$, one could find regions of the instance space Ω where simple classifiers (e.g., logistic regression, or betting for a single class) can be used as specialized market participants for that region.

Acknowledgments

The authors wish to thank Jan Hendrik Schmidt from Innovation Park Gmbh. for stirring in us the excitement for the prediction markets. The authors acknowledge partial support from FSU startup grant and ONR N00014-09-1-0664.

Appendix A. Proofs

Proof [of Theorem 1] From Equation (3), the total budget $\sum_{m=1}^{M} \beta_m$ is conserved if and only if

$$\sum_{m=1}^{M} \sum_{k=1}^{K} \beta_m \phi_m^k(\mathbf{x}, \mathbf{c}) = \sum_{m=1}^{M} \beta_m \phi_m^y(\mathbf{x}, \mathbf{c})/c_y. \tag{18}$$

Denoting $n = \sum_{m=1}^{M} \sum_{k=1}^{K} \beta_m \phi_m^k(\mathbf{x}, \mathbf{c})$, and since the above equation must hold for all y, we obtain that Equation (4) is a necessary condition and also $c_k \neq 0, k = 1, ..., K$, which means $c_k > 0, k = 1, ..., K$. Reciprocally, if $c_k > 0$ and Equation (4) hold for all k, dividing by c_k we obtain Equation (18). ■

Proof [of Remark 2] Since the total budget is conserved and is positive, there exists a $\beta_m > 0$, therefore $\sum_{m=1}^{M} \beta_m \phi_m^k(\mathbf{x}, 0) > 0$, which implies $\lim_{c_k \to 0} f_k(c_k) = \infty$. From the fact that $f_k(c_k)$ is continuous and strictly decreasing, with $\lim_{c_k \to 0} f_k(c_k) = \infty$ and $\lim_{c_k \to 1} f_k(c_k) = 0$, it implies that for every $n > 0$ there exists a unique c_k that satisfies $f_k(c_k) = n$. ■

Proof [of Theorem 3] From Remark 2 we get that for every $n \geq n_k, n > 0$ there is a unique $c_k(n)$ such that $f_k(c_k(n)) = n$. Moreover, following the proof of Remark 2 we see that $c_k(n)$ is continuous and strictly decreasing on (n_k, ∞), with $\lim_{n \to \infty} c_k(n) = 0$.

If $\max_k n_k > 0$, take $n^* = \max_k n_k$. There exists $k \in \{1, ..., K\}$ such that $n_k = n^*$, so $c_k(n^*) = 1$, therefore $\sum_{j=1}^{K} c_j(n^*) \geq 1$.

If $\max_k n_k = 0$ then $n_k = 0, k = 1, ..., K$ which means $\phi_m^k(\mathbf{x}, 1) = 0, k = 1, ..., K$ for all m with $\beta_m > 0$. Let $a_m^k = \min\{c | \phi_m^k(\mathbf{x}, c) = 0\}$. We have $a_m^k > 0$ for all k since $\phi_m^k(\mathbf{x}, 0) > 0$. Thus $\lim_{n \to 0_+} c_k(n) = \max_m a_m^k \geq a_1^k$, where we assumed that $\phi_1(\mathbf{x}, \mathbf{c})$ satisfies Assumption 2. But from Assumption 2 there exists k such that $a_1^k = 1$. Thus $\lim_{n \to 0_+} \sum_{k=1}^{K} c_k(n) \geq \sum_{k=1}^{K} a_1^k > 1$ so there exists n^* such that $\sum_{k=1}^{K} c_k(n^*) \geq 1$.

Either way, since $\sum_{k=1}^{K} c_k(n)$ is continuous, strictly decreasing, and since $\sum_{k=1}^{K} c_k(n^*) \geq 1$ and $\lim_{n \to \infty} \sum_{k=1}^{K} c_k(n) = 0$, there exists a unique $n > 0$ such that $\sum_{k=1}^{K} c_k(n) = 1$. For this n, from The-

orem 1 follows that the total budget is conserved for the price $\mathbf{c} = (c_1(n), ..., c_K(n))$. Uniqueness follows from the uniqueness of $c_k(n)$ and the uniqueness of n. ■

Proof [of Theorem 4] The price equations (4) become:

$$\sum_{m=1}^{M} \beta_m \phi_m^k(\mathbf{x}) = c_k \sum_{k=1}^{K} \sum_{m=1}^{M} \beta_m \phi_m^k(\mathbf{x}), \quad \forall k = 1, ..., K.$$

which give the result from Equation (7).

If $\phi_m^k(\mathbf{x}) = \eta h_m^k(\mathbf{x})$, using $\sum_{k=1}^{K} h_m^k(\mathbf{x}) = 1$, the denominator of Equation (7) becomes

$$\sum_{k=1}^{K} \sum_{m=1}^{M} \beta_m \phi_m^k(\mathbf{x}) = \eta \sum_{m=1}^{M} \beta_m \sum_{k=1}^{K} h_m^k(\mathbf{x}) = \eta \sum_{m=1}^{M} \beta_m$$

so

$$c_k = \frac{\eta \sum_{m=1}^{M} \beta_m h_m^k(\mathbf{x})}{\eta \sum_{m=1}^{M} \beta_m} = \sum_m \alpha_m h_m^k(\mathbf{x}), \quad \forall k = 1, ..., K.$$

■

Proof [of Theorem 5] For the current parameters $\gamma = (\gamma_1, ..., \gamma_M) = (\sqrt{\beta_1}, ..., \sqrt{\beta_m})$ and an observation $(\mathbf{x}_i, y_i)$, we have the market price for label y_i:

$$c_{y_i}(\mathbf{x}_i) = \sum_{m=1}^{M} \gamma_m^2 \phi_m^{y_i}(\mathbf{x}_i) / (\sum_{m=1}^{M} \sum_{k=1}^{K} \gamma_m^2 \phi_m^k(\mathbf{x}_i)). \tag{19}$$

So the log-likelihood is

$$L(\gamma) = \frac{1}{N} \sum_{i=1}^{N} \log c_{y_i}(\mathbf{x}_i) = \frac{1}{N} \sum_{i=1}^{N} \log \sum_{m=1}^{M} \gamma_m^2 \phi_m^{y_i}(\mathbf{x}_i) - \frac{1}{N} \sum_{i=1}^{N} \log \sum_{m=1}^{M} \sum_{k=1}^{K} \gamma_m^2 \phi_m^k(\mathbf{x}_i).$$

We obtain the gradient components:

$$\frac{\partial L(\gamma)}{\partial \gamma_j} = \frac{1}{N} \sum_{i=1}^{N} \left(\frac{\gamma_j \phi_j^{y_i}(\mathbf{x}_i)}{\sum_{m=1}^{M} \gamma_m^2 \phi_m^{y_i}(\mathbf{x}_i)} - \frac{\gamma_j \sum_{k=1}^{K} \phi_j^k(\mathbf{x}_i)}{\sum_{m=1}^{M} \sum_{k=1}^{K} \gamma_m^2 \phi_m^k(\mathbf{x}_i)} \right). \tag{20}$$

Then from (19) we have $\sum_{m=1}^{M} \gamma_m^2 \phi_m^{y_i}(\mathbf{x}_i) = B(\mathbf{x}_i) c_{y_i}(\mathbf{x}_i)$. Hence (20) becomes

$$\frac{\partial L(\gamma)}{\partial \gamma_j} = \frac{\gamma_j}{N} \sum_{i=1}^{N} \frac{1}{B(\mathbf{x}_i)} \left(\frac{\phi_j^{y_i}(\mathbf{x}_i)}{c_{y_i}(\mathbf{x}_i)} - \sum_{k=1}^{K} \phi_j^k(\mathbf{x}_i) \right).$$

Write $u_j = \frac{1}{N} \sum_{i=1}^{N} \frac{1}{B(\mathbf{x}_i)} \left(\frac{\phi_j^{y_i}(\mathbf{x}_i)}{c_{y_i}(\mathbf{x}_i)} - \sum_{k=1}^{K} \phi_j^k(\mathbf{x}_i) \right)$, then $\frac{\partial L(\gamma)}{\partial \gamma_j} = \gamma_j u_j$. The batch update (12) is $\beta_j \leftarrow \beta_j + \eta \beta_j u_j$. By taking the square root we get the update in γ

$$\gamma_j \leftarrow \gamma_j \sqrt{1 + \eta u_j} = \gamma_j + \gamma_j (\sqrt{1 + \eta u_j} - 1) = \gamma_j + \gamma_j \frac{\eta u_j}{\sqrt{1 + \eta u_j} + 1} = \gamma'_j.$$

We can write the Taylor expansion:

$$L(\gamma') = L(\gamma) + (\gamma' - \gamma)^T \nabla L(\gamma) + \frac{1}{2}(\gamma' - \gamma)^T H(L)(\zeta)(\gamma' - \gamma)$$

so

$$L(\gamma') = L(\gamma) + \sum_{j=1}^{M} \gamma_j u_j \frac{\eta \gamma_j u_j}{\sqrt{1+\eta u_j}+1} + \eta^2 A(\eta) = L(\gamma) + \eta \sum_{j=1}^{M} \frac{\gamma_j^2 u_j^2}{\sqrt{1+\eta u_j}+1} + \eta^2 A(\eta)$$

where $|A(\eta)|$ is bounded in a neighborhood of 0.

Now assume that $\nabla L(\gamma) \neq 0$, thus $\gamma_j u_j \neq 0$ for some j. Then $\sum_{j=1}^{M} \frac{\gamma_j^2 u_j^2}{\sqrt{1+\eta u_j}+1} > 0$ hence $L(\gamma') > L(\gamma)$ for any η small enough.

Thus as long as $\nabla L(\gamma) \neq 0$ the batch update (12) with any η sufficiently small will increase the likelihood function.

The batch update (12) can be split into N per-observation updates of the form (13). ■

References

K. J. Arrow, R. Forsythe, M. Gorham, R. Hahn, R. Hanson, J. O. Ledyard, S. Levmore, R. Litan, P. Milgrom, and F. D. Nelson. The promise of prediction markets. *Science*, 320(5878):877, 2008.

A. Barbu, M. Suehling, X. Xu, D. Liu, S. Zhou, and D. Comaniciu. Automatic detection and segmentation of lymph nodes from ct data. *IEEE Trans. on Medical Imaging*, 31(2):240–250, 2012.

S. Basu. Investment performance of common stocks in relation to their price-earnings ratios: A test of the efficient market hypothesis. *The Journal of Finance*, 32(3):663–682, 1977.

M. Belkin and P. Niyogi. Semi-supervised learning on Riemannian manifolds. *Machine Learning*, 56(1):209–239, 2004.

A.L. Berger, V.J.D. Pietra, and S.A.D. Pietra. A maximum entropy approach to natural language processing. *Computational linguistics*, 22(1):39–71, 1996.

C. Blake and CJ Merz. UCI repository of machine learning databases [http://www. ics. uci. edu/ mlearn/MLRepository. html], Department of Information and Computer Science. *University of California, Irvine, CA*, 1998.

A. Bordes, S. Ertekin, J. Weston, and L. Bottou. Fast kernel classifiers with online and active learning. *The Journal of Machine Learning Research*, 6:1619, 2005.

L. Breiman. Random forests. *Machine Learning*, 45(1):5–32, 2001.

G. Cauwenberghs and T. Poggio. Incremental and decremental support vector machine learning. In *NIPS*, page 409, 2001.

Y. Chen and J.W. Vaughan. A new understanding of prediction markets via no-regret learning. In *ACM Conf. on Electronic Commerce*, pages 189–198, 2010.

Y. Chen, J. Abernethy, and J.W. Vaughan. An optimization-based framework for automated market-making. *Proceedings of the EC*, 11:5–9, 2011.

C. Chow. On optimum recognition error and reject tradeoff. *IEEE Trans. on Information Theory*, 16(1):41–46, 1970.

B. Cowgill, J. Wolfers, and E. Zitzewitz. Using prediction markets to track information flows: Evidence from Google. *Dartmouth College*, 2008.

J. Demšar. Statistical comparisons of classifiers over multiple data sets. *The Journal of Machine Learning Research*, 7:30, 2006.

A. Elgammal and C.S. Lee. Inferring 3d body pose from silhouettes using activity manifold learning. In *CVPR*, 2004.

E.F. Fama. Efficient capital markets: A review of theory and empirical work. *Journal of Finance*, pages 383–417, 1970.

C. Ferri, P. Flach, and J. Hernández-Orallo. Delegating classifiers. In *ICML*, 2004.

Y. Freund and R.E. Schapire. Experiments with a new boosting algorithm. In *ICML*, pages 148–156, 1996.

J. Friedman, T. Hastie, and R. Tibshirani. Additive logistic regression: a statistical view of boosting. *Annals of Statistics*, 28(2):337–407, 2000.

J.H. Friedman and B.E. Popescu. Predictive learning via rule ensembles. *Ann. Appl. Stat.*, 2(3): 916–954, 2008.

S. Gjerstad and M.C. Hall. Risk aversion, beliefs, and prediction market equilibrium. *Economic Science Laboratory, University of Arizona*, 2005.

J. Kivinen, AJ Smola, and RC Williamson. Online learning with kernels. *IEEE Trans. on Signal Processing*, 52:2165–2176, 2004.

B. Kulis and P.L. Bartlett. Implicit online learning. In *ICML*, 2010.

N. Lay and A. Barbu. Supervised aggregation of classifiers using artificial prediction markets. In *ICML*, 2010.

B.G. Malkiel. The efficient market hypothesis and its critics. *The Journal of Economic Perspectives*, 17(1):59–82, 2003.

W. R. Mann. Mean value methods in iteration. *Proc. Amer. Math. Soc.*, 4:506–510, 1953.

C.F. Manski. Interpreting the predictions of prediction markets. *Economics Letters*, 91(3):425–429, 2006.

J. Perols, K. Chari, and M. Agrawal. Information market-based decision fusion. *Management Science*, 55(5):827–842, 2009.

C.R. Plott, J. Wit, and W.C. Yang. Parimutuel betting markets as information aggregation devices: Experimental results. *Economic Theory*, 22(2):311–351, 2003.

P.M. Polgreen, F.D. Nelson, and G.R. Neumann. Use of prediction markets to forecast infectious disease activity. *Clinical Infectious Diseases*, 44(2):272–279, 2006.

C. Polk, R. Hanson, J. Ledyard, and T. Ishikida. The policy analysis market: an electronic commerce application of a combinatorial information market. In *ACM Conf. on Electronic Commerce*, pages 272–273, 2003.

A. Ratnaparkhi et al. A maximum entropy model for part-of-speech tagging. In *Conf. on Empirical Methods in Natural Language Processing*, volume 1, pages 133–142, 1996.

L.K. Saul and S.T. Roweis. Think globally, fit locally: Unsupervised learning of low dimensional manifolds. *The Journal of Machine Learning Research*, 4:119–155, 2003.

R.E. Schapire. The boosting approach to machine learning: An overview. *Lect. Notes in Statistics*, pages 149–172, 2003.

A. Storkey. Machine learning markets. *AISTATS*, 2011.

A. Storkey, J. Millin, and K. Geras. Isoelastic agents and wealth updates in machine learning markets. *ICML*, 2012.

F. Tortorella. Reducing the classification cost of support vector classifiers through an ROC-based reject rule. *Pattern Analysis & Applications*, 7(2):128–143, 2004.

J. Wolfers and E. Zitzewitz. Prediction markets. *Journal of Economic Perspectives*, pages 107–126, 2004.

S.C. Zhu, Y. Wu, and D. Mumford. Filters, random fields and maximum entropy (FRAME): Towards a unified theory for texture modeling. *International Journal of Computer Vision*, 27(2): 107–126, 1998.

Journal of Machine Learning Research 13 (2012) 2205-2231 Submitted 10/11; Revised 5/12; Published 7/12

Sign Language Recognition using Sub-Units

Helen Cooper H.M.COOPER@SURREY.AC.UK
Eng-Jon Ong E.ONG@SURREY.AC.UK
Nicolas Pugeault N.PUGEAULT@SURREY.AC.UK
Richard Bowden R.BOWDEN@SURREY.AC.UK
Centre for Vision Speech and Signal Processing
University of Surrey
Guildford. GU2 9PY UK

Editors: Isabelle Guyon and Vassilis Athitsos

Abstract

This paper discusses sign language recognition using linguistic sub-units. It presents three types of sub-units for consideration; those learnt from appearance data as well as those inferred from both 2D or 3D tracking data. These sub-units are then combined using a sign level classifier; here, two options are presented. The first uses Markov Models to encode the temporal changes between sub-units. The second makes use of Sequential Pattern Boosting to apply discriminative feature selection at the same time as encoding temporal information. This approach is more robust to noise and performs well in signer independent tests, improving results from the 54% achieved by the Markov Chains to 76%.

Keywords: sign language recognition, sequential pattern boosting, depth cameras, sub-units, signer independence, data set

1. Introduction

This paper presents several approaches to sub-unit based Sign Language Recognition (SLR) culminating in a real time Kinect^{TM}demonstration system. SLR is a non-trivial task. Sign Languages (SLs) are made up of thousands of different signs; each differing from the other by minor changes in motion, handshape, location or Non-Manual Featuress (NMFs). While Gesture Recognition (GR) solutions often build a classifier per gesture, this approach soon becomes intractable when recognising large lexicons of signs, for even the relatively straightforward task of citation-form, dictionary look-up. Speech recognition was faced with the same problem; the emergent solution was to recognise the subcomponents (phonemes), then combine them into words using Hidden Markov Models (HMMs). Sub-unit based SLR uses a similar two stage recognition system, in the first stage, sign linguistic sub-units are identified. In the second stage, these sub-units are combined together to create a sign level classifier.

Linguists also describe SLs in terms of component sub-units; by using these sub-units, not only can larger sign lexicons be handled efficiently, allowing demonstration on databases of nearly 1000 signs, but they are also more robust to the natural variations of signs, which occur on both an inter and an intra signer basis. This makes them suited to real-time signer independent recognition as described later. This paper will focus on 4 main sub-unit categories based on *HandShape*, *Location*, *Motion* and *Hand-Arrangement*. There are several methods for labelling these sub-units and this

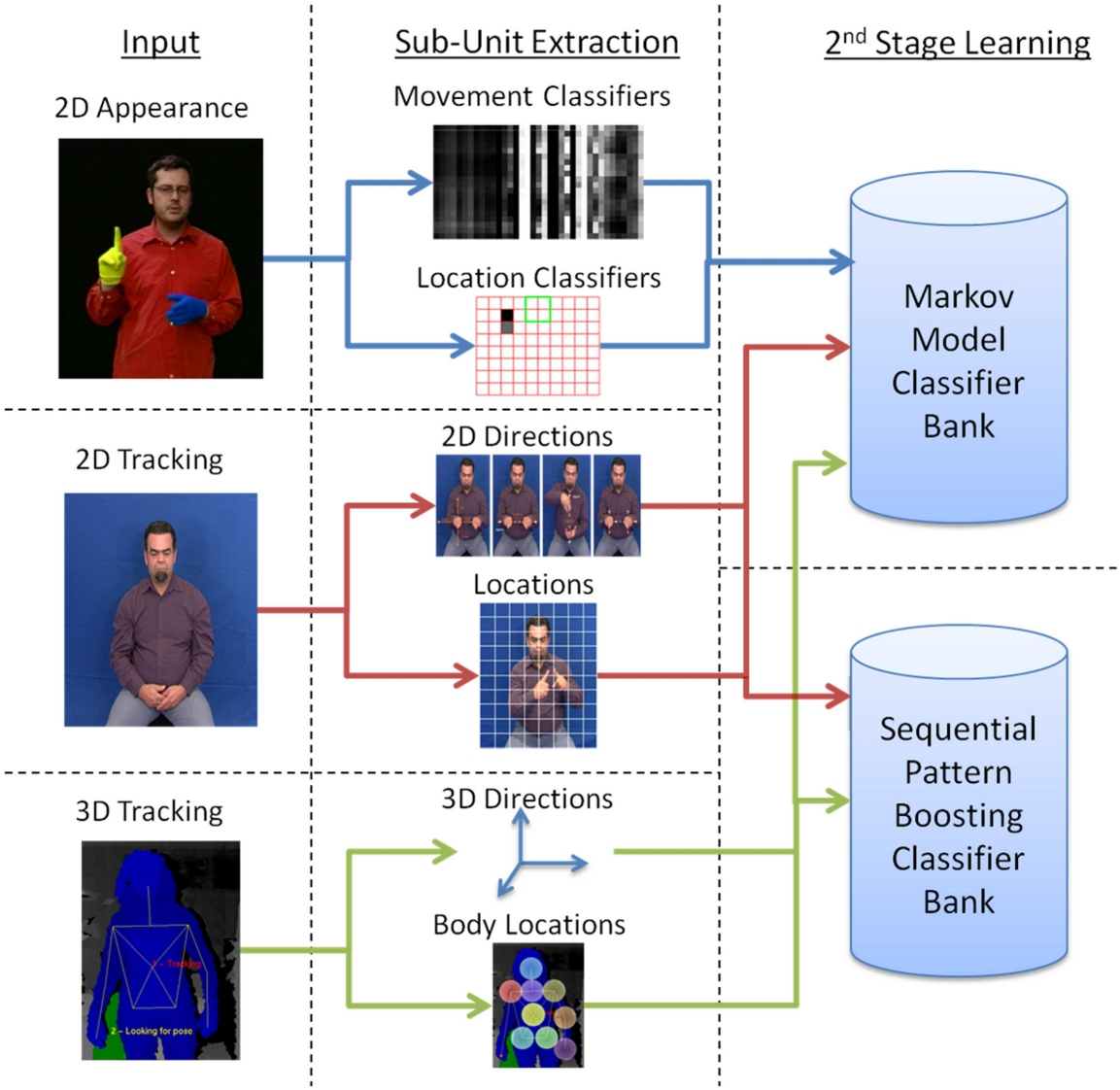


Figure 1: Overview of the 3 types of sub-units extracted and the 2 different sign level classifiers used.

work builds on both the Ha, Tab, Sig, Dez system from the BSL dictionary (British Deaf Association, 1992) and The Hamburg Notation System (HamNoSys), which has continued to develop over recent years to allow more detailed description of signs from numerous SLs (Hanke and Schmaling, 2004).

This paper presents a comparison of sub-unit approaches, focussing on the advantages and disadvantages of each. Also presented is a newly released Kinect data set, containing multiple users performing signs in various environments. There are three different types of sub-units considered; those based on appearance data alone, those which use 2D tracking data with appearance based handshapes and those which use 3D tracking data produced by a Kinect^{TM}sensor. Each of these

three sub-unit types is tested with a Markov model approach to combine sub-units into sign level classifiers. A further experiment is performed to investigate the discriminative learning power of Sequential Pattern (SP) Boosting for signer independent recognition. An overview is shown in Figure 1.

2. Background

The concept of using sub-units for SLR is not novel. Kim and Waldron (1993) were among the first adopters, they worked on a limited vocabulary of 13-16 signs, using data gloves to get accurate input information. Using the work of Stokoe (1960) as a base, and their previous work in telecommunications (Waldron and Simon, 1989), they noted the need to break signs into their component sub-units for efficiency. They continued this throughout the remainder of their work, where they used phonemic recognition modules for hand shape, orientation, position and movement recognition (Waldron and Kim, 1994). They made note of the dependency of position, orientation and motion on one another and removed the motion aspect allowing the other sub-units to compensate (on a small vocabulary, a dynamic representation of position is equivalent to motion) (Waldron and Kim, 1995).

The early work of Vogler and Metaxas (1997) borrowed heavily from the studies of sign language by Liddell and Johnson (1989), splitting signs into motion and pause sections. Their later work (Vogler and Metaxas, 1999), used parallel HMMs on both hand shape and motion sub-units, similar to those proposed by the linguist Stokoe (1960). Kadir et al. (2004) took this further by combining head, hand and torso positions, as well as hand shape, to create a system based on hard coded sub-unit classifiers that could be trained on as little as a single example.

Alternative methods have looked at data driven approaches to defining sub-units. Yin et al. (2009) used an accelerometer glove to gather information about a sign, they then applied discriminative feature extraction and 'similar state tying' algorithms, to decide sub-unit level segmentation of the data. Whereas Kong and Ranganath (2008) and Han et al. (2009) looked at automatic segmentation of sign motion into sub-units, using discontinuities in the trajectory and acceleration to indicate where segments begin and end. These were then clustered into a code book of possible exemplar trajectories using either Dynamic Time Warping (DTW) distance measures Han et al. or Principal Component Analysis (PCA) Kong and Ranganath.

Traditional sign recognition systems use tracking and data driven approaches (Han et al., 2009; Yin et al., 2009). However, there is an increasing body of research that suggests using linguistically derived features can offer superior performance. Cooper and Bowden (2010) learnt linguistic sub-units from hand annotated data which they combined with Markov models to create sign level classifiers, while Pitsikalis et al. (2011) presented a method which incorporated phonetic transcriptions into sub-unit based statistical models. They used HamNoSys annotations combined with the Postures, Detentions, Transitions, Steady Shifts (PDTS) phonetic model to break the signs and annotations into labelled sub-units. These were used to construct statistical sub-unit models which they combined via HMMs.

The frequent requirement of tracked data means that the Kinect^{TM}device has offered the sign recognition community a short-cut to real-time performance. In the relatively short time since its release, several proof of concept demonstrations have emerged. Ershaed et al. (2011) have focussed on Arabic sign language and have created a system which recognises isolated signs. They present a system working for 4 signs and recognise some close up handshape information (Ershaed et al.,

2011). At ESIEA they have been using Fast Artificial Neural Networks to train a system which recognises two French signs (Wassner, 2011). This small vocabulary is a proof of concept but it is unlikely to be scalable to larger lexicons. It is for this reason that many sign recognition approaches use variants of HMMs (Starner and Pentland, 1997; Vogler and Metaxas, 1999; Kadir et al., 2004; Cooper and Bowden, 2007). One of the first videos to be uploaded to the web came from Zafrulla et al. (2011) and was an extension of their previous CopyCat game for deaf children (Zafrulla et al., 2010). The original system uses coloured gloves and accelerometers to track the hands. By tracking with a Kinect™, they use solely the upper part of the torso and normalise the skeleton according to arm length (Zafrulla et al., 2011). They have an internal data set containing 6 signs; 2 subject signs, 2 prepositions and 2 object signs. The signs are used in 4 sentences (subject, preposition, object) and they have recorded 20 examples of each. Their data set is currently single signer, making the system signer dependent, while they list under further work that signer independence would be desirable. By using a cross validated system they train HMMs (Via the Georgia Tech Gesture Toolkit Lyons et al., 2007) to recognise the signs. They perform 3 types of tests, those with full grammar constraints achieving 100%, those where the number of signs is known achieving 99.98% and those with no restrictions achieving 98.8%.

2.1 Linguistics

Sign language sub-units can be likened to speech phonemes, but while a spoken language such as English has only 40-50 phonemes (Shoup, 1980), SLs have many more. For example, *The Dictionary of British Sign Language/English* (British Deaf Association, 1992) lists 57 'Dez' (*HandShape*), 36 'Tab' (*Location*), 8 'Ha' (*Hand-Arrangement*), 28 'Sig' (*Motion*) (plus 4 modifiers, for example, short and repeated) and there are two sets of 6 'ori' (*Orientation*), one for the fingers and one for the palm.

HamNoSys uses a more combinatorial approach to sub-units. For instance, it lists 12 basic handshapes which can be augmented using finger bending, thumb position and openeness characteristics to create a single *HandShape* sub-unit. These handshapes are then combined with palm and finger orientations to describe the final hand posture. *Motion* sub-units can be simple linear directions, known as 'Path Movements' these can also be modified by curves, wiggles or zigzags. *Motion* sub-units can also be modified by locations, for example, move from A to B with a curved motion or move down beside the nose.

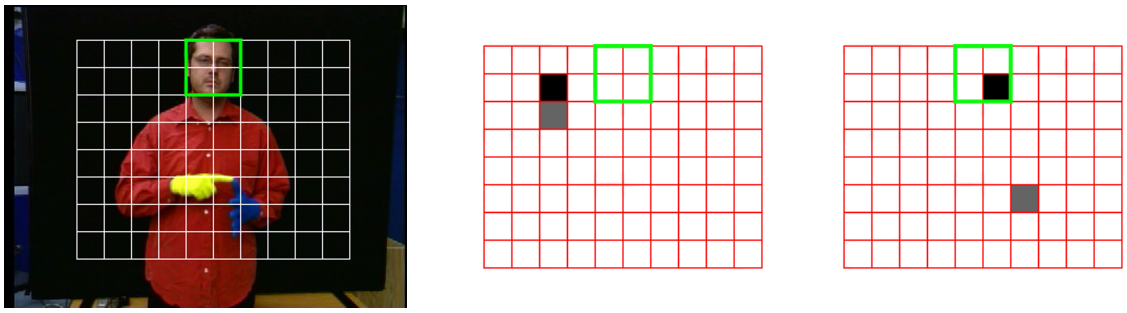
In addition, whereas spoken phonemes are broadly sequential, sign sub-units are parallel, with some sequential elements added where required. This means that each of the 57 British Sign Language (BSL) *HandShape* options can (theoretically) be in any one of the 36 BSL *Orientation* combinations. In practice, due to the physical constraints of the human body, only a subset of comfortable combinations occur, yet this subset is still considerable.

An advantage of the parallel nature of sub-units, is that they can be recognised independently using different classifiers, then combined at the word level. The reason this is advantageous is that *Location* classifiers need to be spatially variant, since they describe where a sign happens. *Hand-Arrangement* should be spatially invariant but not rotationally variant, since they describe positional relationships between the hands. While *Motion* are a mixture of spatially, temporally, rotationally and scale variant sub-units since they describe types of motion which can be as generic as 'hands move apart' or more specific such as 'hand moves left'. Therefore each type of sub-unit can be recognised by classifiers incorporating the correct combination of invariances. This paper presents

three methods for extracting sub-units; learnt appearance based (Section 3), hard coded 2D tracking based (Section 4) and hard coded 3D tracking based (Section 5).

3. Learning Appearance Based Sub-units

The work in this section learns a subset of each type of sub-unit using AdaBoost from hand labelled data. As has been previously discussed, not all types of sub-units can be detected using the same type of classifier. For *Location* sub-units, there needs to be correlation between where the motion is happening and where the person is; to this end spatial grid features centred around the face of the signer are employed. For *Motion* sub-units, the salient information is what type of motion is occurring, often regardless of its position, orientation or size. This is approached by extracting moment features and using Binary Patterns (BPs) and additive classifiers based on their changes over time. *Hand-Arrangement* sub-units look at where the hands are in relation to each other, so these are only relevant for bi-manual signs. This is done using the same moment features as for *Motion* but this time over a single frame, as there is no temporal context required. All of these sub-unit level classifiers are learnt using AdaBoost (Freund and Schapire, 1995). The features used in this section require segmentation of the hands and knowledge of where the face is. The Viola Jones face detector (Viola and Jones, 2001) is used to locate the face. Skin segmentation could be used to segment the hands, but since sub-unit labels are required this work uses the data set from the work of Kadir et al. (2004) for which there is an in-house set of sub-unit labels for a portion of the data. This data set was created using a gloved signer and as such a colour segmentation algorithm is used in place of skin segmentation.



(a) The grid applied over the signer

(b) On Right Shoulder

(c) Lower Face/Chin

Figure 2: Grid features for two stage classification. (a) shows an example of the grid produced from the face dimensions while (b) and (c) show grid features chosen by boosting for two of the 18 *Location* sub-units. The highlighted box shows the face location and the first and second features chosen, are shown in black and grey respectively.

3.1 *Location* Features

In order that the sign can be localised in relation to the signer, a grid is applied to the image, dependent upon the position and scale of the face detection. Each cell in the grid is a quarter of the face size and the grid is 10 rectangles wide by 8 deep, as shown in Figure 2a. These values are based on the signing space of the signer. However, in this case, the grid does not extend beyond the top of the signers head since the data set does not contain any signs which use that area. The segmented frame is quantised into this grid and a cell fires if over 50% of its pixels are made up of glove/skin. This is shown in Equation 1 where R_{wc} is the weak classifier response and $\Lambda_{skin}(x,y)$ is the likelihood that a pixel contains skin. f is the face height and all the grid values are relative to this dimension.

$$R_{wc} = \begin{cases} 1 \text{ if } \frac{f^2}{8} < \sum_{i=x_1}^{x_2} \sum_{j=y_1}^{y_2} (\Lambda_{skin}(i,j) > 0), \\ 0 \text{ otherwise.} \end{cases}$$

Where x_1, y_1, x_2, y_2 are given by

$$\forall G_x, \forall G_y \begin{cases} x_1 = G_x f, \\ x_2 = (G_x + 0.5) f, \\ y_1 = G_y f, \\ y_2 = (G_y + 0.5) f, \end{cases}$$
$$\text{given } G_x = \{-2.5, -2, -1.5 \ldots 2\},$$
$$G_y = \{-4, -3.5, -3 \ldots 0\}. \quad (1)$$

For each of the *Location* sub-units, a classifier was built via AdaBoost to combine cells which fire for each particular sub-unit, examples of these classifiers are shown in Figures 2b and (c). Note how the first cell to be picked by the boosting (shown in black) is the one directly related to the area indicated by the sub-unit label. The second cell chosen by boosting either adds to this location information, as in Figure 2b, or comments on the stationary, non-dominant hand, as in Figure 2c.

Some of the sub-units types contain values which are not mutually exclusive, this needs to be taken into account when labelling and using sub-unit data. The BSL dictionary (British Deaf Association, 1992) lists several *Location* sub-units which overlap with each other, such as face and mouth or nose. Using boosting to train classifiers requires positive and negative examples. For best results, examples should not be contaminated, that is, the positive set should not contain negatives and the negative set should not contain positives. Trying to distinguish between an area and its sub-areas can prove futile, for example, the mouth is also on the face and therefore there are likely to be false negatives in the training set when training face against mouth. The second stage, sign-level classification does not require the sub-unit classifier responses to be mutually exclusive. As such a hierarchy can be created of *Location* areas and their sub-areas. This hierarchy is shown in Figure 3; a classifier is trained for each node of the tree, using examples which belong to it, or its children, as positive data. Examples which do not belong to it, its parent or its child nodes provide negative data.

This eliminates false negatives from the data set and avoids confusion. In Figure 3 the ringed nodes show the sub-units for which there exist examples. Examples are labelled according to this

hierarchy, for example, face, face_lower or face_lower_mouth which makes finding children and parents easier by using simple string comparisons.

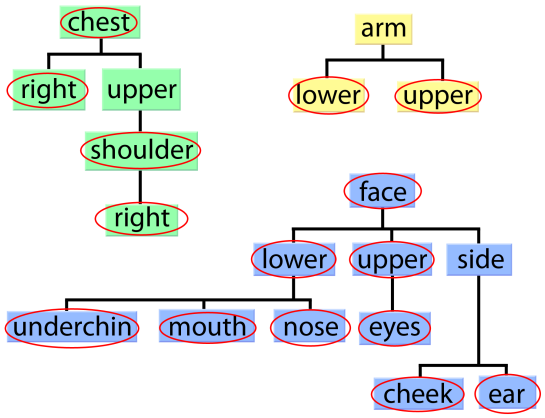


Figure 3: The three *Location* sub-unit trees used for classification. There are three separate trees, based around areas of the body which do not overlap. Areas on the leaves of the tree are sub-areas of their parent nodes. The ringed labels indicate that there are exact examples of that type in the data set.

3.2 *Motion* and *Hand-Arrangement* Moment Feature Vectors

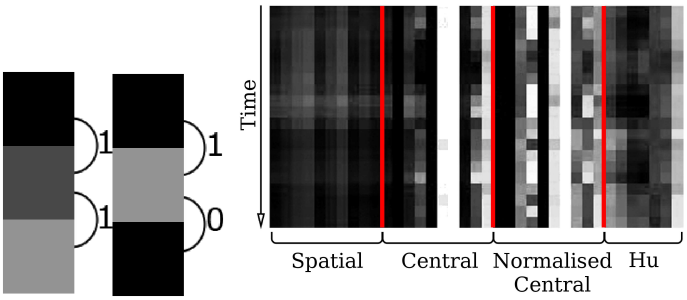
For *Hand-Arrangement* and *Motion*, information regarding the arrangement and motion of the hands is required. Moments offer a way of encoding the shapes in an image; if vectors of moment values per frame are concatenated, then they can encode the change in shape of an image over time.

There are several different types of moments which can be calculated, each of them displaying different properties. Four types were chosen to form a feature vector, $\mathbf{m}$: spatial, m_{ab}, central, μ_{ab}, normalised central, $\bar{\mu}_{ab}$ and the Hu set of invariant moments (Hu, 1962) $\mathcal{H}_1$-$\mathcal{H}_7$. The order of a moment is defined as $a+b$. This work uses all moments, central moments and normalised central moments up to the 3rd order, 10 per type, (00, 01, 10, 11, 20, 02, 12, 21, 30, 03). Finally, the Hu set of invariant moments are considered, there are 7 of these moments and they are created by combining the normalised central moments, see Hu (1962) for full details, they offer invariance to scale, translation, rotation and skew. This gives a 37 dimensional feature vector, with a wide range of different properties.

$$R_{wc} = \begin{cases} 1 \text{ if } \mathcal{T}_{wc} < \mathbf{M}_{i,t}, \\ 0 \text{ otherwise.} \end{cases} \tag{2}$$

Since spatial moments are not invariant to translation and scale, there needs to be a common point of origin and similar scale across examples. To this end, the spatial moments are treated in a similar

way to the spatial features in Section 3.1, by centring and scaling the image about the face of the signer before computation. For training *Hand-Arrangement*, this vector is used to boost a set of thresholds for individual moments, $\mathbf{m}_i$ on a given frame t, Equation 2. For *Motion*, temporal information needs to be included. Therefore the video clips are described by a stack of these vectors, $\mathbf{M}$, like a series of 2D arrays, as shown in Figure 4(a) where the horizontal vectors of moments are concatenated vertically, the lighter the colour, the higher the value of the moment on that frame.



(a) BP example (b) Concatenated Moment Vector

Figure 4: Moment vectors and Binary Patterns for two stage classification. (b) A pictorial description of moment vectors (normalised along each moment type for a selection of examples), the lighter the colour the larger the moment value. (a) BP, working from top to bottom an increase in gradient is depicted by a 1 and a decrease or no change by a 0.

3.3 *Motion* Binary Patterns and Additive Classifiers

As has been previously discussed, the *Motion* classifiers are looking for changes in the moments over time. By concatenating feature vectors temporally as shown in Figure 4(b), these spatio-temporal changes can be found. Component values can either increase, decrease or remain the same, from one frame to the next. If an increase is described as a 1 and a decrease or no change is described as a 0 then a BP can be used to encode a series of increases/decreases. A temporal vector is said to match the given BP if every '1' accompanies an increase between concurrent frames and every '0' a decrease/'no change'. This is shown in Equation 3 where $\mathbf{M}_{i,t}$ is the value of the component, $\mathbf{M}_i$, at time t and $\mathbf{bp}_t$ is the value of the BP at frame t.

$$\begin{aligned} R_{wc} &= ||\max_{\forall t}(BP(\mathbf{M}_{i,t}))| - 1|, \\ BP(\mathbf{M}_{i,t}) &= \mathbf{bp}_t - d(\mathbf{M}_{i,t}, \mathbf{M}_{i,t+1}), \\ d(\mathbf{M}_{i,t}, \mathbf{M}_{i,t+1}) &= \begin{cases} 0 \text{ if } \mathbf{M}_{i,t} \leq \mathbf{M}_{i,t+1}, \\ 1 \text{ otherwise.} \end{cases} \end{aligned} \tag{3}$$

See Figure 5 for an example where feature vector A makes the weak classifier fire, whereas feature vector B fails, due to the ringed gradients being incompatible.

Discarding all magnitude information would possibly remove salient information. To retain this information, boosting is also given the option of using additive classifiers. These look at the average

magnitude of a component over time. The weak classifiers are created by applying a threshold, $\mathcal{T}_{wc}$, to the summation of a given component, over several frames. This threshold is optimised across the training data during the boosting phase. For an additive classifier of size T, over component $\mathbf{m}_i$, the response of the classifier, R_{wc}, can be described as in Equation 4.

$$R_{wc} = \begin{cases} 1 \text{ if } \mathcal{T}_{wc} \leq \sum_{t=0}^{T} \mathbf{M}_{i,t}, \\ 0 \text{ otherwise.} \end{cases} \tag{4}$$

Boosting is given all possible combinations of BPs, acting on each of the possible components. The BPs are limited in size, being between 2 and 5 changes (3 - 6 frames) long. The additive features are also applied to all the possible components, but the lengths permitted are between 1 and 26 frames, the longest mean length of *Motion* sub-units. Both sets of weak classifiers can be temporally offset from the beginning of an example, by any distance up to the maximum distance of 26 frames.

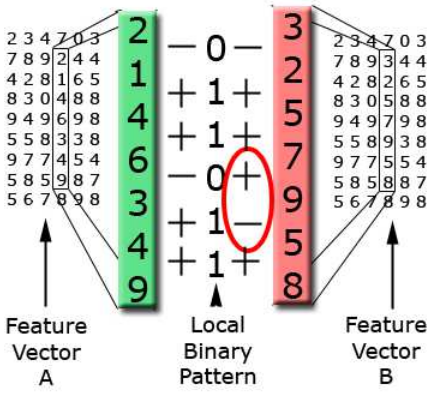
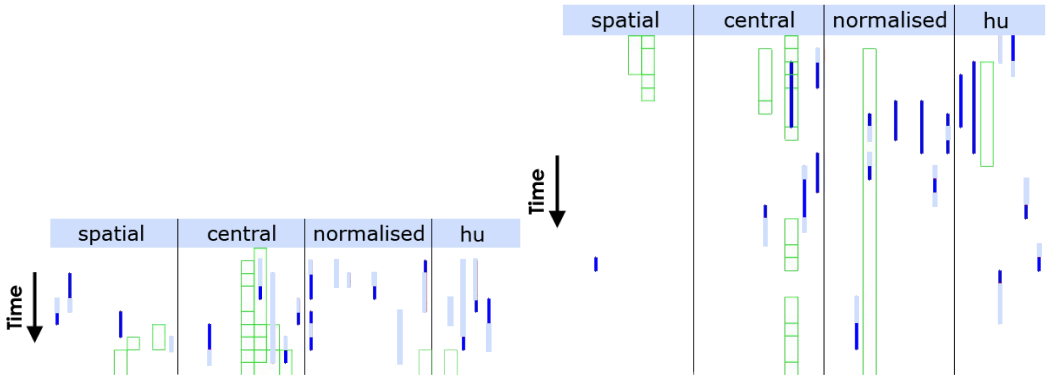


Figure 5: An example of a BP being used to classify two examples. A comparison is made between the elements of the weak classifiers BP and the temporal vector of the component being assessed. If every '1' in the BP aligns with an increase in the component and every '0' aligns with a decrease or 'no change' then the component vector is said to match (e.g., case A). However if there are inconsistencies as ringed in case B then the weak classifier will not fire.

Examples of the classifiers learnt are shown in Figure 6, additive classifiers are shown by boxes, increasing BPs are shown by pale lines and decreasing ones by dark lines. When looking at a sub-unit such as 'hands move apart' (Figure 6a), the majority of the BP classifiers show increasing moments, which is what would be expected, as the eccentricity of the moments is likely to increase as the hands move apart. Conversely, for 'hands move together' (Figure 6b), most of the BPs are decreasing.

Since some *Motion* sub-units occur more quickly than others, the boosted classifiers are not all constrained to being equal in temporal length. Instead, an optimal length is chosen over the training set for each individual sub-unit. Several different length classifiers are boosted starting at 6 frames



(a) hands_move_apart (b) Hands_move_together

Figure 6: Boosted temporal moments BP and additive *Motion* classifiers. The moment vectors are stacked one frame ahead of another. The boxes show where an additive classifier has been chosen, a dark line shows a decreasing moment value and a pale line an increasing value.

long, increasing in steps of 2 and finishing at 26 frames long. Training classification results are then found for each sub-unit and the best length chosen to create a final set of classifiers, of various lengths suited to the sub-units being classified.

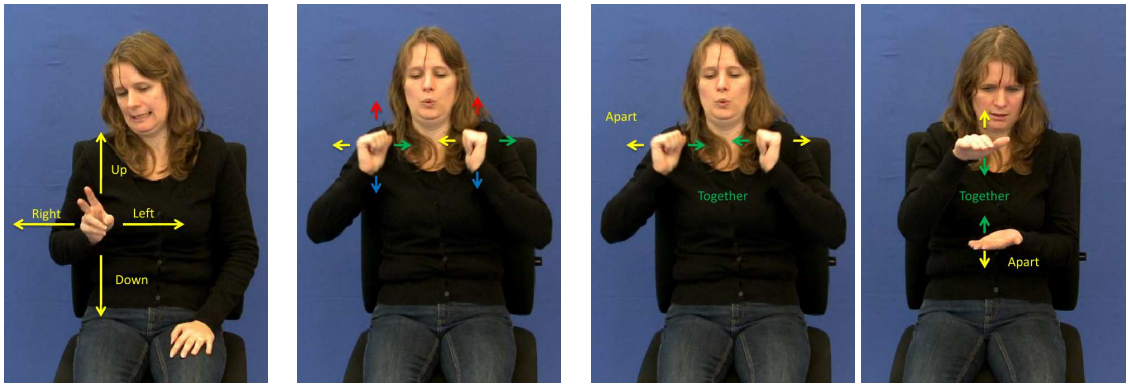
4. 2D Tracking Based Sub-Units

Unfortunately, since the learnt, appearance based, sub-units require expertly annotated data they are limited to data sets with this annotation. An alternative to appearance based features is given by tracking. While tracking errors can propagate to create sub-unit errors, the hand trajectories offer significant information which can aid recognition. With the advances of tracking systems and the real-time solution introduced by the Kinect™, tracking is fast becoming an option for real-time, robust recognition of sign language. This section works with hand and head trajectories, extracted from videos by the work outlined by Roussos et al. (2010). The tracking information is used to extract *Motion* and *Location* information. *HandShape* information is extracted via Histograms of Gradients (HOGs) on hand image patches and learnt from labels using random forests. The labels are taken from the linguistic representations of Sign Gesture Mark-up Language (SiGML) (Elliott et al., 2001) or HamNoSys (Hanke and Schmaling, 2004).[1]

4.1 *Motion* Features

In order to link the x,y co-ordinates obtained from the tracking to the abstract concepts used by sign linguists, rules are employed to extract HamNoSys based information from the trajectories. The approximate size of the head is used as a heuristic to discard ambient motion (that less than 0.25 the head size) and the type of motion occurring is derived directly from deterministic rules on the

1. Note that conversion between the two forms is possible. However while HamNoSys is usually presented as a font for linguistic use, SiGML is more suited to automatic processing.



(a) Single handed (b) Bimanual: Synchronous (c) Bimanual: Together/Apart

Figure 7: Motions detected from tracking

x and y co-ordinates of the hand position. The types of motions encoded are shown in Figure 7, the single handed motions are available for both hands and the dual handed motions are orientation independent so as to match linguistic concepts.

4.2 *Location* Features

Similarly the x and y co-ordinates of the sign location need to be described relative to the signer rather than in absolute pixel positions. This is achieved via quantisation of the values into a code-book based on the signer's head position and scale in the image. For any given hand position (x_h, y_h) the quantised version (x'_h, y'_h) is achieved using the quantisation rules shown in Equation 5, where (x_f, y_f) is the face position and (w_f, h_f) is the face size.

$$\begin{aligned} x' &= (x_h - x_f)/w_f, \\ y' &= (y_h - y_f)/h_f. \end{aligned} \tag{5}$$

Due to the limited size of a natural signing space, this gives values in the range of $y' \in \{0..10\}$ and $x' \in \{0..8\}$ which can be expressed as a binary feature vector of size 36, where the x and y positions of the hands are quantised independently.

4.3 *HandShape* Features

While just the motion and location of the signs can be used for recognition of many examples, it has been shown that adding the handshape can give significant improvement (Kadir et al., 2004). HOG descriptors have proven efficient for sign language hand shape recognition (Buehler et al., 2009) and these are employed as the base feature unit. In each frame, the signer's dominant hand is segmented using the x,y position and a skin model. These image patches are rotated to their principal axis and scaled to a square, 256 pixels in size. Examples of these image patches are shown in Figure 8 beside the frame from which they have been extracted. HOGs are calculated over these squares at a cell size of 32 pixels square with 9 orientation bins and with 2x2 overlapping blocks, these are also shown in Figure 8. This gives a feature vector of 1764 histogram bins which describes the appearance of a hand.

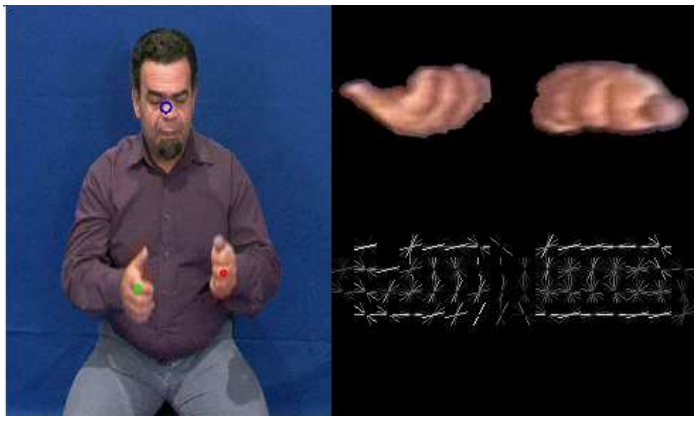


Figure 8: Example HOGs extracted from a frame

4.4 *HandShape* Classifiers

This work focusses on just the 12 basic handshapes, building multi-modal classifiers to account for the different orientations. A list of these handshapes is shown in Figure 9.

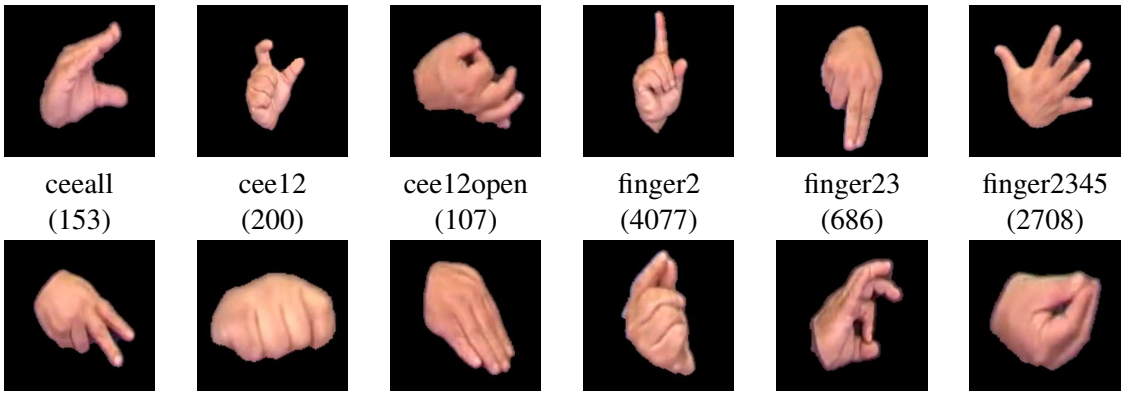


Figure 9: The base handshapes (Number of occurrences in the data set)

Unfortunately, linguists annotating sign do so only at the *sign* level while most sub-units occur for only *part* of a sign. Also, not only do handshapes change throughout the sign, they are made more difficult to recognise due to motion blur. Using the motion of the hands, the sign can be split into its component parts (as in Pitsikalis et al., 2011), that are then aligned with the sign annotations. These annotations are in HamNoSys and have been prepared by trained experts, they include the sign breakdown but not the temporal alignment. The frames most likely to contain a static handshape (i.e., those with limited or no motion) are extracted for training.

Note that, as shown in Figure 10, a single SiGML class (in this case 'finger2') may contain examples which vary greatly in appearance, making visual classification an extremely difficult task.

Figure 10: A variety of examples for the HamNoSys/SiGML class 'finger2'.

The extracted hand shapes are classified using a multi-class random forest. Random forests were proposed by Amit and Geman (1997) and Breiman (2001). They have been shown to yield good performance on a variety of classification and regression problems, and can be trained efficiently in a parallel manner, allowing training on large feature vectors and data sets. In this system, the forest is trained from automatically extracted samples of all 12 handshapes in the data set, shown in Figure 9. Since signs may have multiple handshapes or several instances of the same handshape, the total occurrences are greater than the number of signs, however they are not equally distributed between the handshape classes. The large disparities in the number of examples between classes (see Figure 9) may bias the learning, therefore the training set is rebalanced before learning by selecting 1,000 random samples for each class, forming a new balanced data set. The forest used consists of $N = 100$ multi-class decision trees T_i, each of which is trained on a random subset of the training data. Each tree node splits the feature space in two by applying a threshold on one dimension of the feature vector. This dimension (chosen from a random subset) and the threshold value are chosen to yield the largest reduction in entropy in the class distribution. This recursive partitioning of the data set continues until a node contains a subset of examples that belong to one single class, or if the tree reaches a maximal depth (set to 10). Each leaf is then labelled according to the mode of the contained samples. As a result, the forest yields a probability distribution over all classes, where the likelihood for each class is the proportion of trees that voted for this class. Formally, the confidence that feature vector x describes the handshape c is given by:

$$p[c] = \frac{1}{N} \sum_{i<N} \delta_c(T_i(x)),$$

where N is the number of trees in the forest, $T_i(x)$ is the leaf of the ith tree T_i into which x falls, and $\delta_c(a)$ is the Kronecker delta function ($\delta_c(a) = 1$ iff. $c = a$, $\delta_c(a) = 0$ otherwise).

The performance of this hand shape classification on the test set is recorded on Table 1, where each row corresponds to a shape, and each column corresponds to a predicted class (empty cells signify zero). Lower performance is achieved for classes that are more frequent in the data set. The more frequently a handshape occurs in the data set the more orientations it is likely to be used in. This in turn makes the appearance of the class highly variable; see, for example, Figure 10 for the case of 'finger2'—the worst performing case. Also noted is the high confusion between 'finger2' and 'fist' most likely due to the similarity of these classes when the signer is pointing to themselves.

The handshape classifiers are evaluated for the right hand only during frames when it is not in motion. The sign recognition system is evaluated using two different encodings for the detected hand shapes. As will be described in Section 6, the next stage classifier requires inputs in the form of binary feature vectors. Two types of 12 bit binary feature vector can be produced from the classifier results. The first method applies a strict Winner Takes All (WTA) on the multi-class forest's response: the class with the highest probability is set to one, and the others to zero. For

handshape	predictions											
flat	**0.35**	0.19	0.09	0.03	0.08	0.06	0.03	0.06	0.06	0.01	0.03	0.01
fist	0.03	**0.69**	0.02	0.04	0.11	0.05		0.02	0.03			0.02
finger2345	0.16	0.19	**0.36**	0.02	0.03	0.05		0.06	0.02	0.03	0.06	0.01
finger2	0.02	0.33	0.07	**0.31**	0.11	0.05	0.02	0.03	0.02		0.04	
pinchall	0.03	0.09	0.04	0.01	**0.65**	0.11	0.01	0.01				0.04
pinch12	0.02	0.20	0.01	0.02	0.13	**0.56**	0.01		0.01		0.01	0.02
finger23	0.05	0.17	0.04	0.02	0.05	0.04	**0.54**	0.01			0.07	0.01
pinch12open	0.03	0.12	0.07	0.01	0.15	0.04	0.01	**0.56**				0.01
cee12	0.01	0.05	0.01	0.03	0.04			0.01	**0.82**		0.01	
cee12open					0.01					**0.99**		
finger23spread	0.01	0.15	0.02		0.06	0.01	0.05	0.02			**0.65**	
ceeall	0.01	0.08	0.03		0.08	0.01	0.02	0.01			0.01	**0.77**

Table 1: Confusion matrix of the handshape recognition, for all 12 classes.

every non-motion frame, the vector contains a true value in the highest scoring class. The second method applies a fixed threshold ($\tau = 0.25$) on the confidences provided by the classifier for each of the 12 handshapes classes. Handshapes that have a confidence above threshold ($p[c] > \tau$) are set to one, and the others to zero. This soft approach carries the double advantage that a) the feature vector may encode the ambiguity between handshapes, which may itself carry information, and b) may contain only zeros if confidences in all classes are small.

5. 3D Tracking Based Sub-Units

With the availability of the Kinect™, real-time tracking in 3D is now a realistic option. Due to this, this final sub-unit section expands on the previous tracking sub-units to work in 3D. The tracking is obtained using the OpenNI framework (Ope, 2010) with the PrimeSense tracker (Pri, 2010). Two types of features are extracted, those encoding the *Motion* and *Location* of the sign being performed.

5.1 *Motion* Features

Again, the focus is on linear motion directions, as with the sub-units described in Section 4.1, but this time with the z axis included. Specifically, individual hand motions in the x plane (left and right), the y plane (up and down) and the z plane (towards and away from the signer). This is augmented by the bi-manual classifiers for 'hands move together', 'hands move apart' and 'hands move in sync', again, these are all now assessed in 3D. The approximate size of the head is used as a heuristic to discard ambient motion (that less than 0.25 the head size) and the type of motion occurring is derived directly from deterministic rules on the x,y,z co-ordinates of the hand position. The resulting feature vector is a binary representation of the found linguistic values. The list of 17 motion features extracted is shown in Table 2.

5.2 *Location* Features

Whereas previously, with 2D tracking, a coarse grid is applied, in this section the skeleton returned by the PrimeSense tracker can now be leveraged. This allows signer related locations to be described with higher confidence. As such, the location features are calculated using the distance of the dominant hand from skeletal joints. A feature will fire if the dominant hand is closer than $H^{head}/2$ of the joint in question. A list of the 9 joints considered is shown in Table 2 and displayed to scale

Locations	Motions		
	Right or Left Hand		Bi-manual
head	left	$\Delta x > \lambda$	in sync
neck	right	$\Delta x < -\lambda$	$\lvert\delta(L,R)\rvert < \lambda$
torso	up	$\Delta y > \lambda$	and
L shoulder	down	$\Delta y < -\lambda$	$F^R = F^L$
L elbow	towards	$\Delta z > \lambda$	together
L hand	away	$\Delta z < -\lambda$	$\Delta(\delta(L,R)) < -\lambda$
L hip	none	$\Delta L < \lambda$	apart
R shoulder		$\Delta R < \lambda$	$\Delta(\delta(L,R)) > \lambda$
R hip			

Table 2: Table listing the locations and hand motions included in the feature vectors. The conditions for motion are shown with the label. Where x, y, z is the position of the hand, either left (L) or right (R), Δ indicates a change from one frame to the next and $\delta(L,R)$ is the Euclidean distance between the left and right hands. λ is the threshold value to reduce noise and increase generalisation, this is set to be a quarter the head height. F^R and F^L are the motion feature vectors relating to the right and left hand respectively.

in Figure 11. While displayed in 2D, the regions surrounding the joints are actually 3D spheres. When the dominant hand (in this image shown by the smaller red dot) moves into the region around a joint then that feature will fire. In the example shown, it would be difficult for two features to fire at once. When in motion, the left hand and elbow regions may overlap with other body regions meaning that more than one feature fires at a time.

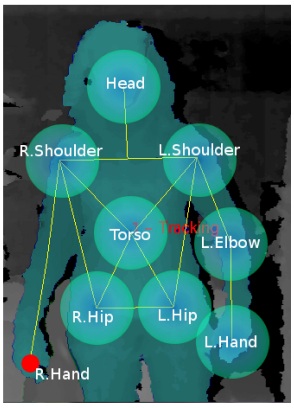


Figure 11: Body joints used to extract sign locations

6. Sign Level classification

Each of the different sub-unit classifier sets is now combined with a sign-level classifier. The groups of binary feature vectors are each concatenated to create a single binary feature vector $F = (f_i)_{i=1}^{D}$

per frame, where $f_i \in \{0,1\}$ and D is the number of dimensions in the feature vector. This feature vector is then used as the input to a sign level classifier for recognition. By using a binary approach, better generalisation is obtained. This requires far less training data than approaches which must generalise over both a continuous input space as well as the variability between signs (e.g., HMMs). Two sign level classification methods are investigated. Firstly, Markov models which use the feature vector as a whole and secondly Sequential Patten Boosting which performs discriminative feature selection.

6.1 Markov Models

HMMs are a proven technology for time series analysis and recognition. While they have been employed for sign recognition, they have issues due to the large training requirements. Kadir et al. (2004) overcame these issues by instead using a simpler Markov model when the feature space is discrete. The symbolic nature of linguistic sub-units means that the discrete time series of events can be modelled without a hidden layer. To this end a Markov chain is constructed for each sign in a lexicon. An ergodic model is used and a Look Up Table (LUT) employed to maintain as little of the chain as is required. Code entries not contained within the LUT are assigned a nominal probability. This is done to avoid otherwise correct chains being assigned zero probabilities if noise corrupts the input signal. The result is a sparse state transition matrix, $P_\omega(F_t|F_{t-1})$, for each word ω giving a classification bank of Markov chains. During creation of this transition matrix, secondary transitions can be included, where $P_\omega(F_t|F_{t-2})$. This is similar to adding skip transitions to the left-right hidden layer of a HMM which allows deletion errors in the incoming signal. While it could be argued that the linguistic features constitute discrete emission probabilities; the lack of a doubly stochastic process and the fact that the hidden states are determined directly from the observation sequence, separates this from traditional HMMs which cannot be used due to their high training requirements. During classification, the model bank is applied to incoming data in a similar fashion to HMMs. The objective is to calculate the chain which best describes the incoming data, that is, has the highest probability that it produced the observation F. Feature vectors are found in the LUT using an L1 distance on the binary vectors. The probability of a model matching the observation sequence is calculated as

$$P(\omega|s) = \upsilon_w \prod_{t=1}^{l} P_\omega(F_t|F_{t-1}),$$

where l is the length of the word in the test sequence and υ_ω is the prior probability of a chain starting in any one of its states. In this work, without grammar, $\forall\omega, \upsilon_\omega = 1$.

6.2 SP Boosting

One limitation of Markov models is that they encode exact series of transitions over all features rather than relying only on discriminative features. This leads to reliance on user dependant feature combinations which if not replicated in test data, will result in poor recognition performance. Sequential Patterns (SPs), on the other hand, compare the input data for relevant features and ignore the irrelevant features. A SP is a sequence of discriminative *itemsets* (i.e., feature subsets) that occur in positive examples and not negative examples (see Figure 12). We define an itemset T as the dimensions of the feature vector $F = (f_i)_{i=1}^{D}$ that have the value of 1: $T \subset \{1,...,D\}$ is a set of

integers where $\forall t \in T, f_t = 1$. Following this, we define a SP $\mathbf{T}$ of length $|\mathbf{T}|$ as: $\mathbf{T} = (T_i)_{i=1}^{|\mathbf{T}|}$, where T_i is an itemset.

In order to use SPs for classification, we first define a method for detecting SPs in an input sequence of feature vectors. To this end, firstly let $\mathbf{T}$ be a SP we wish to detect. Suppose the given feature vector input sequence of $|\mathbf{F}|$ frames is $\mathbf{F} = (F_t)_{t=1}^{|F|}$, where F_t is the binary feature vector defined in Section 6. We firstly convert $\mathbf{F}$ into the SP $\mathbf{I} = (I_t)_{t=1}^{|\mathbf{F}|}$, where I_t is the itemset of feature vector F_t. We say that the SP $\mathbf{T}$ is present in $\mathbf{I}$ if there exists a sequence $(\beta_i)_{i=1}^{|\mathbf{T}|}$, where $\beta_i < \beta_j$ when $i < j$ and $\forall i = \{1, ..., |\mathbf{T}|\}, T_i \subset I_{\beta_i}$. This relationship is denoted with the $\subset_S$ operator, that is, $\mathbf{T} \subset_S \mathbf{I}$. Conversely, if the sequence $(\beta_i)_{i=1}^{|\mathbf{T}|}$ does not exist, we denote it as $\mathbf{T} \not\subset_S \mathbf{I}$.

From this, we can then define a SP weak classifier as follows: Let $\mathbf{T}$ be a given SP and $\mathbf{I}$ be an itemset sequence derived from some input binary vector sequence F. A *SP weak classifier*, $h^{\mathbf{T}}(\mathbf{I})$, can be constructed as follows:

$$h^{\mathbf{T}}(\mathbf{I}) = \begin{cases} 1, & \text{if } \mathbf{T} \subset_S \mathbf{I}, \\ -1, & \text{if } \mathbf{T} \not\subset_S \mathbf{I}. \end{cases}$$

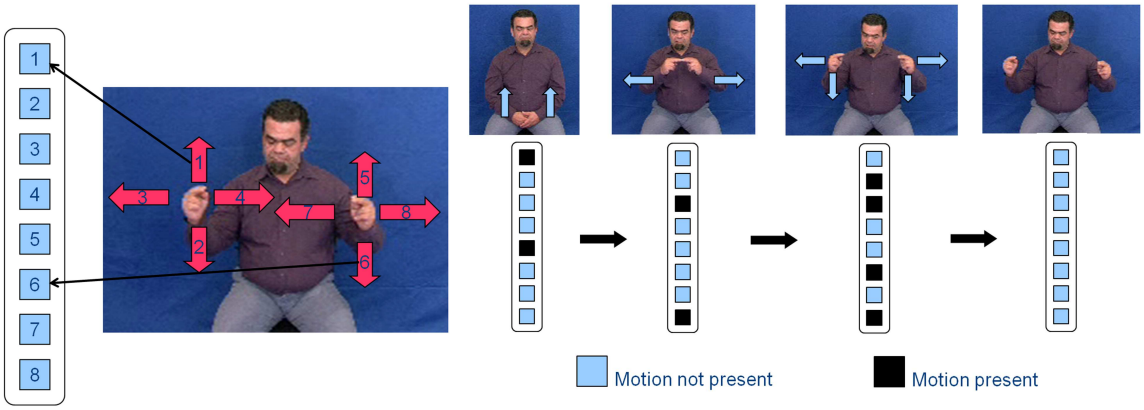
A strong classifier can be constructed by linearly combining a number (S) of selected SP weak classifiers in the form of:

$$H(I) = \sum_{i=1}^{S} \alpha_i h_i^{\mathbf{T}_i}(I).$$

The weak classifiers h_i are selected iteratively based on example weights formed during training. In order to determine the optimal weak classifier at each Boosting iteration, the common approach is to exhaustively consider the entire set of candidate weak classifiers and finally select the best weak classifier (i.e., that with the lowest weighted error). However, finding SP weak classifiers corresponding to optimal SPs this way is not possible due to the immense size of the SP search space. To this end, the method of SP Boosting is employed (Ong and Bowden, 2011). This method poses the learning of discriminative SPs as a tree based search problem. The search is made efficient by employing a set of pruning criteria to find the SPs that provide optimal discrimination between the positive and negative examples. The resulting tree-search method is integrated into a boosting framework; resulting in the SP-Boosting algorithm that combines a set of unique and optimal SPs for a given classification problem. For this work, classifiers are built in a one-vs-one manner and the results aggregated for each sign class.

7. Appearance Based Results

This section of work uses the same 164 sign data set as Kadir et al. (2004) and therefore a direct comparison can be made between their hard coded tracking based system and the learnt sub-unit approach using detection based sub-units. For this work, extra annotation was required as Kadir et al. (2004) used only sign boundaries. 7410 *Location* examples, 322 *Hand-Arrangement* examples and 578 *Motion* were hand labelled for training sub-unit classifiers. The data set consists of 1640 examples (ten of each sign). Signs were chosen randomly rather than picking specific examples which are known to be easy to separate. The sub-unit classifiers are built using only data from four of the ten examples of each sign and the word level classifier is then trained on five examples (including the four previously seen by the sub-unit classifiers) leaving five completely unseen examples for



(a) Feature vector (b) SP

Figure 12: Pictorial description of SPs. (a) shows an example feature vector made up of 2D motions of the hands. In this case the first element shows 'right hand moves up', the second 'right hand moves down' etc. (b) shows a plausible pattern that might be found for the sign 'bridge'. In this sign the hands move up to meet each other, they move apart and then curve down as if drawing a hump-back bridge.

testing purposes. The second stage classifier is trained on the previously used four training examples plus one other, giving five training examples per sign. The results are acquired from the five unseen examples of each of the 164 signs. This is done for all six possible combinations of training/test data. Results are shown in Table 3 alongside the results from Kadir et al. (2004). The first three columns show the results of combining each type of appearance sub-unit with the second stage sign classifier. Unsurprisingly, none of the individual types contains sufficient information to be able to accurately separate the data. However, when combined, the appearance based classifiers learnt from the data are comparable to the hard coded classifiers used on perfectly tracked data. The performance drops by only 6.6 Percentage Points (pp), from 79.2% to 72.6% whilst giving the advantage of not needing the high quality tracking system.

Figure 13, visually demonstrates the sub-unit level classifiers being used with the second stage classifier. The output from the sub-unit classifiers are shown on the right hand side in a vector format on a frame by frame basis. It shows the repetition of features for the sign 'Box'. As can be seen there is a pattern in the vector which repeats each time the sign is made. It is this repetition which the second stage classifier is using to detect signs.

8. 2D Tracking Results

The data set used for these experiments contains 984 Greek Sign Language (GSL) signs with 5 examples of each performed by a single signer (for a total of 4920 samples). The handshape classifiers are learnt on data from the first 4 examples of each sign. The sign level classifiers are trained on the same 4 examples, the remaining sign of each type is reserved for testing.

	Hand-Arrangement	***Location***	***Motion***	**Combined**	**(Kadir et al., 2004)**
Minimum (%)	31.6	30.7	28.2	68.7	76.1
Maximum (%)	35.0	32.2	30.5	74.3	82.4
Std Dev	0.9	0.4	0.6	1.5	2.1
Mean (%)	33.2	31.7	29.4	72.6	79.2

Table 3: Classification performance of the appearance based two-stage detector. Using the appearance based sub-unit classifiers. Kadir et al. (2004) results are included for comparison purposes.

Figure 13: Repetition of the appearance based sub-unit classifier vector. The band down the right hand side of the frame shows the sub-unit level classifier firing patterns for the last 288 frames, the vector for the most recent frame is at the bottom. The previous video during the 288 frames shows four repetitions of the sign 'Box'.

Table 4 shows sign level classification results. It is apparent from these results, that out of the independent vectors, the location information is the strongest. This is due to the strong combination of a detailed location feature vector and the temporal information encoded by the Markov chain.

Shown also is the improvement afforded by using the handshape classifiers with a threshold vs a WTA implementation. By allowing the classifiers to return multiple possibilities more of the data about the handshape is captured. Conversely, when none of the classifiers is confident, a 'null' response is permitted which reduces the amount of noise. Using the non-mutually exclusive version of the handshapes in combination with the motion and location, the percentage of signs correctly

Motion	25.1%
Location	60.5%
HandShape	3.4%
All: WTA	52.7%
All: Thresh	68.4%
All + Skips ($P(F_t\|F_{t-2})$)	**71.4%**

Table 4: Sign level classification results using 2D tracked features and the Markov Models. The first three rows show the results when using the features independently with the Markov chain (The handshapes used are non-mutually exclusive). The next three rows give the results of using all the different feature vectors. Including the improvement gained by allowing the handshapes to be non-mutually exclusive (thresh) versus the WTA option. The final method is the combination of the superior handshapes with the location, motion and the second order skips.

	Markov Chains		SPs	
	Top 1	Top 4	Top 1	Top 4
recall	**71.4%**	**82.3%**	**74.1%**	**89.2%**

Table 5: Comparison of recall results on the 2D tracking data using both Markov chains and SPs

returned is 68.4%. By including the 2nd order transitions whilst building the Markov chain there is a 3 pp boost to 71.4%.

This work was developed for use as a sign dictionary, within this context, when queried by a video search, the classification would not return a single response. Instead, like a search engine, it should return a ranked list of possible signs. Ideally the target sign would be close to the top of this list. To this end we show results for 2 possibilities; The percentage of signs which are correctly ranked as the first possible sign (Top 1) and the percentage which are ranked in the top 4 possible signs.

This approach is applied to the best sub-unit features above combined with either the Markov Chains or the SP trees. The results of these tests are shown in Table 5. When using the the same combination of sub-unit features as found to be optimal with the Markov Chains, the SP trees are able to improve on the results by nearly 3 pp, increasing the recognition rate from 71.4% to 74.1%. A further improvement is also found when expanding the search results list, within the top 4 signs the recall rate increases from 82.3% to 89.2%.

9. 3D Tracking Results

While the Kinect^{TM}work is intended for use as a live system, quantitative results can be obtained by the standard method of splitting pre-recorded data into training and test sets. The split between test and training data can be done in several ways. This work uses two versions, the first to show results on signer dependent data, as is often used, the second shows performance on unseen signers, a signer independent test.

	Test	Markov Models		SP-Boosting	
		Top 1	Top 4	Top 1	Top 4
Independent	1	56%	80%	72%	91%
	2	61%	79%	80%	98%
	3	30%	45%	67%	89%
	4	55%	86%	77%	95%
	5	58%	75%	78%	98%
	6	63%	83%	80%	98%
	Mean	**54%**	**75%**	**76%**	**95%**
	StdDev	12%	15%	5%	4%
Dependent Mean		**79%**	**92%**	**92%**	**99.90%**

Table 6: Results across the 20 sign GSL data set.

9.1 Data Sets

Two data sets were captured for training; The first is a data set of 20 GSL signs, randomly chosen and containing both similar and dissimilar signs. This data includes six people performing each sign an average of seven times. The signs were all captured in the same environment with the Kinect™and the signer in approximately the same place for each subject. The second data set is larger and more complex. It contains 40 Deutsche Gebärdensprache - German Sign Language (DGS) signs, chosen to provide a phonetically balanced subset of HamNoSys phonemes. There are 15 participants each performing all the signs 5 times. The data was captured using a mobile system giving varying view points.

9.2 GSL Results

Two variations of tests were performed; firstly the signer dependent version, where one example from each signer was reserved for testing and the remaining examples were used for training. This variation was cross-validated multiple times by selecting different combinations of train and test data. Of more interest for this application however, is signer independent performance. For this reason the second experiment involves reserving data from a subject for testing, then training on the remaining signers. This process is repeated across all signers in the data set. The results of both the Markov models and the Sequential Patten Boosting applied to the basic 3D features are shown in Table 6.

As is noted in Section 6.2, while the the Markov models perform well when they have training data which is close to the test data, they are less able to generalise. This is shown by the dependent results being high, average 92% within the top 4, compared to the average independent result which is 17 pp lower at 75%. It is even more noticeable when comparing the highest ranked sign only, which suffers from a drop of 25 pp, going from 79% to 54%. When looking at the individual results of the independent test it can be seen that there are obvious outliers in the data, specifically signer 3 (the only female in the data set), where the recognition rates are markedly lower. This is reflected in statistical analysis which gives high standard deviation across the signers in both the top 1 and top 4 rankings when using the Markov Chains.

	Subject Dependent		Subject Independent	
	Top 1	Top 4	Top 1	Top 4
Min	56.7%	90.5%	39.9%	74.9%
Max	64.5%	94.6%	67.9%	92.4%
StdDev	1.9%	1.0%	8.5%	5.2%
Mean	**59.8%**	**91.9%**	**49.4%**	**85.1%**

Table 7: Subject Independent (SI) and Subject Dependent (SD) test results across 40 signs in the DGS data set.

When the SP-Boosting is used, again the dependant case produces higher results, gaining nearly 100% when considering the top 4 ranked signs. However, due to the discriminative feature selection process employed; the user independent case does not show such marked degradation, dropping just 4.9 pp within the top 4 signs, going from 99.9% to 95%. When considering the top ranked sign the reduction is more significant at 16 pp, from 92% to 76%, but this is still a significant improvement on the more traditional Markov model. It can also be seen that the variability in results across signers is greatly reduced using SP-Boosting, whilst signer 3 is still the signer with the lowest percentage of signs recognised, the standard deviation across all signs has dropped to 5% for the first ranked signs and is again lower for the top 4 ranked signs.

9.3 DGS Results

The DGS data set offers a more challenging task as there is a wider range of signers and environments. Experiments were run in the same format using the same features as for the GSL data set. Table 7 shows the results of both the dependent and independent tests. As can be seen with the increased number of signs the percentage accuracy for the first returned result is lower than that of the GSL tests at 59.8% for dependent and 49.4% for independent. However the recall rates within the top 4 ranked signs (now only 10% of the data set) are still high at 91.9% for the dependent tests and 85.1% for the independent ones. Again the relatively low standard deviation of 5.2% shows that the SP-Boosting is picking the discriminative features which are able to generalise well to unseen signers.

As can be seen in the confusion matrix (see Figure 14), while most signs are well distinguished, there are some signs which routinely get confused with each other. A good example of this is the three signs ‘already’, ‘Athens’ and ‘Greece’ which share very similar hand motion and location but are distinguishable by handshape which is not currently modelled on this data set.

10. Discussion

Three different approaches to sub-unit feature extraction have been compared in this paper. The first based on appearance only, the latter two on tracking. The advantage of the first approach is that it doesn’t depend on high quality tracking for good results. However, it would be easily confused via cluttered backgrounds or short sleeves (often a problem with sign language data sets). The other advantage of the appearance based classification is that it includes information not available

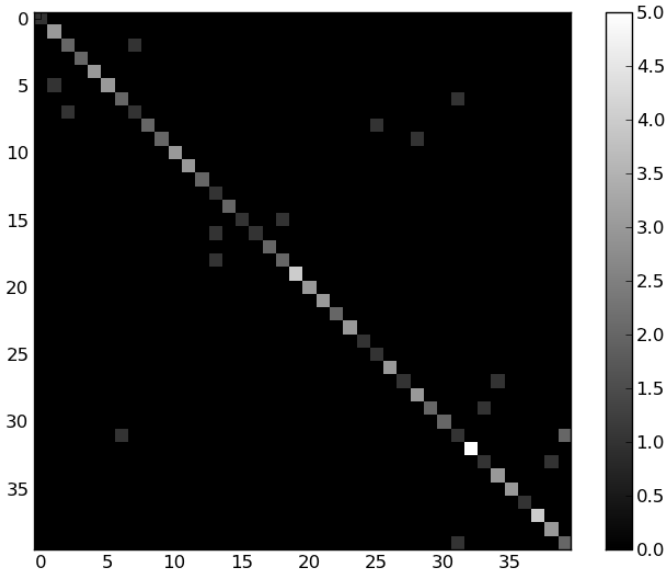


Figure 14: Aggregated confusion matrix of the first returned result for each subject independent test on the DGS data set.

by trajectories alone, thus encoding information about handshape within the moment based classifiers. While this may aid classification on small data sets it makes it more difficult to de-couple the handshape from the motion and location sub-units. This affects the generalisation ability of the classifiers due to the differences between signers.

Where 2D tracking is available, the results are superior in general to the appearance based results. This is shown in the work by Kadir et al. (2004), who achieve equivalent results on the same data using tracking trajectories when compared to the appearance based ones presented here. Unfortunately, it is not always possible to accurately track video data and this is why it is still valid to examine appearance based approaches. The 2D tracking *Location* sub-features presented here are based around a grid, while this is effective in localising the motion it is not as desirable as the HamNoSys derived features used in the improved 3D tracking features. The grid suffers from boundary noise as the hands move between cells. This noise causes problems when the features are used in the second stage of classification. With the 3D features this is less obvious due to them being relative to the signer in 3D and therefore the locations are not arbitrarily used by the signer in the same way as the grid is. For example if a signer puts their hands to their shoulders, this will cause multiple cells of the grid to fire and it may not be the same one each time. When using 3D, if the signer puts their hands to their shoulders then the shoulder feature fires. This move from an arbitrary grid to consciously decided body locations reduces boundary effect around significant areas in the signing space.

This in turn leads to the sign level classifiers. The Markov chains are very good at recognising signer dependent, repetitive motion, in these cases they are almost on a par with the SPs. However, they are much less capable of managing signer independent classification as they are unable to distinguish between the signer accents and the signs themselves and therefore over-fit the data.

Instead the SPs look for the discriminative features between the examples, ignoring any signer specific features which might confuse the Markov Chains.

11. Conclusions

This work has presented three approaches to sub-unit based sign recognition. Tests were conducted using boosting to learn three types of sub-units based on appearance features, which are then combined with a second stage classifier to learn word level signs. These appearance based features offer an alternative to costly tracking.

The second approach uses a 2D tracking based set of sub-units combined with some appearance based handshape classifiers. The results show that a combination of these robust, generalising features from tracking and learnt handshape classifiers overcomes the high ambiguity and variability in the data set to achieve excellent recognition performance: achieving a recognition rate of 73% on a large data set of 984 signs.

The third and final approach translates these tracking based sub-units into 3D, this offers user independent, real-time recognition of isolated signs. Using this data a new learning method is introduced, combining the sub-units with SP-Boosting as a discriminative approach. Results are shown on two data sets with the recognition rate reaching 99.9% on a 20 sign multi-user data set and 85.1% on a more challenging and realistic subject independent, 40 sign test set. This demonstrates that true signer independence is possible when more discriminative learning methods are employed. In order to strengthen comparisons within the SLR field the data sets created within this work have been released for use within the community.

12. Future Work

The learnt sub-units show promise and, as shown by the work of Pitsikalis et al. (2011), there are several avenues which can be explored. However, for all of these directions, more linguistically annotated data is required across multiple signers to allow the classifiers to discriminate between the features which are signer specific and those which are independent. In addition, handshapes are a large part of sign, while the work on the multi-signer depth data set has given good results, handshapes should be included in future work using depth cameras. Finally, the recent creation of a larger, multi-signer data set has set the ground work in place for better quantitative analysis. Using this data in the same manner as the DGS40 data set should allow bench-marking of Kinect sign recognition approaches, both for signer dependent and independent recognition. Appearance only techniques can also be verified using the Kinect data set where appropriate as the RGB images are also available though they are not used in this paper. Though it should be noted that this is an especially challenging data set for appearance techniques due to the many varying backgrounds and subjects.

Acknowledgments

The research leading to these results has received funding from the European Community's Seventh Framework Programme (FP7/2007-2013) under grant agreement number 231135 Dicta-Sign. The Dicta-Sign data sets used and additional SL resources are available via `http://www.dictasign.eu/`

References

Y. Amit and D. Geman. Shape quantization and recognition with randomized trees. *Neural Computation*, 9:1545–1588, 1997.

L. Breiman. Random forests. *Machine Learning*, pages 5–32, 2001.

British Deaf Association. *Dictionary of British Sign Language/English*. Faber and Faber, 1992.

P. Buehler, M. Everingham, and A. Zisserman. Learning sign language by watching TV (using weakly aligned subtitles). In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 2961 – 2968, Miami, FL, USA, June 20 – 26 2009.

H. Cooper and R. Bowden. Large lexicon detection of sign language. In *Proceedings of the IEEE International Conference on Computer Vision: Workshop Human Computer Interaction*, pages 88 – 97, Rio de Janario, Brazil, October 16 – 19 2007. doi: 10.1007/978-3-540-75773-3_10.

H. Cooper and R. Bowden. Sign language recognition using linguistically derived sub-units. In *Proceedings of the Language Resources and Evaluation ConferenceWorkshop on the Representation and Processing of Sign Languages : Corpora and Sign Languages Technologies*, Valetta, Malta, May17 – 23 2010.

R. Elliott, J. Glauert, J. Kennaway, and K. Parsons. D5-2: SiGML Definition. *ViSiCAST Project working document*, 2001.

H. Ershaed, I. Al-Alali, N. Khasawneh, and M. Fraiwan. An arabic sign language computer interface using the xbox kinect. In *Annual Undergraduate Research Conf. on Applied Computing*, May 2011.

Y. Freund and R. E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. In *Proceedings of the European Conference on Computational Learning Theory*, pages 23 – 37, Barcelona, Spain, March 13 – 15 1995. Springer-Verlag. ISBN 3-540-59119-2.

J.W. Han, G. Awad, and A. Sutherland. Modelling and segmenting subunits for sign language recognition based on hand motion analysis. *Pattern Recognition Letters*, 30(6):623 – 633, April 2009.

T Hanke and C Schmaling. *Sign Language Notation System*. Institute of German Sign Language and Communication of the Deaf, Hamburg, Germany, January 2004. URL `http://www.sign-lang.uni-hamburg.de/projects/hamnosys.html`.

M. K. Hu. Visual pattern recognition by moment invariants. *IRE Transactions on Information Theory*, IT-8:179–187, February 1962.

T. Kadir, R. Bowden, E.J. Ong, and A Zisserman. Minimal training, large lexicon, unconstrained sign language recognition. In *Proceedings of the BMVA British Machine Vision Conference*, volume 2, pages 939 – 948, Kingston, UK, September 7 – 9 2004.

S Kim and M.B Waldron. Adaptation of self organizing network for ASL recognition. In *Proceedings of the Annual International Conference of the IEEE Engineering in Engineering in Medicine and Biology Society*, pages 254 – 254, San Diego, California, USA, October 28 – 31 1993.

W.W. Kong and S. Ranganath. Automatic hand trajectory segmentation and phoneme transcription for sign language. In *Proceedings of the IEEE International Conference on Automatic Face and Gesture Recognition*, pages 1 – 6, Amsterdam, The Netherlands, September 17 – 19 2008. doi: 10.1109/AFGR.2008.4813462.

S.K. Liddell and R.E Johnson. American sign language: The phonological base. *Sign Language Studies*, 64:195 – 278, 1989.

K. Lyons, H. Brashear, T. L. Westeyn, J. S. Kim, and T. Starner. Gart: The gesture and activity recognition toolkit. In *Proceedings of the International Conference HCI*, pages 718–727, July 2007.

E. J. Ong and R. Bowden. Learning sequential patterns for lipreading. In *Proceedings of the BMVA British Machine Vision Conference*, Dundee, UK, August 29 – September 10 2011.

OpenNI User Guide. OpenNI organization, November 2010. Last viewed 20-04-2011 18:15.

V. Pitsikalis, S. Theodorakis, C. Vogler, and P. Maragos. Advances in phonetics-based sub-unit modeling for transcription alignment and sign language recognition. In *Proceedings of the International Conference IEEE Computer Society Conference on Computer Vision and Pattern RecognitionWorkshop : Gesture Recognition*, Colorado Springs, CO, USA, June 21 – 23 2011.

Prime Sensor™NITE 1.3 Algorithms notes. PrimeSense Inc., 2010. Last viewed 20-04-2011 18:15.

A. Roussos, S. Theodorakis, V. Pitsikalis, and P. Maragos. Hand tracking and affine shape-appearance handshape sub-units in continuous sign language recognition. In *Proceedings of the International Conference European Conference on Computer VisionWorkshop : SGA*, Heraklion, Crete, September 5 – 11 2010.

J. E. Shoup. Phonological aspects of speech recognition. In Wayne A. Lea, editor, *Trends in Speech Recognition*, pages 125 – 138. Prentice-Hall, Englewood Cliffs, NJ, 1980.

T. Starner and A. Pentland. Real-time american sign language recognition from video using hidden markov models. *Computational Imaging and Vision*, 9:227 – 244, 1997.

W.C Stokoe. Sign language structure: An outline of the visual communication systems of the american deaf. *Studies in Linguistics: Occasional Papers*, 8:3 – 37, 1960.

P. Viola and M. Jones. Rapid object detection using a boosted cascade of simple features. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 1, pages 511 – 518, Kauai, HI, USA, December 2001.

C. Vogler and D Metaxas. Adapting hidden markov models for ASL recognition by using three-dimensional computer vision methods. In *Proceedings of the IEEE International Conference on Systems, Man, and Cybernetics*, volume 1, pages 156 – 161, Orlando, FL, USA, October 12 – 15 1997.

C. Vogler and D Metaxas. Parallel hidden markov models for american sign language recognition. In *Proceedings of the IEEE International Conference on Computer Vision*, volume 1, pages 116 – 122, Corfu, Greece, September 21 – 24 1999.

M. B. Waldron and S Kim. Increasing manual sign recognition vocabulary through relabelling. In *Proceedings of the IEEE International Conference on Neural Networks IEEE World Congress on Computational Intelligence*, volume 5, pages 2885 – 2889, Orlando, Florida, USA, June 27 – July 2 1994. doi: 10.1109/ICNN.1994.374689.

M. B. Waldron and S Kim. Isolated ASL sign recognition system for deaf persons. *IEEE Transactions on Rehabilitation Engineering*, 3(3):261 – 271, September 1995. doi: 10.1109/86.413199.

M.B. Waldron and D Simon. Parsing method for signed telecommunication. In *Proceedings of the Annual International Conference of the IEEE Engineering in Engineering in Medicine and Biology Society: Images of the Twenty-First Century*, volume 6, pages 1798 – 1799, Seattle, Washington, USA, November 1989. doi: 10.1109/IEMBS.1989.96461.

H. Wassner. kinect + reseau de neurone = reconnaissance de gestes. http://tinyurl.com/5wbteug, May 2011.

P. Yin, T. Starner, H. Hamilton, I. Essa, and J.M. Rehg. Learning the basic units in american sign language using discriminative segmental feature selection. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 4757 – 4760, Taipei, Taiwan, April 19 – 24 2009. doi: 10.1109/ICASSP.2009.4960694.

Z. Zafrulla, H. Brashear, P. Presti, H. Hamilton, and T. Starner. Copycat - center for accessible technology in sign. http://tinyurl.com/3tksn6s, December 2010. URL `http://www.youtube.com/watch?v=qFH5rSzmgFE&feature=related`.

Z. Zafrulla, H. Brashear, T. Starner, H. Hamilton, and P. Presti. American sign language recognition with the kinect. In *Proceedings of the 13th International Conference on Multimodal Interfaces*, ICMI '11, pages 279–286, New York, NY, USA, 2011. ACM. ISBN 978-1-4503-0641-6. doi: 10.1145/2070481.2070532. URL `http://doi.acm.org/10.1145/2070481.2070532`.

Journal of Machine Learning Research 13 (2012) 2233-2236 Submitted 10/11; Revised 1/12; Published 8/12

A Topic Modeling Toolbox Using Belief Propagation

Jia Zeng J.ZENG@IEEE.ORG
School of Computer Science and Technology
Soochow University
Suzhou 215006, China

Editor: Cheng Soon Ong

Abstract

Latent Dirichlet allocation (LDA) is an important hierarchical Bayesian model for probabilistic topic modeling, which attracts worldwide interests and touches on many important applications in text mining, computer vision and computational biology. This paper introduces a topic modeling toolbox (TMBP) based on the belief propagation (BP) algorithms. TMBP toolbox is implemented by MEX C++/Matlab/Octave for either Windows 7 or Linux. Compared with existing topic modeling packages, the novelty of this toolbox lies in the BP algorithms for learning LDA-based topic models. The current version includes BP algorithms for latent Dirichlet allocation (LDA), author-topic models (ATM), relational topic models (RTM), and labeled LDA (LaLDA). This toolbox is an ongoing project and more BP-based algorithms for various topic models will be added in the near future. Interested users may also extend BP algorithms for learning more complicated topic models. The source codes are freely available under the GNU General Public Licence, Version 1.0 at `https://mloss.org/software/view/399/`.

Keywords: topic models, belief propagation, variational Bayes, Gibbs sampling

1. Introduction

The past decade has seen rapid development of latent Dirichlet allocation (LDA) (Blei et al., 2003) for solving topic modeling problems because of its elegant three-layer graphical representation as well as two efficient approximate inference methods such as Variational Bayes (VB) (Blei et al., 2003) and collapsed Gibbs Sampling (GS) (Griffiths and Steyvers, 2004). Both VB and GS have been widely used to learn variants of LDA-based topic models until our recent work (Zeng et al., 2011) reveals that there is yet another learning algorithm for LDA based on loopy belief propagation (BP). The basic idea of BP is inspired by the collapsed GS algorithm, in which the three-layer LDA can be interpreted as being collapsed into a two-layer factor graph (Kschischang et al., 2001). The sum-product BP algorithm operates on the factor graph (Bishop, 2006). Extensive experiments confirm that BP is faster and more accurate than both VB and GS, and thus is a strong candidate for becoming the standard topic modeling algorithm. For example, we show how to learn three typical variants of LDA-based topic models, such as author-topic models (ATM) (Rosen-Zvi et al., 2004), relational topic models (RTM) (Chang and Blei, 2010), and labeled LDA (LaLDA) (Ramage et al., 2009) using BP based on the novel factor graph representations (Zeng et al., 2011).

We have implemented the topic modeling toolbox called TMBP by MEX C++ in the Matlab/Octave interface based on VB, GS and BP algorithms. Compared with other topic modeling

packages,[1234567] the novelty of this toolbox lies in the BP algorithms for topic modeling. This paper introduces how to use this toolbox for basic topic modeling tasks.

2. Belief Propagation for Topic Modeling

Given a document-word matrix $\mathbf{x} = \{x_{w,d}\}$ ($x_{w,d}$ is the number of word counts at the index $\{w,d\}$) with word indices $1 \leq w \leq W$ in the vocabulary and document indices $1 \leq d \leq D$ in the corpus, the probabilistic topic modeling task is to allocate topic labels $\mathbf{z} = \{z^k_{w,d}\}, z^k_{w,d} \in \{0,1\}, \sum_{k=1}^K z^k_{w,d} = 1, 1 \leq k \leq K$ to partition the nonzero elements $x_{w,d} \neq 0$ into K topics (provided by the user) according to three topic modeling rules:

1. Co-occurrence: the different word indices w in the same document d tend to have the same topic label.

2. Smoothness: the same word indices w in the different documents d tend to have the same topic label.

3. Clustering: all word indices w do not tend to be associated with the same topic label.

Based on the above rules, recent approximate inference methods compute the marginal distribution of topic label $\mu_{w,d}(k) = p(z^k_{w,d} = 1)$ called *message*, and estimate parameters using the iterative EM (Bishop, 2006) algorithm according to the maximum-likelihood criterion. The major difference among these inference methods lies in the message update equation. VB updates messages by complicated digamma functions, which cause bias and slow down message updating (Zeng et al., 2011). GS updates messages by topic labels randomly sampled from the message in the previous iteration. The sampling process does not keep all uncertainty encoded in the previous message. In contrast, BP directly uses the previous message to update the current message without sampling. Similar ideas have also been proposed within the approximate mean-field framework (Asuncion, 2010) as the zero-order approximation of the collapsed VB (CVB0) algorithm (Asuncion et al., 2009). While proper settings of hyperparameters can make the topic modeling performance comparable among different inference methods (Asuncion et al., 2009), we still advocate the BP algorithms because of their ease of use and fast speed. Table 1 compares the message update equations among VB, GS and BP. Compared with BP, VB uses the digamma function Ψ in message update, and GS uses the discrete count of sampled topic labels $n^{-i}_{w,d}$ based on word tokens rather than word index in message update. The Dirichlet hyperparameters α and β can be viewed as the pseudo-messages. The notations $-w$ and $-d$ denote all word indices except w and all document indices except d, and $-i$ denotes all word tokens except the current word token i. More details can be found in our work (Zeng et al., 2011, 2012a).

Because VB and GS have been widely used for learning different LDA-based topic models, it is easy to develop the corresponding BP algorithms for learning these LDA-based topic models by

1. See http://www.cs.princeton.edu/~blei/lda-c/index.html.
2. See http://psiexp.ss.uci.edu/research/programs_data/toolbox.htm.
3. See http://nlp.stanford.edu/software/tmt/tmt-0.3/.
4. See http://CRAN.R-project.org/package=lda.
5. See http://mallet.cs.umass.edu/.
6. See http://www.arbylon.net/projects/.
7. See http://CRAN.R-project.org/package=topicmodels.

Inference methods	Message update equations
VB	$\mu_{w,d}(k) \propto \frac{\exp[\Psi(x_{\cdot,d}\mu_{\cdot,d}(k)+\alpha)]}{\exp[\Psi(\sum_k[x_{\cdot,d}\mu_{\cdot,d}(k)+\alpha])]} \times \frac{x_{w,\cdot}\mu_{w,\cdot}(k)+\beta}{\sum_w[x_{w,\cdot}\mu_{w,\cdot}(k)+\beta]}$
GS	$\mu_{w,d,i}(k) \propto \frac{n^{-i}_{\cdot,d}(k)+\alpha}{\sum_k[n^{-i}_{\cdot,d}(k)+\alpha]} \times \frac{n^{-i}_{w,\cdot}(k)+\beta}{\sum_w[n^{-i}_{w,\cdot}(k)+\beta]}$
BP	$\mu_{w,d}(k) \propto \frac{x_{-w,d}\mu_{-w,d}(k)+\alpha}{\sum_k[x_{-w,d}\mu_{-w,d}(k)+\alpha]} \times \frac{x_{w,-d}\mu_{w,-d}(k)+\beta}{\sum_w[x_{w,-d}\mu_{w,-d}(k)+\beta]}$

Table 1: Comparison of message update equations (Zeng et al., 2011).

either removing the digamma function in the VB or without sampling from the posterior probability in the GS algorithm. For example, we show how to develop the corresponding BP algorithms for two typical LDA-based topic models such as ATM and RTM (Zeng et al., 2011).

3. An Example of Using TMBP

TMBP toolbox contains source codes for learning LDA based on VB, GS, and BP (Zeng et al., 2011, 2012a,b,c), learning author-topic models (ATM) (Rosen-Zvi et al., 2004) based on GS and BP, learning relational topic models (RTM) (Chang and Blei, 2010) and labeled LDA (Ramage et al., 2009) using BP. Implementation details can be found in "readme.pdf", which is distributed with the software. Here, we present a demo for the synchronous BP algorithm. After installation, we run `demo1.m` in the Octave/Matlab environment. The results (the training perplexity at every 10 iterations and the top five words in each of ten topics) are printed on the screen:

```
**********
The sBP Algorithm
**********
    Iteration 10 of 500:    1041.620873
    ...
    ...
    Iteration 490 of 500:   741.946849
Elapsed time is 13.246747 seconds.
**********
Top five words in each of ten topics by sBP
**********
design system reasoning case knowledge
model models bayesian data markov
genetic problem search algorithms programming
algorithm learning number function model
learning paper theory knowledge examples
learning control reinforcement paper state
model visual recognition system patterns
research report technical grant university
network neural networks learning input
data decision training algorithm classification
```

Acknowledgments

This work is supported by NSFC (Grant No. 61003154), Natural Science Foundation of the Jiangsu Higher Education Institutions of China (Grant No. 12KJA520004), the Shanghai Key Laboratory of Intelligent Information Processing, China (Grant No. IIPL-2010-009), and a grant from Baidu.

References

A. Asuncion. Approximate mean field for Dirichlet-based models. In *ICML Workshop on Topic Models*, 2010.

A. Asuncion, M. Welling, P. Smyth, and Y. W. Teh. On smoothing and inference for topic models. In *UAI*, pages 27–34, 2009.

C. M. Bishop. *Pattern Recognition and Machine Learning*. Springer, 2006.

D. M. Blei, A. Y. Ng, and M. I. Jordan. Latent Dirichlet allocation. *J. Mach. Learn. Res.*, 3: 993–1022, 2003.

J. Chang and D. M. Blei. Hierarchical relational models for document networks. *Annals of Applied Statistics*, 4(1):124–150, 2010.

T. L. Griffiths and M. Steyvers. Finding scientific topics. *Proc. Natl. Acad. Sci.*, 101:5228–5235, 2004.

F. R. Kschischang, B. J. Frey, and H.-A. Loeliger. Factor graphs and the sum-product algorithm. *IEEE Transactions on Inform. Theory*, 47(2):498–519, 2001.

D. Ramage, D. Hall, R. Nallapati, and C. D. Manning. Labeled LDA: A supervised topic model for credit attribution in multi-labeled corpora. In *Empirical Methods in Natural Language Processing*, pages 248–256, 2009.

M. Rosen-Zvi, T. Griffiths, M. Steyvers, and P. Smyth. The author-topic model for authors and documents. In *UAI*, pages 487–494, 2004.

J. Zeng, W. K. Cheung, and J. Liu. Learning topic models by belief propagation. *arXiv:1109.3437v4 [cs.LG]*, 2011.

J. Zeng, Z.-Q. Liu, and X.-Q. Cao. A new approch to speeding up topic modeling. *arXiv:1204.0170v1 [cs.LG]*, 2012a.

J. Zeng, Z.-Q. Liu, and X.-Q. Cao. Memory-efficient topic modeling. *arXiv:1206.1147v1 [cs.LG]*, 2012b.

J. Zeng, Z.-Q. Liu, and X.-Q. Cao. Residual belief propagation for topic modeling. *arXiv:1204.6610v1 [cs.LG]*, 2012c.

Journal of Machine Learning Research 13 (2012) 2237-2278 Submitted 6/10; Revised 9/11; Published 8/12

MedLDA: Maximum Margin Supervised Topic Models

Jun Zhu DCSZJ@MAIL.TSINGHUA.EDU.CN
State Key Lab of Intelligent Technology and Systems
Tsinghua National Lab for Information Science and Technology
Department of Computer Science and Technology
Tsinghua University
Beijing, 100084, China

Amr Ahmed AMAHMED@CS.CMU.EDU
Eric P. Xing EPXING@CS.CMU.EDU
School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213, USA

Editor: David Blei

Abstract

A supervised topic model can use side information such as ratings or labels associated with documents or images to discover more predictive low dimensional topical representations of the data. However, existing supervised topic models predominantly employ likelihood-driven objective functions for learning and inference, leaving the popular and potentially powerful max-margin principle unexploited for seeking predictive representations of data and more discriminative topic bases for the corpus. In this paper, we propose the maximum entropy discrimination latent Dirichlet allocation (MedLDA) model, which integrates the mechanism behind the max-margin prediction models (e.g., SVMs) with the mechanism behind the hierarchical Bayesian topic models (e.g., LDA) under a unified constrained optimization framework, and yields latent topical representations that are more discriminative and more suitable for prediction tasks such as document classification or regression. The principle underlying the MedLDA formalism is quite general and can be applied for jointly max-margin and maximum likelihood learning of directed or undirected topic models when supervising side information is available. Efficient variational methods for posterior inference and parameter estimation are derived and extensive empirical studies on several real data sets are also provided. Our experimental results demonstrate qualitatively and quantitatively that MedLDA could: 1) discover sparse and highly discriminative topical representations; 2) achieve state of the art prediction performance; and 3) be more efficient than existing supervised topic models, especially for classification.

Keywords: supervised topic models, max-margin learning, maximum entropy discrimination, latent Dirichlet allocation, support vector machines

1. Introduction

Probabilistic latent aspect models such as the latent Dirichlet allocation (LDA) model (Blei et al., 2003) have recently gained much popularity for stratifying a large collection of documents by projecting every document into a low dimensional space spanned by a set of bases that capture the semantic aspects, also known as *topics*, of the collection. An LDA model posits that each document is an admixture of latent topics, of which each topic is represented as a unique unigram distribution

over a given vocabulary. The document-specific admixture proportion vector $\boldsymbol{\theta}$, also known as the *topic vector*, is modeled as a latent Dirichlet random variable, and can be regarded as a low dimensional representation of the document in a topical space. This low dimensional representation can be used for downstream tasks such as classification, clustering, or merely as a tool for structurally visualizing the otherwise unstructured document collection.

The original LDA is an unsupervised model and is typically built on a discrete bag-of-words representation of input contents, which can be text documents (Blei et al., 2003), images (Fei-Fei and Perona, 2005), or even network entities (Airoldi et al., 2008). However, in many practical applications, we can easily obtain useful side information besides the document or image contents. For example, when online users post their reviews for products or restaurants, they usually associate each review with a rating score or a thumb-up/thumb-down opinion; web sites or pages in the public Yahoo! Directory[1] can have their categorical labels; and images in the LabelMe (Russell et al., 2008) database are organized by a visual ontology and additionally each image is associated with a set of annotation tags. Furthermore, there is an increasing trend towards using online crowdsourcing services (such as Amazon Mechanical Turk[2]) to collect large collections of labeled data with a reasonably low price (Snow et al., 2008). Such side information often provides useful high-level or direct summarization of the content, but it is not directly used in the original LDA or models alike to influence topic inference. One would expect that incorporating such information into latent aspect modeling could guide a topic model towards discovering secondary or non-dominant, albeit semantically more salient statistical patterns (Chechik and Tishby, 2002) that may be more interesting or relevant to the user's goal, such as prediction on unlabeled data.

To explore this potential, developing new topic models that appropriately capture side information mentioned above has recently gained increasing attention. Representative attempts include supervised topic model (sLDA) (Blei and McAuliffe, 2007), which captures real-valued document rating as a regression response; multi-class sLDA (Wang et al., 2009), which directly captures discrete labels of documents as a classification response; and discriminative LDA (DiscLDA) (Lacoste-Julien et al., 2008), which also performs classification, but with a mechanism different from that of sLDA. All these models focus on the document-level side information such as document categories or review rating scores to supervise model learning. More variants of supervised topic models can be found in a number of applied domains, such as the aspect rating model (Titov and McDonald, 2008) for predicting ratings for each aspect of a hotel and the credit attribution model (Ramage et al., 2009) that associates each word with a label. In computer vision, several supervised topic models have been designed for understanding complex scene images (Sudderth et al., 2005; Fei-Fei and Perona, 2005; Li et al., 2009). Mimno and McCallum (2008) also proposed a topic model for considering document-level meta-data, for example, publication date and venue of a paper.

It is worth pointing out that among existing supervised topic models for incorporating side information, there are two classes of approaches, namely, *downstream supervised topic model* (DSTM) and *upstream supervised topic model* (USTM). In a DSTM the response variable is predicted based on the latent representation of the document, whereas in an USTM the response variable is being conditioned on to generate the latent representation of the document. Examples of USTM[3] include DiscLDA and the scene understanding models (Sudderth et al., 2005; Li et al., 2009), whereas

1. Yahoo directory can be found at `http://dir.yahoo.com/`.
2. Amazon Mechanical Turk can be found at `https://www.mturk.com/`.
3. The model presented by Mimno and McCallum (2008) is also an upstream model for incorporating document meta-features.

sLDA is an example of DSTM. Another distinction between existing supervised topic models is the training criterion, or more precisely, the choice of objective function in the optimization-based learning. The sLDA model is trained by maximizing the *joint* likelihood of the content data (e.g., text or image) and the responses (e.g., labeling or rating), whereas DiscLDA is trained by maximizing the *conditional* likelihood of the responses given contents. To the best of our knowledge, all the existing supervised topic models are trained by optimizing a likelihood-based objective; the highly successful margin-based objectives such as the hinge loss commonly used in discriminative models such as SVMs have never been employed.

In this paper, we propose *maximum entropy discrimination latent Dirichlet allocation* (MedLDA), a supervised topic model leveraging the maximum margin principle for making more effective use of side information during estimation of latent topical representations. Unlike existing supervised topic models mentioned above, MedLDA employs an arguably more discriminative max-margin learning technique within a probabilistic framework; and unlike the commonly adopted two-stage heuristic which first estimates a latent topic vector for each document using a topic model and then feeds them to another downstream prediction model, MedLDA integrates the mechanism behind the max-margin prediction models (e.g., SVMs) with the mechanism behind the hierarchical Bayesian topic models (e.g., LDA) under a unified constrained optimization framework. It employs a composite objective motivated by a tradeoff between two components—the negative log-likelihood of an underlying topic model which measures the goodness of fit for document contents, and a measure of prediction error on training data. It then seeks a regularized posterior distribution of the predictive function in a feasible space defined by a set of *expected* margin constraints generalized from the SVM-style margin constraints. The resultant inference problem is intractable; to circumvent this, we relax the original objective by using a variational upper bound of the negative log-likelihood and a surrogate convex loss function that upper bounds the training error. Our proposed approach builds on earlier developments in maximum entropy discrimination (MED) (Jaakkola et al., 1999; Jebara, 2001) and partially observed maximum entropy discrimination Markov network (PoMEN) (Zhu et al., 2008), but is significantly different and more powerful. In MedLDA, because of the influence of both the likelihood function over content data (e.g., text or image) and margin constraints induced by the side information, the discovery of latent topics is therefore coupled with the max-margin estimation of model parameters. This interplay can yield latent topical representations that are more discriminative and more suitable for supervised prediction tasks, as we demonstrate in the experimental section.

In fact, the methodology we develop in this paper generalizes beyond learning topic models; it can be applied to perform max-margin learning for various types of graphical models, including directed Bayesian networks, for example, LDA, sLDA and topic models with different priors such as the correlated topic models (Blei and Lafferty, 2005), and undirected Markov networks, for example, exponential family harmoniums (Welling et al., 2004) and replicated softmax (Salakhutdinov and Hinton, 2009) (See Section 4 for an extensive discussion). In this paper, we focus on the scenario of downstream supervised topic models, and we present several concrete examples of MedLDA that build on the original LDA to learn "discriminative topics" that allow more salient topic proportion vector $\boldsymbol{\theta}$ to be inferred for every document, evidenced by a significant improvement of accuracy of both regression and classification of documents based on the $\boldsymbol{\theta}$ resulted from MedLDA, over the $\boldsymbol{\theta}$ resulted from either the vanilla unsupervised LDA or even sLDA and alike. We also present an efficient and easy-to-implement variational approach for inference under MedLDA, with a running time comparable to that of an unsupervised LDA and lower than other likelihood-based supervised

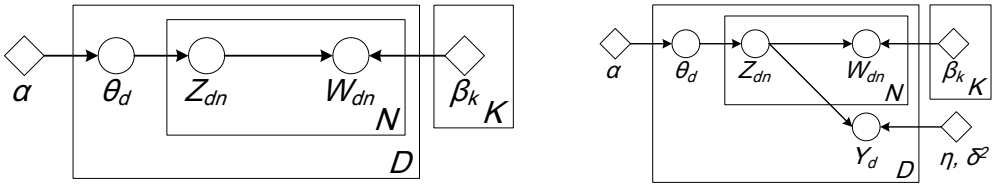


Figure 1: Graphical illustration of (Left) unsupervised LDA (Blei et al., 2003); and (Right) supervised LDA (Blei and McAuliffe, 2007).

LDAs. This advantage stems from the fact that MedLDA can directly optimize a margin-based loss instead of a likelihood-based one, and thereby avoids dealing with the normalization factor resultant from a full probabilistic generative formulation (e.g., sLDA), which generally makes learning harder.

The remainder of this paper is structured as follows. Section 2 introduces the preliminaries that are needed to present MedLDA. Section 3 presents MedLDA models for both regression and classification, together with efficient variational algorithms. Section 4 discusses the generalization of MedLDA to other topic models. Section 5 presents empirical studies of MedLDA. Finally, Section 6 concludes this paper with future research directions discussed. Part of the materials of this paper build on conference proceedings presented earlier in Zhu et al. (2009); Zhu and Xing (2010).

2. Preliminaries

We begin with a brief overview of the fundamentals of topic models, support vector machines, and the maximum entropy discrimination formulism (Jaakkola et al., 1999), which constitute the major building blocks of the proposed MedLDA model.

2.1 Unsupervised and Supervised Topic Models

Latent Dirichlet allocation (LDA) (Blei et al., 2003) is a hierarchical Bayesian model that projects a text document into a latent low dimensional space spanned by a set of automatically learned topical bases. Each topic is a multinomial distribution over M words in a given vocabulary. Let $\mathbf{w} = (w_1, \ldots, w_N)$ denote the vector of words appearing in a document (for notation simplicity, we suppress the indexing subscript of N and assume that all documents have the same length N); assume the number of topics to be an integer K, where K can be manually specified by a user or via cross-validation; and let $\boldsymbol{\beta} = [\boldsymbol{\beta}_1, \ldots, \boldsymbol{\beta}_K]$ denote the $M \times K$ matrix of topic distribution parameters, of which each $\boldsymbol{\beta}_k$ parameterizes a topic-specific multinomial word distribution. Under an LDA, the likelihood of a document d corresponds to the following generative process:

1. Draw a topic mixing proportion vector $\boldsymbol{\theta}_d$ according to a K-dimensional Dirichlet prior: $\boldsymbol{\theta}_d | \boldsymbol{\alpha} \sim \text{Dir}(\boldsymbol{\alpha})$;

2. For the n-th word in document d, where $1 \leq n \leq N$,

   (a) draw a topic assignment z_{dn} according to $\boldsymbol{\theta}_d$: $z_{dn} | \boldsymbol{\theta}_d \sim \text{Mult}(\boldsymbol{\theta}_d)$;

   (b) draw the word instance w_{dn} according to z_{dn}: $w_{dn} | z_{dn}, \boldsymbol{\beta} \sim \text{Mult}(\boldsymbol{\beta}_{z_{dn}})$,

where z_{dn} is a K-dimensional indicator vector (i.e., only one element is 1; all others are 0), an instance of the topic assignment random variable Z_{dn}. With a little abuse of notations, we use $\boldsymbol{\beta}_{z_{dn}}$ to denote the topic that is selected by the non-zero element of z_{dn}.

According to the above generative process, an *unsupervised* LDA defines the following joint distribution for a corpus $\mathcal{D}$ that contains D documents:

$$p(\{\boldsymbol{\theta}_d, \mathbf{z}_d\}, \mathbf{W}|\boldsymbol{\alpha}, \boldsymbol{\beta}) = \prod_{d=1}^{D} p(\boldsymbol{\theta}_d|\boldsymbol{\alpha}) \Big(\prod_{n=1}^{N} p(z_{dn}|\boldsymbol{\theta}_d) p(w_{dn}|z_{dn}, \boldsymbol{\beta}) \Big),$$

where $\mathbf{W} \triangleq \{\mathbf{w}_1; \cdots; \mathbf{w}_D\}$ denotes all the words in $\mathcal{D}$, and $\mathbf{z}_d \triangleq \{z_{d1}; \cdots; z_{dN}\}$. To estimate the unknown parameters $(\boldsymbol{\alpha}, \boldsymbol{\beta})$, and to infer the posterior distributions of latent variables $\{\boldsymbol{\theta}_d, \mathbf{z}_d\}$, an EM procedure is developed to maximize the marginal data likelihood[4] $p(\mathbf{W}|\boldsymbol{\alpha}, \boldsymbol{\beta})$. As we have stated, $\boldsymbol{\theta}_d$ represents the mixing proportion over K topics for document d, which can be treated as a low-dimensional representation of the document. Moreover, since the posterior of z_{dn} represents the probability distribution that word n is assigned to one of the K topics; the average topic assignment $\bar{\mathbf{z}}_d \triangleq \frac{1}{N}\sum_n z_{dn}$ can also be treated as a representation of the document, as commonly done in downstream supervised topic models (Blei and McAuliffe, 2007; Wang et al., 2009).

Due to intractability of the likelihood $p(\mathbf{W}|\boldsymbol{\alpha}, \boldsymbol{\beta})$, approximate inference algorithms based on variational (Blei et al., 2003) or Markov Chain Monte Carlo (MCMC) (Griffiths and Steyvers, 2004) methods have been widely used for parameter estimation and posterior inference under LDA. We focus on variational inference in this paper. The following variational bound for unsupervised LDA will be used later. Let $q(\{\boldsymbol{\theta}_d, \mathbf{z}_d\})$ represent a variational distribution that approximates the true model posterior $p(\{\boldsymbol{\theta}_d, \mathbf{z}_d\}|\boldsymbol{\alpha}, \boldsymbol{\beta}, \mathbf{W})$, one can derive a variational bound $\mathcal{L}^u(q; \boldsymbol{\alpha}, \boldsymbol{\beta})$ for the likelihood under unsupervised LDA:

$$\begin{aligned} \mathcal{L}^u(q; \boldsymbol{\alpha}, \boldsymbol{\beta}) &\triangleq -\mathbb{E}_q[\log p(\{\boldsymbol{\theta}_d, \mathbf{z}_d\}, \mathbf{W}|\boldsymbol{\alpha}, \boldsymbol{\beta})] - \mathcal{H}(q(\{\boldsymbol{\theta}_d, \mathbf{z}_d\})) \\ &\geq -\log p(\mathbf{W}|\boldsymbol{\alpha}, \boldsymbol{\beta}), \end{aligned} \tag{1}$$

where $\mathcal{H}(q) \triangleq -\mathbb{E}_q[\log q]$ is the entropy of q. By making some independence assumption (e.g., mean field) about q, $\mathcal{L}^u(q)$ can be efficiently optimized (Blei et al., 2003).

As we have stated, the unsupervised LDA described above does not use side information for learning topics and inferring topic vectors $\boldsymbol{\theta}$. In order to consider side information appropriately for discovering more predictive representations, supervised topic models (sLDA) (Blei and McAuliffe, 2007) introduce a response variable Y to LDA for each document, as shown in Figure 1. For regression, where $y \in \mathbb{R}$, the generative process of sLDA is similar to LDA, but with an additional step—*draw a response variable:* $y|\mathbf{z}_d, \boldsymbol{\eta}, \delta^2 \sim \mathcal{N}(\boldsymbol{\eta}^\top \bar{\mathbf{z}}_d, \delta^2)$ *for each document* d, where $\boldsymbol{\eta}$ is the regression weight vector and δ^2 is a noise variance parameter. Then, the joint distribution of sLDA is:

$$p(\{\boldsymbol{\theta}_d, \mathbf{z}_d\}, \mathbf{y}, \mathbf{W}|\boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\eta}, \delta^2) = \prod_{d=1}^{D} p(\boldsymbol{\theta}_d|\boldsymbol{\alpha}) \Big(\prod_{n=1}^{N} p(z_{dn}|\boldsymbol{\theta}_d) p(w_{dn}|z_{dn}, \boldsymbol{\beta}) \Big) p(y_d|\boldsymbol{\eta}^\top \bar{\mathbf{z}}_d, \delta^2), \tag{2}$$

4. We restrict ourselves to treat $\boldsymbol{\beta}$ as unknown parameters, as done in Blei and McAuliffe (2007) and Wang et al. (2009). Extension to a Bayesian treatment of $\boldsymbol{\beta}$ (i.e., by putting a prior over $\boldsymbol{\beta}$ and inferring its posterior) can be easily done both in LDA as shown in the literature (Blei et al., 2003) and in the MedLDA proposed here based on the regularized Bayesian inference framework (Zhu et al., 2011b). But a systematical discussion is beyond the scope of this paper.

where $\mathbf{y} \triangleq \{y_1; \cdots; y_D\}$. In this case, the joint likelihood is $p(\mathbf{y}, \mathbf{W}|\boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\eta}, \delta^2)$. Given a new document, the prediction is the expected response value

$$\hat{y} \triangleq \mathbb{E}[Y|\mathbf{w}, \boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\eta}, \delta^2] = \boldsymbol{\eta}^\top \mathbb{E}[\bar{Z}|\mathbf{w}, \boldsymbol{\alpha}, \boldsymbol{\beta}, \delta^2], \tag{3}$$

where the average topic assignment random variable $\bar{Z} \triangleq \frac{1}{N}\sum_n Z_n$ ($\bar{\mathbf{z}}$ is an instance of $\bar{Z}$), and the expectation is taken with respect to the posterior distribution of $\mathbf{Z} \triangleq \{Z_1; \cdots; Z_N\}$. However, exact inference is again intractable, and one can use the following variational upper bound $\mathcal{L}^s(q; \boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\eta}, \delta^2)$ for *supervised* sLDA for approximate inference:

$$\begin{aligned} \mathcal{L}^s(q; \boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\eta}, \delta^2) &\triangleq -\mathbb{E}_q[\log p(\{\boldsymbol{\theta}_d, \mathbf{z}_d\}, \mathbf{y}, \mathbf{W}|\boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\eta}, \delta^2)] - \mathcal{H}(q(\{\theta_d, \mathbf{z}_d\})) \\ &\geq -\log p(\mathbf{y}, \mathbf{W}|\boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\eta}, \delta^2). \end{aligned} \tag{4}$$

By changing the model of generating Y, sLDA can deal with other types of response variables, such as discrete ones for classification (Wang et al., 2009) using the multi-class logistic regression

$$p(y|\boldsymbol{\eta}, \mathbf{z}) = \frac{\exp(\boldsymbol{\eta}_y^\top \bar{\mathbf{z}})}{\sum_{y'} \exp(\boldsymbol{\eta}_{y'}^\top \bar{\mathbf{z}})}, \tag{5}$$

where $\boldsymbol{\eta}_y$ is the parameter vector associated with class label y. However, posterior inference in an sLDA classification model can be more challenging than that in the sLDA regression model. This is because the non-Gaussian probability distribution in Equation (5) is highly nonlinear of $\boldsymbol{\eta}$ and $\mathbf{z}$ and its normalization factor can make the topic assignments of different words in the same document strongly coupled. Variational methods were successfully used to approximate the normalization factor (Wang et al., 2009), but they can be computationally expensive as we shall demonstrate in the experimental section.

DiscLDA (Lacoste-Julien et al., 2008) is yet another supervised topic model for classification. DiscLDA is an upstream supervised topic model and as such the unknown parameter is the transformation matrix that is used to generate the document latent representations conditioned on the class label; and this transformation matrix is learned by maximizing the conditional marginal likelihood of the text given class labels.

This progress notwithstanding, to the best of our knowledge, current developments of supervised topic models have been solely built on a likelihood-driven probabilistic inference paradigm. The arguably sometimes more powerful max-margin based techniques widely used in learning discriminative models have not been exploited to learn supervised topic models. The main goal of this paper is to systematically investigate how the max-margin principe can be exploited inside a topic model to learn topics that are better at discriminating documents than current likelihood-driven learning achieves while retaining semantic interpretability as the later allows. For this purpose, below we briefly review the max-margin principle underlying a major technique built on this principle, the support vector machines.

2.2 Support Vector Machines

Max-margin methods, such as support vector machines (SVMs) (Vapnik, 1998) and max-margin Markov networks (M^3N) (Taskar et al., 2003), have been successfully applied to a wide range of discriminative problems such as document categorization and handwritten character recognition. It has been shown that such methods enjoy strong generalization guarantees (Vapnik, 1998; Taskar

et al., 2003). Depending on the nature of the response variable, the max-margin principle can be exploited in both classification and regression. Below we use document rating prediction as an example to recapitulate the ideas behind support vector regression (SVR) (Smola and Schölkopf, 2003), which we will shortly leverage to build our first instance of max-margin topic model.

Let $\mathcal{D} = \{(\mathbf{x}_1, y_1), \cdots, (\mathbf{x}_D, y_D)\}$ be a training set, where $\mathbf{x} \in \mathcal{X}$ are inputs such as document-feature vectors, and $y \in \mathbb{R}$ are response values such as user ratings. Using SVR, one obtains a function $h(\mathbf{x}) \in \mathcal{F}$ that makes at most ε deviation from the true response value y for each training example, and at the same time is as flat as possible. One common choice of the function family $\mathcal{F}$ is linear functions, that is, $h(\mathbf{x};\boldsymbol{\eta}) = \boldsymbol{\eta}^\top \mathbf{f}(\mathbf{x})$, where $\mathbf{f} = \{f_1, \cdots, f_I\}$ is a vector of feature functions $f_i : \mathcal{X} \to \mathbb{R}$, and $\boldsymbol{\eta}$ is the corresponding weight vector. Formally, the linear SVR finds an optimal linear function by solving the following constrained optimization problem:

$$\text{P0(SVR)}: \quad \min_{\boldsymbol{\eta}, \boldsymbol{\xi}, \boldsymbol{\xi}^*} \quad \frac{1}{2}\|\boldsymbol{\eta}\|_2^2 + C\sum_{d=1}^{D}(\xi_d + \xi_d^*)$$
$$\forall d, \text{ s.t.}: \quad \begin{cases} y_d - \boldsymbol{\eta}^\top \mathbf{f}(\mathbf{x}_d) \leq \varepsilon + \xi_d \\ -y_d + \boldsymbol{\eta}^\top \mathbf{f}(\mathbf{x}_d) \leq \varepsilon + \xi_d^* \\ \xi_d, \xi_d^* \geq 0 \end{cases},$$

where $\|\boldsymbol{\eta}\|_2 \triangleq \sqrt{\boldsymbol{\eta}^\top \boldsymbol{\eta}}$ is the ℓ_2-norm; $\boldsymbol{\xi}$ and $\boldsymbol{\xi}^*$ are slack variables that tolerate some errors in the training data; ε is a precision parameter; and C is a positive regularization constant. Problem P0 can be equivalently formulated as a regularized empirical loss minimization, where the loss is the so-called ε-insensitive loss (Smola and Schölkopf, 2003).

Under a standard SVR, P0 is a quadratic programming (QP) problem and can be easily solved in a Lagrangian dual formulation. Samples with non-zero lagrange multipliers are called support vectors, as in the SVM classification model. There exist several free packages for solving standard SVR, such as SVM-light (Joachims, 1999). We will use these methods as a sub-routine in our proposed approach, as we will detail in the sequel.

2.3 Maximum Entropy Discrimination

To unite the principles behind topic models and SVR, namely, Bayesian inference and max-margin learning, we employ a formalism known as *maximum entropy discrimination* (MED) (Jaakkola et al., 1999; Jebara, 2001), which learns a distribution of all possible regression/classification models that belong to a particular parametric family, subject to a set of margin-based constraints. For instance, the MED regression model, or simply MED^r, learns a distribution $q(\boldsymbol{\eta})$ through solving the following optimization problem:

$$\text{P1(MED}^\text{r}\text{)}: \quad \min_{q(\boldsymbol{\eta}), \boldsymbol{\xi}, \boldsymbol{\xi}^*} \quad KL(q(\boldsymbol{\eta}) \| p_0(\boldsymbol{\eta})) + C\sum_{d=1}^{D}(\xi_d + \xi_d^*)$$
$$\forall d, \text{ s.t.}: \quad \begin{cases} y_d - \mathbb{E}[\boldsymbol{\eta}]^\top \mathbf{f}(\mathbf{x}_d) \leq \varepsilon + \xi_d \\ -y_d + \mathbb{E}[\boldsymbol{\eta}]^\top \mathbf{f}(\mathbf{x}_d) \leq \varepsilon + \xi_d^* \\ \xi_d, \xi_d^* \geq 0 \end{cases},$$

where $p_0(\boldsymbol{\eta})$ is a prior distribution over the parameters and $KL(p\|q) \triangleq \mathbb{E}_p[\log(p/q)]$ is the Kullback-Leibler (KL) divergence.

As studied in Jebara (2001), this MED problem leads to an entropic-regularized posterior distribution of the SVR coefficients, $q(\boldsymbol{\eta})$; and the resultant predictor $\hat{y} = \mathbb{E}_{q(\boldsymbol{\eta})}[h(\mathbf{x};\boldsymbol{\eta})]$ enjoys several nice properties and subsumes the standard SVR as special cases when the prior $p_0(\eta)$ is standard normal (Jebara, 2001). Moreover, as shown in Zhu and Xing (2009); Zhu et al. (2011a), with different choices of the prior over $\boldsymbol{\eta}$, such as a sparsity-inducing Laplace or a nonparametric Dirichlet process, the resultant $q(\boldsymbol{\eta})$ can exhibit a wide variety of characteristics and are suitable for diverse utilities such as feature selection or learning complex non-linear discriminating functions. Finally, the recent developments of the maximum entropy discrimination Markov network (MaxEnDNet) (Zhu and Xing, 2009) and partially observed MaxEnDNet (PoMEN) (Zhu et al., 2008) have extended the basic MED to the much broader scenarios of learning structured prediction functions with or without latent variables.

To apply the MED idea to learn a supervised topic model, a major difficulty is the presence of heterogeneous latent variables in the topic models, such as the topic vector $\boldsymbol{\theta}$ and topic indicator Z. In the sequel, we present a novel formalism called *maximum entropy discrimination LDA* (MedLDA) that extends the basic MED to make this possible, and at the same time discovers latent discriminating topics present in the study corpus based on available discriminant side information.

3. MedLDA: Maximum Margin Supervised Topic Models

Now we present a new class of supervised topic models that explicitly employ labeling information in the context of document classification or regression, under a unified statistical framework that jointly optimizes over the cross entropy between a user supplied model prior and the aimed model posterior, and over the margin of ensuing predictive tasks based on the learned model. This is to contrast conventional heuristics that first learn a topic model, and then independently train a classifier such as SVM using the per-document topic vectors resultant from the first step as inputs. In such a heuristic, the document labels are never able to influence the way topics can be learned, and the per-document topic vectors are often found to be not strongly predictive (Xing et al., 2005).

3.1 Regressional MedLDA

We first consider the scenario where the numerical-valued rating of documents in the corpus is available, and our goal is to learn a supervised topic model specialized at predicting the rating of new documents through a regression function. We call this model a Regressional MedLDA, or simply, *MedLDA*^{r}.

Instead of learning a point estimate of regression coefficient $\boldsymbol{\eta}$ as in sLDA or SVR, we take the more general Bayesian-style (i.e., an averaging model) approach as in MED and learn a joint distribution[5] $q(\boldsymbol{\eta}, \mathbf{z})$ in a max-margin manner. For prediction, we take a weighted average over all the possible models (represented by $\boldsymbol{\eta}$) and latent topical representations $\mathbf{z}$, or more precisely, an expectation of the prediction over $q(\boldsymbol{\eta}, \mathbf{z})$, which is similar to that in Equation (3), but now over both $\boldsymbol{\eta}$ and $\mathbf{Z}$, rather than only over $\mathbf{Z}$:

$$\hat{y} \triangleq \mathbb{E}[Y|\mathbf{w}, \boldsymbol{\alpha}, \boldsymbol{\beta}, \delta^2] = \mathbb{E}[\boldsymbol{\eta}^\top \bar{Z}|\mathbf{w}, \boldsymbol{\alpha}, \boldsymbol{\beta}, \delta^2]. \tag{6}$$

Now, the question underlying the prediction rule (6) is how we can devise an appropriate objective function as well as constraints to learn a $q(\cdot)$ that leverages both the max-margin principle (for

5. In principle, we can perform Bayesian-style estimation for other parameters, like δ^2. For simplicity, we only consider $\boldsymbol{\eta}$ as a random variable in this paper.

strong predictivity) and the topic model architecture (for topic discovery). Below we begin with a simple reformulation of the sLDA that makes this possible.

3.1.1 MAX-MARGIN TRAINING OF SLDA

Without loss of generality, we let $q(\boldsymbol{\eta}, \mathbf{z}) = \int_{\boldsymbol{\theta}} q(\boldsymbol{\eta}) q(\mathbf{z}, \boldsymbol{\theta} | \boldsymbol{\eta})$, where $q(\boldsymbol{\eta})$ is the learned distribution of the predictive regression coefficient, and $q(\mathbf{z}, \boldsymbol{\theta} | \boldsymbol{\eta})$ is the learned distribution of the topic elements of the documents analogous to an sLDA-style topic model, but estimated from a different learning paradigm that leverages margin-based supervised training. As reviewed in Section 2.1, two good templates for $q(\mathbf{z}, \boldsymbol{\theta} | \boldsymbol{\eta})$ can be the original LDA or sLDA. For brevity, here we present a regressional MedLDA that uses the supervised sLDA as the underlying topic model. As we shall see in Section 3.2 and Appendix B, the underlying topic model can also be an unsupervised LDA.

Let $p_0(\boldsymbol{\eta})$ denote a prior distribution of $\boldsymbol{\eta}$, then MedLDA^r defines a joint distribution

$$p(\boldsymbol{\eta}, \{\boldsymbol{\theta}_d, \mathbf{z}_d\}, \mathbf{y}, \mathbf{W} | \boldsymbol{\alpha}, \boldsymbol{\beta}, \delta^2) = p_0(\boldsymbol{\eta}) p(\{\boldsymbol{\theta}_d, \mathbf{z}_d\}, \mathbf{y}, \mathbf{W} | \boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\eta}, \delta^2),$$

where the second factor has the same form as Equation (2) for sLDA, except that now $\boldsymbol{\eta}$ is a random variable and follows a prior $p_0(\boldsymbol{\eta})$. Accordingly, the likelihood $p(\mathbf{y}, \mathbf{W} | \boldsymbol{\alpha}, \boldsymbol{\beta}, \delta^2)$ is an expectation of the likelihood of sLDA under $p_0(\boldsymbol{\eta})$, which makes it even harder than in sLDA to directly optimize. Therefore, we choose to optimize a variational upper bound of the log-likelihood. We will discuss other approximation methods in Section 4.

Let $q(\boldsymbol{\eta}, \{\boldsymbol{\theta}_d, \mathbf{z}_d\})$ be a variational approximation to the posterior $p(\boldsymbol{\eta}, \{\boldsymbol{\theta}_d, \mathbf{z}_d\} | \boldsymbol{\alpha}, \boldsymbol{\beta}, \delta^2, \mathbf{y}, \mathbf{W})$. Then, an upper bound[6] $\mathcal{L}^{bs}(q; \boldsymbol{\alpha}, \boldsymbol{\beta}, \delta^2)$ of the negative log-likelihood is

$$\begin{aligned} \mathcal{L}^{bs}(q; \boldsymbol{\alpha}, \boldsymbol{\beta}, \delta^2) &\triangleq -\mathbb{E}_q[\log p(\boldsymbol{\eta}, \{\boldsymbol{\theta}_d, \mathbf{z}_d\}, \mathbf{y}, \mathbf{W} | \boldsymbol{\alpha}, \boldsymbol{\beta}, \delta^2)] - \mathcal{H}(q(\boldsymbol{\eta}, \{\boldsymbol{\theta}_d, \mathbf{z}_d\})) \\ &= KL(q(\boldsymbol{\eta}) \| p_0(\boldsymbol{\eta})) + \mathbb{E}_{q(\boldsymbol{\eta})}[\mathcal{L}^s]. \end{aligned} \tag{7}$$

We can see that the bound is also an expectation of sLDA's variational bound $\mathcal{L}^s$ in Equation (4). To derive Equation (7), we should note that the variational distribution for sLDA is "conditioned on" its model parameters, which include $\boldsymbol{\eta}$. Similarly, the distribution q in $\mathcal{L}^{bs}$ depends on the parameters $(\boldsymbol{\alpha}, \boldsymbol{\beta}, \delta^2)$. For notation clarity, we have omitted the explicit dependence on parameters in variational distributions.

Based on the MED principle and the variational bound in Equation (7), we define the learning problem of MedLDA^r as follows:

$$\text{P2(MedLDA}^r\text{)}: \min_{q, \boldsymbol{\alpha}, \boldsymbol{\beta}, \delta^2, \boldsymbol{\xi}, \boldsymbol{\xi}^*} \mathbb{E}_{q(\boldsymbol{\eta})}[\mathcal{L}^s(q; \boldsymbol{\alpha}, \boldsymbol{\beta}, \delta^2)] + KL(q(\boldsymbol{\eta}) \| p_0(\boldsymbol{\eta})) + C \sum_{d=1}^{D} (\xi_d + \xi_d^*)$$

$$\forall d, \text{ s.t.}: \begin{cases} y_d - \mathbb{E}[\boldsymbol{\eta}^\top \bar{Z}_d] \leq \varepsilon + \xi_d \\ -y_d + \mathbb{E}[\boldsymbol{\eta}^\top \bar{Z}_d] \leq \varepsilon + \xi_d^* \\ \xi_d, \xi_d^* \geq 0, \end{cases}$$

where $\boldsymbol{\xi}, \boldsymbol{\xi}^*$ are slack variables, and ε is a precision parameter as in SVR. The margin constraints in P2 are of the same form as those in P0, but in an expectation version because both the topic assignments Z and parameters $\boldsymbol{\eta}$ are latent random variables in MedLDA^r.

6. "bs" stands for "Bayesian Supervised".

It is easy to verify that at the optimum, at most one of ξ_d and ξ_d^* can be non-zero and $\xi_d + \xi_d^* = \max(0, |y_d - \mathbb{E}[\boldsymbol{\eta}^\top \bar{Z}_d]| - \varepsilon)$, which is known as ε-insensitive loss (Smola and Schölkopf, 2003), that is, if the current prediction $\hat{y}$ as in Equation (6) does not deviate from the true response value too much (i.e., less than ε), there is no loss; otherwise, a linear loss will be penalized. Mathematically, problem P2 can be equivalently written as a loss minimization problem without using slack variables:

$$\min_{q,\boldsymbol{\alpha},\boldsymbol{\beta},\delta^2} \mathcal{L}^{bs}(q;\boldsymbol{\alpha},\boldsymbol{\beta},\delta^2) + C\sum_{d=1}^{D} \max(0, |y_d - \mathbb{E}[\boldsymbol{\eta}^\top \bar{Z}_d]| - \varepsilon), \tag{8}$$

where the variational bound $\mathcal{L}^{bs}$ plays two roles—*regularization* and *maximum likelihood estimation*. Specifically, as shown in Equation (7), $\mathcal{L}^{bs}$ decomposes into two parts. The first part of KL-divergence is an entropic regularizer for $q(\boldsymbol{\eta})$; and the second term is an expected bound of the data likelihood, as we have discussed. Therefore, problem P2 is a joint maximum margin learning and maximum likelihood estimation (with appropriate regularization), and the two components are coupled by sharing latent topic assignments Z and parameters $\boldsymbol{\eta}$.

The rationale underlying MedLDA^r is that: by minimizing an integrated objective function, we aim to find a latent topical representation and a document-rating prediction function which, on one hand, can predict accurately on unseen data with a sufficient margin, and on the other hand, can explain the data well (i.e., minimizing a variational bound of the negative log-likelihood). The max-margin learning and topic discovery procedure are coupled together via the constraints, which are defined on the expectations of model parameters $\boldsymbol{\eta}$ and latent topical assignments Z. This interplay will yield a topical representation that could be more suitable for prediction tasks, as explained below and verified in experiments.

3.1.2 VARIATIONAL APPROXIMATION ALGORITHM FOR MEDLDA^r

Minimizing $\mathcal{L}^{bs}$ is intractable. Here, we use mean field methods (Jordan et al., 1999) widely employed in fitting LDA and sLDA to efficiently obtain an approximate q for problem P2. Specifically, we assume that q is a fully factorized mean-field approximation to p:

$$q(\boldsymbol{\eta}, \{\boldsymbol{\theta}_d, \mathbf{z}_d\}) = q(\boldsymbol{\eta}) \prod_{d=1}^{D} q(\boldsymbol{\theta}_d|\boldsymbol{\gamma}_d) \prod_{n=1}^{N} q(z_{dn}|\boldsymbol{\phi}_{dn}),$$

where $\boldsymbol{\gamma}_d$ is a K-dimensional vector of Dirichlet parameters and each $\boldsymbol{\phi}_{dn}$ parameterizes a multinomial distribution over K topics. It is easy to verify that:

$$\mathbb{E}[Z_{dn}] = \boldsymbol{\phi}_{dn}, \text{ and } \mathbb{E}[\boldsymbol{\eta}^\top \bar{Z}_d] = \mathbb{E}[\boldsymbol{\eta}]^\top (\frac{1}{N}\sum_{n=1}^{N} \boldsymbol{\phi}_{dn}).$$

Now, we develop a coordinate descent algorithm to solve the equivalent "unconstrained" formulation (8). The algorithm is outlined in Algorithm 1 and detailed below.

(1) *Solve for* $(\boldsymbol{\alpha}, \boldsymbol{\beta}, \delta^2)$ *and* $q(\boldsymbol{\eta})$: When $q(\{\boldsymbol{\theta}_d, \mathbf{z}_d\})$ is fixed, this substep (in an equivalent constrained form) is to solve

$$\min_{q(\boldsymbol{\eta}),\boldsymbol{\alpha},\boldsymbol{\beta},\delta^2,\boldsymbol{\xi},\boldsymbol{\xi}^*} \mathbb{E}_{q(\boldsymbol{\eta})}[\mathcal{L}^s(q;\boldsymbol{\alpha},\boldsymbol{\beta},\delta^2)] + KL(q(\boldsymbol{\eta})\|p_0(\boldsymbol{\eta})) + C\sum_{d=1}^{D}(\xi_d + \xi_d^*) \tag{9}$$

Algorithm 1 Variational MedLDA^r

Input: corpus $\mathcal{D} = \{(\mathbf{y}, \mathbf{W})\}$, constants C and ε, and topic number K.
Output: Dirichlet parameters $\boldsymbol{\gamma}$, posterior distribution $q(\boldsymbol{\eta})$, parameters $\boldsymbol{\alpha}$, $\boldsymbol{\beta}$ and δ^2.
repeat
  for $d = 1$ **to** D **do**
    Update $\boldsymbol{\gamma}_d$ as in Equation (13).
    for $n = 1$ **to** N **do**
      Update $\boldsymbol{\phi}_{dn}$ as in Equation (14).
    end for
  end for
  Solve the dual problem D2 to get $q(\boldsymbol{\eta})$, $\hat{\boldsymbol{\mu}}$ and $\hat{\boldsymbol{\mu}}^*$.
  Update $\boldsymbol{\beta}$ using Equation (10), and update δ^2 using Equation (11). Optimize $\boldsymbol{\alpha}$ with gradient descent or fix $\boldsymbol{\alpha}$ as $1/K$ times the ones vector.
until convergence

$$\forall d, \text{ s.t. : } \begin{cases} y_d - \mathbb{E}[\boldsymbol{\eta}^\top \bar{Z}_d] \le \varepsilon + \xi_d, & (\mu_d) \\ -y_d + \mathbb{E}[\boldsymbol{\eta}^\top \bar{Z}_d] \le \varepsilon + \xi_d^*, & (\mu_d^*) \\ \xi_d \ge 0, & (v_d) \\ \xi_d^* \ge 0, & (v_d^*), \end{cases}$$

where $\{\mu_d, \mu_d^*, v_d, v_d^*\}$ are lagrange multipliers. Since the margin constraints are not dependent on $(\boldsymbol{\alpha}, \boldsymbol{\beta}, \delta^2)$, we can solve for them using the same procedure as in sLDA, when $q(\boldsymbol{\eta})$ and $q(\{\boldsymbol{\theta}_d, \mathbf{z}_d\})$ are given. Specifically, for $\boldsymbol{\alpha}$, the same gradient descent method as in Blei et al. (2003) can be applied; for $\boldsymbol{\beta}$, the update equations are the same as for sLDA:

$$\beta_{kw} \propto \sum_{d=1}^{D} \sum_{n=1}^{N} \mathbb{I}(w_{dn} = w) \phi_{dn}^k, \tag{10}$$

where $\mathbb{I}(\cdot)$ is an indicator function that equals to 1 if the condition holds; otherwise 0; and for δ^2, the update rule is similar as that of sLDA but in an expected version, because $\boldsymbol{\eta}$ is a random variable:

$$\delta^2 = \frac{1}{D}\Big(\mathbf{y}^\top \mathbf{y} - 2\mathbf{y}^\top \mathbb{E}[A]\mathbb{E}[\boldsymbol{\eta}] + \mathbb{E}[\boldsymbol{\eta}^\top \mathbb{E}[A^\top A]\boldsymbol{\eta}]\Big), \tag{11}$$

where $\mathbb{E}[\boldsymbol{\eta}^\top \mathbb{E}[A^\top A]\boldsymbol{\eta}] = \mathrm{tr}(\mathbb{E}[A^\top A]\mathbb{E}[\boldsymbol{\eta}\boldsymbol{\eta}^\top])$, and A is a $D \times K$ matrix whose rows are the vectors $\bar{Z}_d^\top$.

Solving for $q(\boldsymbol{\eta})$ can be done using Lagrangian methods, but it is a bit more delicate. For brevity, we postpone the details of this step after we have finished presenting the overall procedure. We denote the optimum lagrange multipliers by $(\hat{\boldsymbol{\mu}}, \hat{\boldsymbol{\mu}}^*)$ and the optimum slack variables by $(\hat{\boldsymbol{\xi}}, \hat{\boldsymbol{\xi}}^*)$.

(2) *Solve for* $q(\{\boldsymbol{\theta}_d, \mathbf{z}_d\})$: By fixing $q(\boldsymbol{\eta})$ and $(\boldsymbol{\alpha}, \boldsymbol{\beta}, \delta^2)$, this substep (in an equivalent constrained form) is to solve

$$\min_{q(\{\boldsymbol{\theta}_d, \mathbf{z}_d\}), \boldsymbol{\xi}, \boldsymbol{\xi}^*} \mathbb{E}_{q(\boldsymbol{\eta})}[\mathcal{L}^s(q; \boldsymbol{\alpha}, \boldsymbol{\beta}, \delta^2)] + C \sum_{d=1}^{D} (\xi_d + \xi_d^*) \tag{12}$$

$$\forall d,\ \text{s.t.}: \begin{cases} y_d - \mathbb{E}[\boldsymbol{\eta}^\top \bar{Z}_d] \leq \varepsilon + \xi_d \\ -y_d + \mathbb{E}[\boldsymbol{\eta}^\top \bar{Z}_d] \leq \varepsilon + \xi_d^* \\ \xi_d, \xi_d^* \geq 0, \end{cases}$$

Since the constraints are not dependent on $\boldsymbol{\gamma}_d$ and $q(\boldsymbol{\eta})$ is also not directly connected with $\boldsymbol{\theta}_d$, we get the same update rule for $\boldsymbol{\gamma}_d$ as in sLDA:

$$\boldsymbol{\gamma}_d = \boldsymbol{\alpha} + \sum_{n=1}^{N} \boldsymbol{\phi}_{dn}. \tag{13}$$

For $q(\mathbf{z}_d)$, in theory, we can do the optimization to get the optimal solution of $\boldsymbol{\phi}$ and the corresponding optimal lagrange multipliers. But the full optimization would be expensive, especially considering that this sub-step is within the most inner iteration loop and it would be performed for many times. Here, we adopt an approximation strategy, which performs a single step update of $\boldsymbol{\phi}$, rather than a full optimization. Note that this one-step approximation could lead to a slight increase of the objective function during the iterations. Our empirical studies show that this increase is usually within an acceptable range. More specifically, we fix $(\boldsymbol{\xi}, \boldsymbol{\xi}^*)$ at $(\hat{\boldsymbol{\xi}}, \hat{\boldsymbol{\xi}}^*)$ (the optimum solution of the previous step) and set the lagrange multipliers to be $(\hat{\boldsymbol{\mu}}, \hat{\boldsymbol{\mu}}^*)$. Then, we have the closed-form update equation[7]

$$\boldsymbol{\phi}_{dn} \propto \exp\Big(\mathbb{E}[\log \boldsymbol{\theta}_d | \boldsymbol{\gamma}_d] + \log p(w_{dn}|\boldsymbol{\beta}) + \frac{y_d}{N\delta^2}\mathbb{E}[\boldsymbol{\eta}] - \frac{2\mathbb{E}[\boldsymbol{\eta}^\top \boldsymbol{\phi}_{d,-n}\boldsymbol{\eta}] + \mathbb{E}[\boldsymbol{\eta} \circ \boldsymbol{\eta}]}{2N^2\delta^2} + \frac{\mathbb{E}[\boldsymbol{\eta}]}{N}(\hat{\mu}_d - \hat{\mu}_d^*)\Big), \tag{14}$$

where $\boldsymbol{\phi}_{d,-n} \triangleq \sum_{i \neq n} \boldsymbol{\phi}_{di}$; $\boldsymbol{\eta} \circ \boldsymbol{\eta}$ is the element-wise product; and the result of exponentiating a vector is a vector of the exponentials of its corresponding components. Note that the first two terms in the exponential are the same as those in LDA.

Remark 1 *From the update rule of $\boldsymbol{\phi}$ in Equation (14), we can see that the essential differences between MedLDA^r and sLDA lie in the last three terms in the exponential of $\boldsymbol{\phi}_{dn}$. Firstly, the third and fourth terms are similar to those of sLDA, but in an expected version since we are learning the distribution $q(\boldsymbol{\eta})$ instead of a point estimate of $\boldsymbol{\eta}$. The second-order expectations $\mathbb{E}[\boldsymbol{\eta}^\top \boldsymbol{\phi}_{d,-n}\boldsymbol{\eta}]$ and $\mathbb{E}[\boldsymbol{\eta} \circ \boldsymbol{\eta}]$ mean that the co-variances of $\boldsymbol{\eta}$ (See Corollary 3 for an example) affect the distribution over topics. This makes our approach significantly different from a point estimation method, like sLDA, where no expectations or co-variances are involved in updating $\boldsymbol{\phi}_{dn}$. Secondly, the last term is from the max-margin regression formulation. For a document d, which lies on the decision boundary, that is, a support vector, either μ_d or μ_d^* is non-zero, and the last term biases $\boldsymbol{\phi}_{dn}$ towards a distribution that favors a more accurate prediction on the document. Moreover, the last term is fixed for words in the document and thus will directly affect the latent representation of the document, that is, $\boldsymbol{\gamma}_d$. Therefore, the latent representation $\boldsymbol{\theta}_d$ inferred under MedLDA^r can be more suitable for supervised prediction tasks. Our empirical studies further verify this, as we shall see in Section 5.*

7. Before we update $\boldsymbol{\phi}$, $(\hat{\boldsymbol{\mu}}, \hat{\boldsymbol{\mu}}^*)$ and $(\hat{\boldsymbol{\xi}}, \hat{\boldsymbol{\xi}}^*)$ satisfy the optimal conditions (e.g., KKT conditions) of problem (12). So, they are the initially optimal solutions. But after we have updated $\boldsymbol{\phi}$, the KKT conditions do not hold. This is the reason why our strategy of not updating $(\boldsymbol{\mu}, \boldsymbol{\mu}^*)$ and $(\boldsymbol{\xi}, \boldsymbol{\xi}^*)$ could lead to a slight increase of the objective function.

Now, we turn to the sub-step of solving for $q(\boldsymbol{\eta})$, as well as the slack variables and lagrange multipliers. Specifically, we have the following result.

Proposition 2 *For MedLDA^r, the optimum solution of $q(\boldsymbol{\eta})$ has the form:*

$$q(\boldsymbol{\eta}) = \frac{p_0(\boldsymbol{\eta})}{Z} \exp\Big(\boldsymbol{\eta}^\top \sum_{d=1}^{D} (\hat{\mu}_d - \hat{\mu}_d^* + \frac{y_d}{\delta^2})\mathbb{E}[\bar{Z}_d] - \boldsymbol{\eta}^\top \frac{\mathbb{E}[A^\top A]}{2\delta^2}\boldsymbol{\eta}\Big),$$

where $\mathbb{E}[A^\top A] = \sum_{d=1}^{D} \mathbb{E}[\bar{Z}_d \bar{Z}_d^\top]$, and $\mathbb{E}[\bar{Z}_d \bar{Z}_d^\top] = \frac{1}{N^2}(\sum_{n=1}^{N}\sum_{m\neq n} \boldsymbol{\phi}_{dn}\boldsymbol{\phi}_{dm}^\top + \sum_{n=1}^{N} \mathrm{diag}\{\boldsymbol{\phi}_{dn}\})$. The lagrange multipliers $(\hat{\boldsymbol{\mu}}, \hat{\boldsymbol{\mu}}^*)$ are the solution of the dual problem of (9):

$$\begin{aligned} \text{D2}: \quad & \max_{\boldsymbol{\mu},\boldsymbol{\mu}^*} \quad -\log Z - \varepsilon \sum_{d=1}^{D} (\mu_d + \mu_d^*) + \sum_{d=1}^{D} y_d(\mu_d - \mu_d^*) \\ & \forall d, \text{ s.t.}: \ \mu_d, \mu_d^* \in [0, C]. \end{aligned}$$

Proof (sketch) By setting the partial derivative of the Lagrangian functional over $q(\boldsymbol{\eta})$ equal to zero, we can get the solution of $q(\boldsymbol{\eta})$. Plugging $q(\boldsymbol{\eta})$ into the Lagrangian functional and solving for the optimal (v_d, v_d^*) and (ξ_d, ξ_d^*) as in the standard SVR to get the box constraints, we get the dual problem. ■

In MedLDA^r, we can choose different priors to introduce some regularization effects. For the standard normal prior, we have the following corollary:

Corollary 3 *Assume the prior $p_0(\boldsymbol{\eta}) = \mathcal{N}(0, I)$, where I is the identity matrix, then the optimum solution of $q(\boldsymbol{\eta})$ is*

$$q(\boldsymbol{\eta}) = \mathcal{N}(\boldsymbol{\lambda}, \Sigma),$$

where $\boldsymbol{\lambda} = \Sigma(\sum_{d=1}^{D}(\hat{\mu}_d - \hat{\mu}_d^* + \frac{y_d}{\delta^2})\mathbb{E}[\bar{Z}_d])$ is the mean and $\Sigma = (I + 1/\delta^2 \mathbb{E}[A^\top A])^{-1}$ is a $K \times K$ covariance matrix. The dual problem D2 is now:

$$\begin{aligned} & \max_{\boldsymbol{\mu},\boldsymbol{\mu}^*} \quad -\frac{1}{2}\boldsymbol{\omega}^\top \Sigma \boldsymbol{\omega} - \varepsilon \sum_{d=1}^{D} (\mu_d + \mu_d^*) + \sum_{d=1}^{D} y_d(\mu_d - \mu_d^*) \\ & \forall d, \text{ s.t.}: \ \mu_d, \mu_d^* \in [0, C], \end{aligned} \tag{15}$$

where $\boldsymbol{\omega} = \sum_{d=1}^{D}(\mu_d - \mu_d^* + \frac{y_d}{\delta^2})\mathbb{E}[\bar{Z}_d]$.

In the above Corollary, computation of Σ can be done robustly through Cholesky decomposition of $\delta^2 I + \mathbb{E}[A^\top A]$, an $O(K^3)$ procedure. Another example is the Laplace prior, which can lead to a shrinkage effect (Zhu and Xing, 2009) that is useful in sparse problems. In this paper, we focus on the normal prior and extension to the Laplace prior can be done similarly as in Zhu and Xing (2009). For the standard normal prior, the dual optimization problem is a QP problem and can be solved with any standard QP solvers, although they may not be so efficient. To leverage recent developments in learning support vector regression models, we first prove the following corollary:

Corollary 4 *Assume the prior $p_0(\boldsymbol{\eta}) = \mathcal{N}(0, I)$, then the mean $\boldsymbol{\lambda}$ of $q(\boldsymbol{\eta})$ in problem (9) is the optimum solution of the following problem:*

$$\min_{\boldsymbol{\lambda},\boldsymbol{\xi},\boldsymbol{\xi}^*} \quad \frac{1}{2}\boldsymbol{\lambda}^\top \Sigma^{-1}\boldsymbol{\lambda} - \boldsymbol{\lambda}^\top \Big(\sum_{d=1}^{D} \frac{y_d}{\delta^2}\mathbb{E}[\bar{Z}_d]\Big) + C\sum_{d=1}^{D}(\xi_d + \xi_d^*) \tag{16}$$

$$\forall d, \text{ s.t. : } \begin{cases} y_d - \boldsymbol{\lambda}^\top \mathbb{E}[\bar{Z}_d] \le \varepsilon + \xi_d \\ -y_d + \boldsymbol{\lambda}^\top \mathbb{E}[\bar{Z}_d] \le \varepsilon + \xi_d^* \\ \xi_d, \xi_d^* \ge 0 \end{cases}$$

Proof See Appendix A for details. ■

The above primal form can be re-formulated as a standard SVR problem. Specifically, we do Cholesky decomposition $\Sigma^{-1} = U^\top U$, where U is an upper triangular matrix with strict positive diagonal entries. Let $\boldsymbol{\nu} = \sum_{d=1}^{D} \frac{y_d}{\delta^2}\mathbb{E}[\bar{Z}_d]$, and we define $\boldsymbol{\lambda}' = U(\boldsymbol{\lambda} - \Sigma\boldsymbol{\nu})$; $y'_d = y_d - \boldsymbol{\nu}^\top \Sigma \mathbb{E}[\bar{Z}_d]$; and $\mathbf{x}_d = (U^{-1})^\top \mathbb{E}[\bar{Z}_d]$. Then, the above primal problem in Corollary 4 can be re-formulated as the following standard form:

$$\min_{\boldsymbol{\lambda}',\boldsymbol{\xi},\boldsymbol{\xi}^*} \quad \frac{1}{2}\|\boldsymbol{\lambda}'\|_2^2 + C\sum_{d=1}^{D}(\xi_d + \xi_d^*) \tag{17}$$

$$\forall d, \text{ s.t. : } \begin{cases} y'_d - (\boldsymbol{\lambda}')^\top \mathbf{x}_d \le \varepsilon + \xi_d \\ -y'_d + (\boldsymbol{\lambda}')^\top \mathbf{x}_d \le \varepsilon + \xi_d^* \\ \xi_d, \xi_d^* \ge 0 \end{cases}.$$

Then, we can solve the standard SVR problem using existing algorithms, such as the working set selection algorithm implemented in SVM-light (Joachims, 1999), to get the dual parameters[8] $\hat{\boldsymbol{\mu}}$ and $\hat{\boldsymbol{\mu}}^*$ (as well as slack variables $\hat{\boldsymbol{\xi}}$ and $\hat{\boldsymbol{\xi}}^*$), which are needed to infer $\boldsymbol{\phi}$ as defined in (14), and the primal parameters $\boldsymbol{\lambda}'$ which we use to get $\boldsymbol{\lambda}$ by doing a reverse transformation since $\boldsymbol{\lambda}' = U(\boldsymbol{\lambda} - \Sigma\boldsymbol{\nu})$ as defined above. The other lagrange multipliers, which are not explicitly involved in topic inference and estimation of $q(\boldsymbol{\eta})$, are solved according to KKT conditions.

3.2 Classificational MedLDA

Now, we present the MedLDA classification model, of which the discrete labels of the documents are available, and our goal is to learn a supervised topic model specialized at predicting the labels of new documents through a discriminant function. We call this model a Classificational MedLDA, or simply, *MedLDA^c*.

Denoting the discrete response variable by Y, for brevity, we only consider the multi-class classification, where y takes values from a finite set $\mathcal{C} \triangleq \{1, 2, \cdots, J\}$. The binary case, where $\mathcal{C} \triangleq \{+1, -1\}$, can be easily defined based on a binary SVM and the optimization problem can be solved similarly. For classification, if the latent topic assignments $\mathbf{z} \triangleq \{z_1; \cdots; z_N\}$ of all the words in a document are given, we define the *latent* linear discriminant function

$$F(y, \mathbf{z}, \boldsymbol{\eta}; \mathbf{w}) = \boldsymbol{\eta}_y^\top \bar{\mathbf{z}},$$

8. Not all existing solvers return the dual parameters $\hat{\boldsymbol{\mu}}$ and $\hat{\boldsymbol{\mu}}^*$. SVM-light is one nice package that provides both primal parameters $\boldsymbol{\lambda}'$ and the dual parameters. Note that the above transformation from (16) to (17) is done in the primal form and does not affect the solution of dual parameters of (15).

where $\bar{\mathbf{z}} \triangleq 1/N\sum_n z_n$, the same as in the case of MedLDA regression model; $\boldsymbol{\eta}_y$ is a class-specific K-dimensional parameter vector associated with class y; and $\boldsymbol{\eta}$ is a $|\mathcal{C}|K$-dimensional vector by stacking the elements of $\boldsymbol{\eta}_y$. Equivalently, F can be written as $F(y,\mathbf{z},\boldsymbol{\eta};\mathbf{w}) = \boldsymbol{\eta}^\top \mathbf{f}(y,\bar{\mathbf{z}})$, where $\mathbf{f}(y,\bar{\mathbf{z}})$ is a feature vector whose components from $(y-1)K+1$ to yK are those of the vector $\bar{\mathbf{z}}$ and all the others are 0.

However, we cannot directly use the latent function $F(y,\mathbf{z},\boldsymbol{\eta};\mathbf{w})$ to make prediction for an observed input $\mathbf{w}$ of a document because the topic assignments $\mathbf{z}$ are hidden variables. Here, we also treat $\boldsymbol{\eta}$ as a random vector and consider the general case to learn a distribution of $q(\boldsymbol{\eta})$. In order to deal with the uncertainty of $\mathbf{z}$ and $\boldsymbol{\eta}$, similar to MedLDA^r, we take the expectation over $q(\boldsymbol{\eta},\mathbf{z})$ and define the *effective* discriminant function

$$F(y;\mathbf{w}) = \mathbb{E}[F(y,\mathbf{Z},\boldsymbol{\eta};\mathbf{w})] = \mathbb{E}[\boldsymbol{\eta}^\top \mathbf{f}(y,\bar{Z})|\boldsymbol{\alpha},\boldsymbol{\beta},\mathbf{w}],$$

where $\mathbf{Z} \triangleq \{Z_1;\cdots;Z_N\}$ is the set of topic assignment random variables and $\bar{Z} \triangleq 1/N\sum_n Z_n$ is the average topic assignment random variable as defined before. Then, the prediction rule for multi-class classification is naturally

$$\hat{y} = \operatorname*{argmax}_{y\in\mathcal{C}} F(y;\mathbf{w}) = \operatorname*{argmax}_{y\in\mathcal{C}} \mathbb{E}[\boldsymbol{\eta}^\top \mathbf{f}(y,\bar{Z})|\boldsymbol{\alpha},\boldsymbol{\beta},\mathbf{w}]. \tag{18}$$

Our goal here is to learn an optimal set of parameters $(\boldsymbol{\alpha},\boldsymbol{\beta})$ and distribution $q(\boldsymbol{\eta})$. As in MedLDA^r, we have the option of using either a supervised sLDA (Wang et al., 2009) or an unsupervised LDA as a building block of MedLDA^c to discover latent topical representations. However, as we have discussed in Section 2.1 and shown by Wang et al. (2009) as well as Section 5.3.1, inference under sLDA can be harder and slower because the probability model of discrete Y in Equation (5) is highly nonlinear over $\boldsymbol{\eta}$ and Z, both of which are latent variables in our case, and its normalization factor strongly couples the topic assignments of different words in the same document. Therefore, in this paper we focus on the case of using an LDA that only models the likelihood of document contents $\mathbf{W}$ but not document label Y as the underlying topic model to discover latent representations Z. Even with this likelihood model, document labels can still influence topic learning and inference because they induce margin constraints pertinent to the topical distributions. As we shall see, the resultant MedLDA classification model can be easily and efficiently learned by using existing high-performance SVM solvers. Moreover, since the goal of max-margin learning is to directly minimize a hinge loss (i.e., an upper bound of the empirical loss), we do not need a normalized distribution model for response variables Y.

3.2.1 MAX-MARGIN LEARNING OF LDA FOR CLASSIFICATION

The LDA component inside the MedLDA^c defines a likelihood function $p(\mathbf{W}|\boldsymbol{\alpha},\boldsymbol{\beta})$ over the corpus $\mathcal{D}$, which is known to be intractable. Therefore, we choose to optimize its variational bound $\mathcal{L}^u(q;\boldsymbol{\alpha},\boldsymbol{\beta})$ in Equation (1), which facilitates efficient approximation algorithms. The integrated problem of discovering latent topical representations and learning a distribution of classifiers is defined as follows:

$$\text{P3(MedLDA}^c\text{)}: \min_{q,q(\boldsymbol{\eta}),\boldsymbol{\alpha},\boldsymbol{\beta},\boldsymbol{\xi}} \quad \mathcal{L}^u(q;\boldsymbol{\alpha},\boldsymbol{\beta}) + KL(q(\boldsymbol{\eta})||p_0(\boldsymbol{\eta})) + \frac{C}{D}\sum_{d=1}^{D}\xi_d$$

$$\forall d,\ y\in\mathcal{C},\ \text{s.t.}: \quad \begin{cases} \mathbb{E}[\boldsymbol{\eta}^\top \Delta\mathbf{f}_d(y)] \geq \Delta\ell_d(y) - \xi_d \\ \xi_d \geq 0, \end{cases}$$

where q denotes the variational distribution $q(\{\boldsymbol{\theta}_d, \mathbf{z}_d\})$; $\Delta\ell_d(y)$ is a non-negative cost function (e.g., 0/1 cost as typically used in SVMs) that measures how different the prediction y is from the true class label y_d; $\Delta\mathbf{f}_d(y) \triangleq \mathbf{f}(y_d, \bar{Z}_d) - \mathbf{f}(y, \bar{Z}_d)$; and $\boldsymbol{\xi}$ are slack variables.[9] It is typically assumed that $\Delta\ell_d(y_d) = 0$, that is, no cost for a correct prediction. Finally,

$$\mathbb{E}[\boldsymbol{\eta}^\top \Delta\mathbf{f}_d(y)] = F(y_d; \mathbf{w}_d) - F(y; \mathbf{w}_d)$$

is the "*expected* margin" by which the true label y_d is favored over a prediction y.

Note that we have taken a full expectation to define $F(y; \mathbf{w})$, instead of taking the mode as done in latent SVMs (Felzenszwalb et al., 2010; Yu and Joachims, 2009), because expectation is a nice linear functional of the distributions under which it is taken, whereas taking the mode involves the highly nonlinear *argmax* function for discrete Z, which could lead to a harder inference task. Furthermore, due to the same reason to avoid dealing with a highly nonlinear discriminant function, we did not adopt the method in Jebara (2001) either, which uses log-likelihood ratio to define the discriminant function when considering latent variables in MED. Specifically, in our case, the max-margin constraints of the standard MED would be

$$\forall d,\ \forall y \in \mathcal{C},\ \log \frac{p(y_d|\mathbf{w}_d, \boldsymbol{\alpha}, \boldsymbol{\beta})}{p(y|\mathbf{w}_d, \boldsymbol{\alpha}, \boldsymbol{\beta})} \geq \Delta\ell_d(y) - \xi_d,$$

which are highly nonlinear due to the complex form of the marginal likelihood $p(y|\mathbf{w}_d, \boldsymbol{\alpha}, \boldsymbol{\beta}) = \int_{\boldsymbol{\theta}_d} \sum_{\mathbf{z}_d} p(y, \boldsymbol{\theta}_d, \mathbf{z}_d|\mathbf{w}_d, \boldsymbol{\alpha}, \boldsymbol{\beta})$. Our linear expectation operator is an effective tool to deal with latent variables in the context of maximum margin learning. In fact, besides the present work, we have successfully applied this operator to other challenging settings of learning latent variable structured prediction models with nontrivial dependence structures among output variables (Zhu et al., 2008) and learning nonparametric Bayesian models (Zhu et al., 2011b,a). These expected margin constraints also make MedLDA^c fundamentally different from the mixture of conditional max-entropy models (Pavlov et al., 2003), where constraints are based on moment matching, that is, empirical expectations of features equal to their model expectations.

By setting $\boldsymbol{\xi}$ to their optimum solutions, that is, $\xi_d = \max_y(\Delta\ell_d(y) - \mathbb{E}[\boldsymbol{\eta}^\top \Delta\mathbf{f}_d(y)])$, we can rewrite problem P3 in the form of regularized empirical loss minimization

$$\min_{q, q(\boldsymbol{\eta}), \boldsymbol{\alpha}, \boldsymbol{\beta}} \mathcal{L}^u(q; \boldsymbol{\alpha}, \boldsymbol{\beta}) + KL(q(\boldsymbol{\eta})||p_0(\boldsymbol{\eta})) + C\mathcal{R}(q, q(\boldsymbol{\eta})), \tag{19}$$

where

$$\mathcal{R}(q, q(\boldsymbol{\eta})) \triangleq \frac{1}{D}\sum_{d=1}^{D} \max_{y \in \mathcal{C}}(\Delta\ell_d(y) - \mathbb{E}[\boldsymbol{\eta}^\top \Delta\mathbf{f}_d(y)])$$

is an upper bound of the training error of the prediction rule in Equation (18) and C is again the regularization constant. However, different from MedLDA^r, which uses a Bayesian supervised sLDA as the underlying likelihood model, here the variational bound $\mathcal{L}^u$ does not contain a cross-entropy term on $q(\boldsymbol{\eta})$ for its regularization (as in $\mathcal{L}^{bs}$ in Equation (7)). Therefore, we include the KL-divergence in problem P3 as an explicit entropic regularizer for the distribution $q(\boldsymbol{\eta})$.

9. Since multi-class SVM is a special case of max-margin Markov networks, we follow the common conventions and use the same notations as in structured max-margin methods (Taskar et al., 2003; Joachims et al., 2009).

The rationale underlying MedLDA^c is similar to that of MedLDA^r, that is, we want to find latent topical representations $q(\{\boldsymbol{\theta}_d, \mathbf{z}_d\})$ and a model parameter distribution $q(\boldsymbol{\eta})$ which on one hand tend to predict as accurate as possible on training data, while on the other hand tend to explain the data well. The two parts are closely coupled by the expected margin constraints.

3.2.2 VARIATIONAL ALGORITHM FOR MEDLDA^c

As in MedLDA^r, we make the fully-factorized mean field assumption that

$$q(\{\boldsymbol{\theta}_d, \mathbf{z}_d\}) = \prod_{d=1}^{D} q(\boldsymbol{\theta}_d|\boldsymbol{\gamma}_d) \prod_{n=1}^{N} q(z_{dn}|\boldsymbol{\phi}_{dn}),$$

where $\boldsymbol{\gamma}_d$ and $\boldsymbol{\phi}_{dn}$ are variational parameters, having the same meaning as in MedLDA^r. Then, we have $\mathbb{E}[\boldsymbol{\eta}^\top \mathbf{f}(y, \bar{Z}_d)] = \mathbb{E}[\boldsymbol{\eta}]^\top \mathbf{f}(y, 1/N \sum_{n=1}^{N} \boldsymbol{\phi}_{dn})$. We develop a similar coordinate descent algorithm to solve the "unconstrained" formulation in (19). Since the constraints in P3 are not on $\boldsymbol{\gamma}$, $\boldsymbol{\alpha}$ or $\boldsymbol{\beta}$, their update rules are the same as in the case of MedLDA^r and we omit the details here. Below, we explain the optimization over $q(\{\mathbf{z}_d\})$ and $q(\boldsymbol{\eta})$ and show the insights of the max-margin topic model.

Optimize over $q(\boldsymbol{\eta})$: As in the case of regression, we have the following solution:

Corollary 5 *When* $(\boldsymbol{\alpha}, \boldsymbol{\beta})$ *and* $q(\{\boldsymbol{\theta}_d, \mathbf{z}_d\})$ *are fixed, the optimum solution* $q(\boldsymbol{\eta})$ *of MedLDA*^{c} *in problem P3 has the form:*

$$q(\boldsymbol{\eta}) = \frac{1}{Z} p_0(\boldsymbol{\eta}) \exp\Big(\boldsymbol{\eta}^\top (\sum_{d=1}^{D} \sum_{y \in C} \hat{\mu}_d^y \mathbb{E}[\Delta \mathbf{f}_d(y)])\Big),$$

where the lagrange multipliers $\hat{\boldsymbol{\mu}}$ *are the optimum solution of the dual problem:*

$$\text{D3}: \quad \max_{\boldsymbol{\mu}} -\log Z + \sum_{d=1}^{D} \sum_{y \in C} \mu_d^y \Delta \ell_d(y)$$
$$\forall d, \text{ s.t. }: \sum_{y \in C} \mu_d^y \in [0, \frac{C}{D}],$$

Again, we can choose different priors in MedLDA^c for different regularization effects. We consider the normal prior in this paper. For the standard normal prior $p_0(\boldsymbol{\eta}) = \mathcal{N}(0, I)$, we can get: $q(\boldsymbol{\eta})$ is a normal with a shifted mean, that is, $q(\boldsymbol{\eta}) = \mathcal{N}(\boldsymbol{\lambda}, I)$, where $\boldsymbol{\lambda} = \sum_{d=1}^{D} \sum_{y \in C} \mu_d^y \mathbb{E}[\Delta \mathbf{f}_d(y)]$, and the dual problem D3 thus becomes the same as the dual problem of a standard multi-class SVM (Crammer and Singer, 2001):

$$\max_{\boldsymbol{\mu}} \quad -\frac{1}{2} \| \sum_{d=1}^{D} \sum_{y \in C} \mu_d^y \mathbb{E}[\Delta \mathbf{f}_d(y)] \|_2^2 + \sum_{d=1}^{D} \sum_{y \in C} \mu_d^y \Delta \ell_d(y) \tag{20}$$
$$\forall d, \text{ s.t. }: \quad \sum_{y \in C} \mu_d^y \in [0, \frac{C}{D}].$$

The primal form of problem (20) is

$$\min_{\boldsymbol{\lambda}, \boldsymbol{\xi}} \quad \frac{1}{2} \|\boldsymbol{\lambda}\|_2^2 + \frac{C}{D} \sum_{d=1}^{D} \xi_d$$
$$\forall d, \ \forall y \in C, \text{ s.t. }: \quad \begin{cases} \boldsymbol{\lambda}^\top \mathbb{E}[\Delta \mathbf{f}_d(y)] \geq \Delta \ell_d(y) - \xi_d \\ \xi_d \geq 0. \end{cases}$$

Optimize over $q(\{\mathbf{z}_d\})$: again, since q is fully factorized, we can perform the optimization on each document separately. We have

$$\boldsymbol{\phi}_{dn} \propto \exp\Big(\mathbb{E}[\log \boldsymbol{\theta}_d|\boldsymbol{\gamma}_d] + \log p(w_{dn}|\boldsymbol{\beta}) + \frac{1}{N}\sum_{y\in C}\hat{\mu}_d^y \mathbb{E}[\boldsymbol{\eta}_{y_d} - \boldsymbol{\eta}_y]\Big), \tag{21}$$

where we can see that the first two terms in Equation (21) are the same as in unsupervised LDA (Blei et al., 2003), and the last term is due to the max-margin formulation of P3 and reflects our intuition that the discovered latent topical representation is influenced by the margin constraints. For those examples that are on the decision boundary, that is, support vectors, their associated lagrange multipliers are non-zero and thus the last term acts as a regularizer that biases the model towards discovering latent representations that tend to make more accurate prediction on these difficult examples. Moreover, this term is fixed for words in the document and thus will directly affect the latent representation of the document (i.e., $\boldsymbol{\gamma}_d$) and therefore leads to a discriminative latent representation. As we shall see in Section 5, such an estimate is more suitable for the classification task: for instance, MedLDA^c needs much fewer support vectors than the max-margin classifiers that are built on raw text or the topical representations discovered by LDA.

The above formulation of MedLDA^c has a slack variable associated with each document. This is known as the *n-slack* formulation (Joachims et al., 2009). Another equivalent formulation, which can be more efficiently solved, is the so called *1-slack* formulation. The 1-slack MedLDA^c can be written as follows

$$\begin{aligned}
\text{P4(1-slack MedLDA}^c\text{)}: \quad & \min_{q, q(\boldsymbol{\eta}), \boldsymbol{\alpha}, \boldsymbol{\beta}, \xi} \quad \mathcal{L}^u(q) + KL(q(\boldsymbol{\eta})||p_0(\boldsymbol{\eta})) + C\xi \\
& \forall(\bar{y}_1, \cdots, \bar{y}_D), \text{ s.t.}: \quad \begin{cases} \frac{1}{D}\sum_{d=1}^D \mathbb{E}[\boldsymbol{\eta}^\top \Delta\mathbf{f}_d(\bar{y}_d)] \geq \frac{1}{D}\sum_{d=1}^D \Delta\ell_d(\bar{y}_d) - \xi \\ \xi \geq 0. \end{cases}
\end{aligned}$$

By using the above developed variational algorithm and the cutting plane algorithm for solving the 1-slack as well as n-slack multi-class SVMs (Joachims et al., 2009), which is implemented in the SVM^{struct} package,[10] we can solve the 1-slack or n-slack MedLDA^c model efficiently, as we shall see in Section 5.3.1. SVM^{struct} provides the solutions of the primal parameters $\boldsymbol{\lambda}$ as well as the dual parameters $\boldsymbol{\mu}$, which are needed to do inference.

4. MedTM: A General Framework

We have presented two variants of MedLDA for discovering predictive latent topical representations of documents, as well as learning discriminating topics from the corpus; and we have shown that the underlying topic model that defines data likelihood can be either a supervised or an unsupervised LDA. In fact, the likelihood component of MedLDA can be any other form of generative topic model, such as correlated topic models (Blei and Lafferty, 2005), or latent space Markov random fields, such as exponential family harmoniums (Welling et al., 2004; Xing et al., 2005; Chen et al., 2010). The same principle can also be applied to upstream latent topic models, which have been widely used in computer vision applications (Sudderth et al., 2005; Fei-Fei and Perona, 2005; Zhu et al., 2010). In this section, we formulate a general framework of applying the max-margin principle to learn discriminative latent topic models when supervising side information is available, and we discuss more insights on developing approximate inference algorithms.

10. SVM^{struct} can be found at `http://svmlight.joachims.org/svm\_multiclass.html`.

Formally, a *maximum entropy discrimination topic model* (MedTM) consists of two components—an underlying topic model that fits observed data and a MED max-margin model that performs prediction. In an MedTM, we distinguish two types of latent variables—we use Υ to denote the parameters of the model pertaining to the prediction task (e.g., $\boldsymbol{\eta}$ in sLDA), and H to denote the topic assignment and mixing variables (e.g., $\mathbf{z}$ and $\boldsymbol{\theta}$). Let Ψ denote the parameters of the underlying topic model (e.g., the Dirichlet parameter $\boldsymbol{\alpha}$ and topics $\boldsymbol{\beta}$). Then, $p(\mathcal{D}|\Psi)$ is the marginal data likelihood of the corpus $\mathcal{D}$, which may or may not include the supervising side information depending on choice of specific form of the underlying topic model.

As discussed before, for a general topic model, $p(\mathcal{D}|\Psi)$ is intractable, therefore a generic variational method can be employed. Let $q(\Upsilon, H)$ be a variational distribution to approximate the posterior $p(\Upsilon, H|\mathcal{D}, \Psi)$. By the properties of KL-divergence, the following equality holds if we do not make any restricting assumption of $q(\Upsilon, H)$

$$\begin{aligned}
-\log p(\mathcal{D}|\Psi) &= \min_{q(\Upsilon,H)} \Big(-\mathbb{E}_{q(\Upsilon,H)}[\log p(\Upsilon, H, \mathcal{D}|\Psi)] - \mathcal{H}(q(\Upsilon, H)) \Big) \\
&= \min_{q(\Upsilon,H)} \Big(\mathbb{E}_{q(\Upsilon)} \Big[-\mathbb{E}_{q(H|\Upsilon)}[\log p(H, \mathcal{D}|\Psi, \Upsilon)] - \mathcal{H}(q(H|\Upsilon)) \Big] + KL(q(\Upsilon)\|p_0(\Upsilon)) \Big),
\end{aligned}$$

where $p_0(\Upsilon)$ is the prior distribution of Υ. Let us define

$$\mathcal{L}^t(q(H|\Upsilon); \Psi, \Upsilon) \triangleq -\mathbb{E}_{q(H|\Upsilon)}[\log p(H, \mathcal{D}|\Psi, \Upsilon)] - \mathcal{H}(q(H|\Upsilon)).$$

Then, $\mathcal{L}^t(q(H|\Upsilon); \Psi, \Upsilon)$ is the variational bound of the data likelihood associated with the underlying topic model. For instance, when the underlying topic model is supervised sLDA, $\mathcal{L}^t$ reduces to $\mathcal{L}^s$, as we discussed in Equation (7). When the underlying topic model is unsupervised LDA, the corpus $\mathcal{D}$ only contains document contents, and $p(H, \mathcal{D}|\Psi, \Upsilon) = p(H, \mathcal{D}|\Psi)$. The reduction of $\mathcal{L}^t$ to $\mathcal{L}^u$ needs a simplifying assumption that $q(\Upsilon, H) = q(\Upsilon)q(H)$ (in fact, much stricter assumptions on q are usually needed to make the learning of MedLDA^c tractable).

Mathematically, we define MedTM as solving the following entropic-regularized problem:

$$\begin{aligned}
\text{P5(MedTM)}: \min_{q(\Upsilon,H),\Psi,\boldsymbol{\xi}} \; & \mathbb{E}_{q(\Upsilon)} \Big[\mathcal{L}^t(q(H|\Upsilon); \Psi, \Upsilon) \Big] + KL(q(\Upsilon)\|p_0(\Upsilon)) + U(\boldsymbol{\xi}) \\
\text{s.t.}: \; & q(\Upsilon, H) \text{ satisfies the expected margin constraints.}
\end{aligned}$$

where U is a convex function over slack variables, such as $U(\boldsymbol{\xi}) = \frac{C}{D}\sum_d \xi_d$ in MedLDA^c. As we have discussed in Section 3.2.1, by using the linear expectation operator, our expected margin constraints are different from and simpler than those derived using a log-likelihood ratio function in the standard MED with latent variables (Jebara, 2001).

This formulation allows efficient approximate inference to be developed. In general, the difficulty of solving the optimization problem of MedTM lies in two aspects. First, the data likelihood or its equivalent variational form as involved in the objective function is generally intractable to compute if we do not make any restricting assumption about $q(\Upsilon, H)$. Second, the posterior inference (e.g., in LDA) as required in evaluating the margin constraints is generally intractable. Based on recent developments on learning latent topic models, two commonly used approaches can be applied to get an approximate solution to P5(MedTM), namely, Markov Chain Monte Carlo (MCMC) (Griffiths and Steyvers, 2004) and variational (Blei et al., 2003; Teh et al., 2006) methods. For variational methods, which are our focus in this paper, we need to make some additional restricting

assumptions, such as the commonly used mean field assumption, about the distribution $q(\Upsilon, H)$. Then, P5 can be efficiently solved with a coordinate descent procedure, similar to what we have done for MedLDA^r and MedLDA^c. For MCMC methods, the difference lies in sampling from the distribution $q(\Upsilon, H)$ under margin constraints—evaluating the expected margin constraints is easy once we obtain samples from the posterior. Several approaches were proposed to deal with the problem of sampling from a distribution under some constraints such as Schofield (2007), Griffiths (2002), Rodriguez-Yam et al. (2004) and Damien and Walker (2001) to name a few, and we plan to investigate their suitability to our case in the future.

Finally, based on the recent extensions of MED to the structured prediction setting (Zhu and Xing, 2009; Zhu et al., 2008), the basic principle of MedLDA can be similarly extended to perform structured prediction, where multiple response variables are predicted simultaneously and thus their mutual dependencies can be exploited to achieve globally consistent and optimal predictions. Likelihood based structured prediction latent topic models have been developed in different scenarios, such as image annotation (He and Zemel, 2008) and statistical machine translation (Zhao and Xing, 2007). Extension of MedLDA to the structured prediction setting could provide a promising alternative for such problems.

5. Experiments

In this section, we provide qualitative as well as quantitative evaluation of MedLDA on topic estimation, document classification and regression. For MedLDA and other topic models (except DiscLDA whose implementation details are explained in footnote 14), we optimize the K-dimensional Dirichlet parameters $\boldsymbol{\alpha}$ using the Newton-Raphson method (Blei et al., 2003). For initialization, we set $\boldsymbol{\phi}$ to be uniform and each topic $\boldsymbol{\beta}_k$ to be a uniform distribution plus a very small random noise, and the posterior mean of $\boldsymbol{\eta}$ to be zero. We have published our implementation on the website: *http://www.ml-thu.net/∼jun/software.html*. In all the experimental results, by default, we also report the standard deviation for a topic model with five randomly initialized runs.

5.1 Topic Estimation

We begin with an empirical assessment of topic estimation by MedLDA on the 20 Newsgroups data set with a standard list of stop words[11] removed. The data set contains about 20,000 postings in 20 related categories. We compare with unsupervised LDA.[12] We fit the data set to a 110-topic MedLDA^c model, which exploits the supervising category information, and a 110-topic unsupervised LDA, which ignores category information.

Figure 2 shows the 2D embedding of the inferred topic proportions $\boldsymbol{\theta}$ (approximated by the inferred variational posterior means) by MedLDA^c and LDA using the t-SNE stochastic neighborhood embedding (van der Maaten and Hinton, 2008) method, where each dot represents a document and each color-shape pair represents a category. Visually, the max-margin based MedLDA^c produces a better grouping and separation of the documents in different categories. In contrast, unsupervised LDA does not produce a well separated embedding, and documents in different categories tend to mix together. Intuitively, a well-separated representation is more discriminative for document categorization. This is further empirically supported in Section 5.2. Note that a similar embedding

11. Stop word list can be found at `http://mallet.cs.umass.edu/`.
12. We implemented LDA based on the public variational inference code by Dr. David Blei, using same data structures as MedLDA for fair comparison.

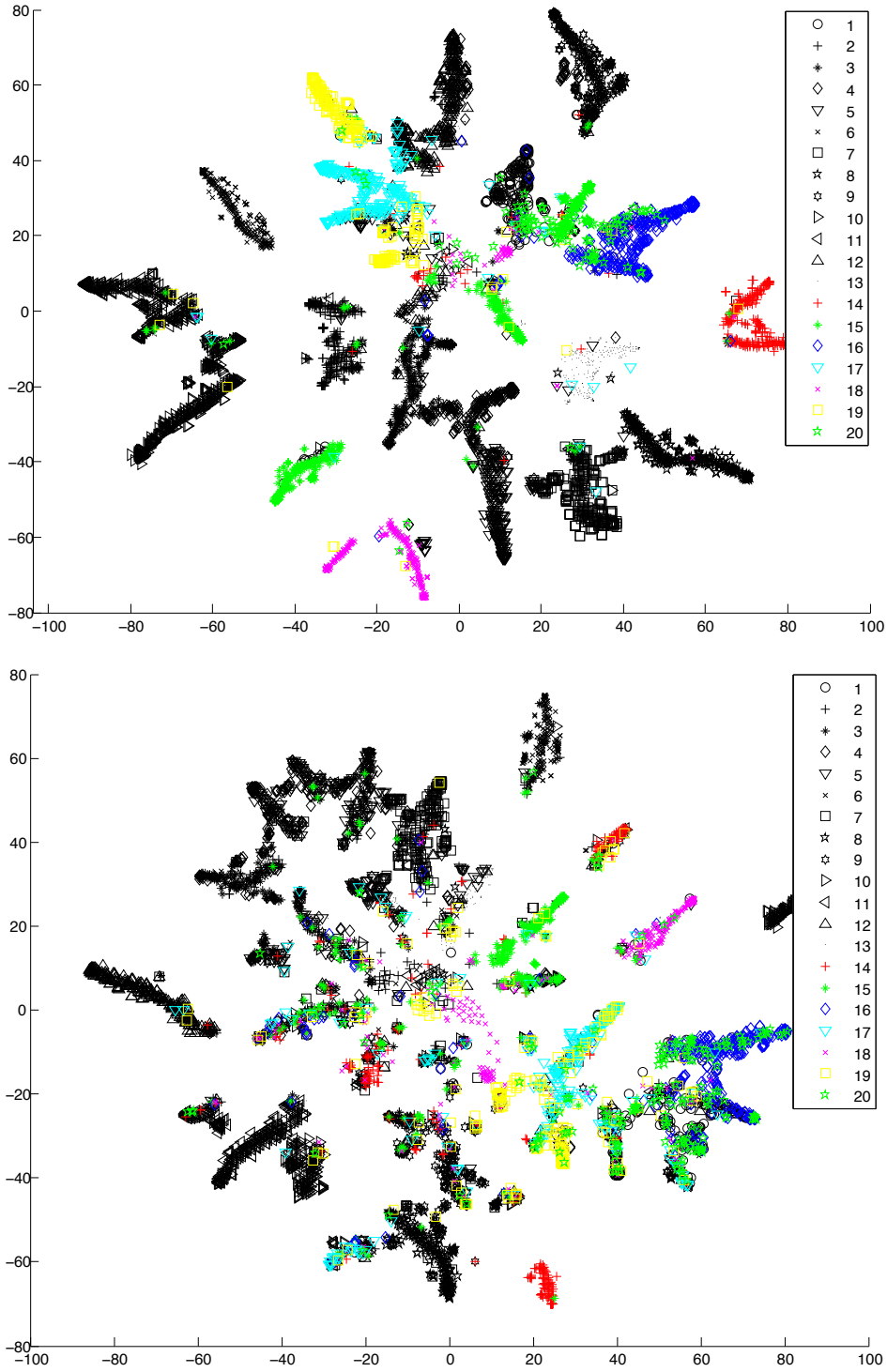


Figure 2: t-SNE 2D embedding of the topical representation by: MedLDA^c (above) and unsupervised LDA (below). The mapping between each index and category name can be found in: http://people.csail.mit.edu/jrennie/20Newsgroups/.

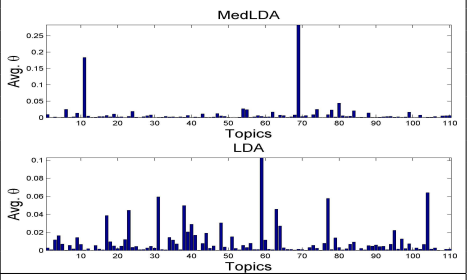
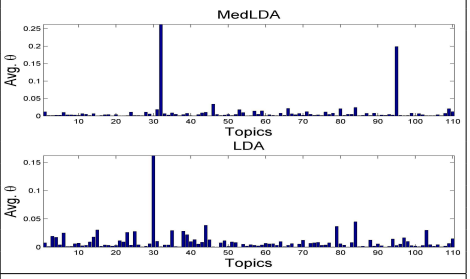
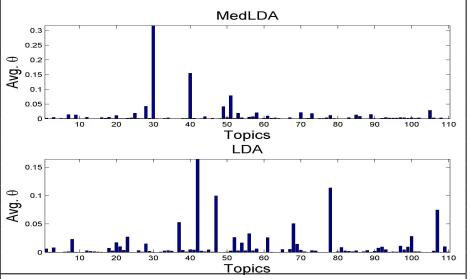
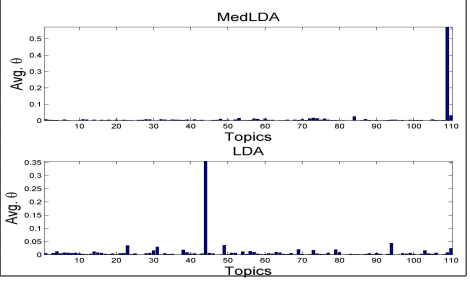
Class	MedLDA			LDA			Average θ per class
	T 69	T 11	T 80	T 59	T 104	T 31	
comp.graphics	image	graphics	db	image	ftp	card	
	jpeg	image	key	jpeg	pub	monitor	
	gif	data	chip	color	graphics	dos	
	file	ftp	encryption	file	mail	video	
	color	software	clipper	gif	version	apple	
	files	pub	system	images	tar	windows	
	bit	mail	government	format	file	drivers	
	images	package	keys	bit	information	vga	
	format	fax	law	files	send	cards	
	program	images	escrow	display	server	graphics	
	T 32	T 95	T 46	T 30	T 84	T 44	
sci.electronics	ground	audio	source	power	water	sale	
	wire	output	rs	ground	energy	price	
	power	input	time	wire	air	offer	
	wiring	signal	john	circuit	nuclear	shipping	
	don	chip	cycle	supply	loop	sell	
	current	high	low	voltage	hot	interested	
	circuit	data	dixie	current	cold	mail	
	neutral	mhz	dog	wiring	cooling	condition	
	writes	time	weeks	signal	heat	email	
	work	good	face	cable	temperature	cd	
	T 30	T 40	T 51	T 42	T 78	T 47	
politics.mideast	israel	turkish	israel	israel	jews	armenian	
	israeli	armenian	lebanese	israeli	jewish	turkish	
	jews	armenians	israeli	peace	israel	armenians	
	arab	armenia	lebanon	writes	israeli	armenia	
	writes	people	people	article	arab	turks	
	people	turks	attacks	arab	people	genocide	
	article	greek	soldiers	war	arabs	russian	
	jewish	turkey	villages	lebanese	center	soviet	
	state	government	peace	lebanon	jew	people	
	rights	soviet	writes	people	nazi	muslim	
	T 109	T 110	T 84	T 44	T 94	T 49	
misc.forsale	sale	drive	mac	sale	don	drive	
	price	scsi	apple	price	mail	scsi	
	shipping	mb	monitor	offer	call	disk	
	offer	drives	bit	shipping	package	hard	
	mail	controller	mhz	sell	writes	mb	
	condition	disk	card	interested	send	drives	
	interested	ide	video	mail	number	ide	
	sell	hard	speed	condition	ve	controller	
	email	bus	memory	email	hotel	floppy	
	dos	system	system	cd	credit	system	

Figure 3: Top topics under each class as discovered by the MedLDA and LDA models.

was presented by Lacoste-Julien et al. (2008), where the transformation matrix in their model is pre-designed. The results of MedLDA^c in Figure 2 are *automatically* learned.

It is also interesting to examine the discovered topics and their relevance to class labels. In Figure 3 we show the top topics in four example categories as discovered by both MedLDA^c and LDA. Here, the semantic meaning of each topic is represented by the first 10 high probability words.

To visually illustrate the discriminative power of the latent representations, that is, the topic proportion vector $\boldsymbol{\theta}$ of documents, we illustrate and compare the per-class distribution over topics for each model at the right side of Figure 3. This distribution is computed by averaging the expected topic vector of the documents in each class. We can see that MedLDA^c yields sharper, sparser and fast decaying per-class distributions over topics. For the documents in different categories, we

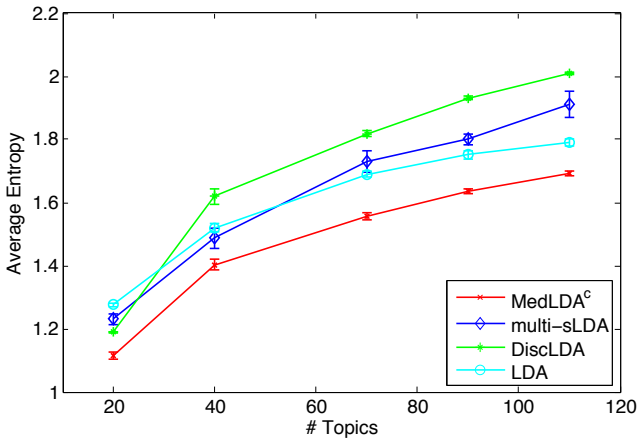


Figure 4: The average entropy of $\boldsymbol{\theta}$ over documents of different topic models on 20 Newsgroups data.

can see that their per-class average distributions over topics are very different, which suggests that the topical representations by MedLDA^c have a good discrimination power. Also, the sharper and sparser representations by MedLDA^c can result in a simpler max-margin classifier (e.g., with fewer support vectors), as we shall see in Section 5.2.1. All these observations suggest that the topical representations discovered by MedLDA^c have a better discriminative power and are more suitable for prediction tasks (Please see Section 5.2 for prediction performance). This behavior of MedLDA^c is in fact due to the regularization effect enforced over $\boldsymbol{\phi}$ as shown in Equation (21). On the other hand, LDA seems to discover topics that model the fine details of documents, possibly at the cost of achieving weaker discrimination power (i.e., it discovers different variations of the same topic which results in a flat per-class distribution over topics). For instance, in the class *comp.graphics*, MedLDA^c mainly models documents in this class using two salient, discriminative topics (T69 and T11) whereas LDA results in a much flatter distribution. Moreover, in the cases where LDA and MedLDA^c discover comparably the same set of topics in a given class (like *politics.mideast* and *misc.forsale*), MedLDA^c results in a sharper low dimensional representation.

A quantitative measure for the sparsity or sharpness of the distributions over topics is the entropy. We compute the entropy of the inferred topic proportion for each document and take the average over the corpus. Here, we compare MedLDA^c with unsupervised LDA, supervised sLDA for multi-class classification (multi-sLDA)[13] (Wang et al., 2009) and DiscLDA[14] (Lacoste-Julien

13. We thank the authors for providing their implementation, on which we made necessary slight modifications, for example, improving the time efficiency and optimizing $\boldsymbol{\alpha}$.

14. DiscLDA is a conditional model that uses class-specific topics and shared topics. Since the code is not publicly available, we implemented an in-house version by following the same strategy in the original paper and share K_1 topics across classes and allocate K_0 topics to each class, where $K_1 = 2K_0$, and we varied $K_0 = \{1, 2, \cdots\}$. We should note here that Lacoste-Julien et al. (2008); Lacoste-Julien (2009) gave an optimization algorithm for learning the topic structure (i.e., a transformation matrix), however since the code is not available, we resorted to one of the fixed splitting strategies mentioned in the paper. Moreover, for the multi-class case, the authors only reported results using the same fixed splitting strategy we mentioned above. For the number of iterations for training and inference, we followed Lacoste-Julien (2009). Moreover, following Lacoste-Julien (2009) and personal communication with the first author, we used symmetric Dirichlet priors on $\boldsymbol{\beta}$ and $\boldsymbol{\theta}$, and set the Dirichlet parameters at 0.01 and $0.1/(K_0+K_1)$, respectively.

et al., 2008). For DiscLDA, as in the original paper, we fix the transformation matrix and set it to be diagonally sparse. We use the standard training/testing split[15] to fit the models on training data and infer the topic distributions on testing documents. Figure 4 shows the average entropy of different models on testing documents when different topic numbers are chosen. For DiscLDA, we set the class-specific topic number $K_0 = 1, 2, 3, 4, 5$ and correspondingly $K = 22, 44, 66, 88, 110$. We can see that MedLDA^c yields the smallest entropy, which indicates that the probability mass is concentrated on quite a few topics, consistent with the observations in Figure 3. In contrast, for unsupervised LDA, the probability mass is more uniformly distributed on many topics (again consistent with Figure 3), which results in a higher entropy. For DiscLDA, although the transformation matrix is designed to be diagonally sparse, the distributions over the class-specific topics and shared topics are flat. Therefore, the entropy is also high. Using automatically learned transition matrices might improve the sparsity of DiscLDA.

5.2 Prediction Accuracy

In this subsection, we provide a quantitative evaluation of MedLDA on prediction performance for both document classification and regression.

5.2.1 CLASSIFICATION

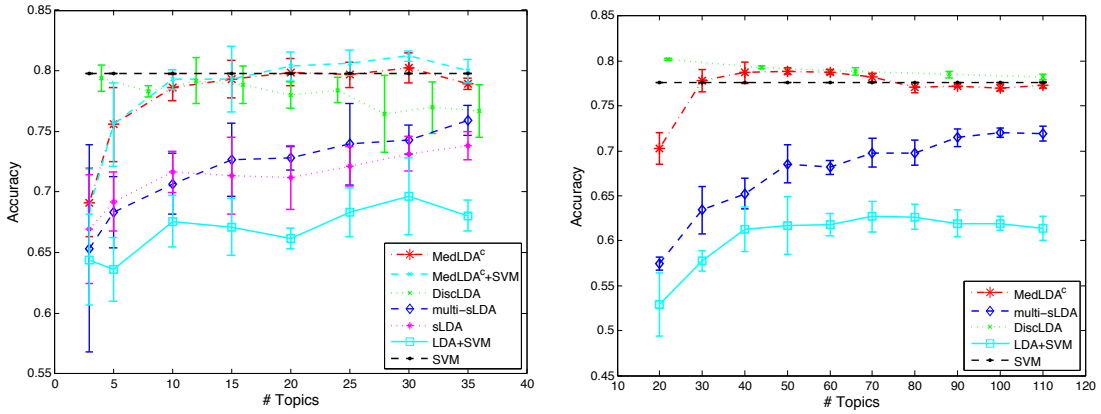
We perform binary and multi-class classification on the 20 Newsgroup data set. To obtain a baseline, we first fit all the data to an LDA model, and then use the latent representation of the training[16] documents as features to build a binary or multi-class SVM classifier. We denote this baseline by *LDA+SVM*.

Binary Classification: As Lacoste-Julien et al. (2008) did, the binary classification is to distinguish postings of the newsgroup *alt.atheism* and the postings of the group *talk.religion.misc*. The training set contains 856 documents with a split of 480/376 over the two categories, and the test set contains 569 documents with a split of 318/251 over the two categories. Therefore, the *naïve baseline* that predicts the most frequent category for all test documents has accuracy 0.672.

We compare the binary MedLDA^c with supervised LDA, DiscLDA, LDA+SVM, and the standard binary SVM built on raw text features. For supervised LDA, we use both the regression model (sLDA) (Blei and McAuliffe, 2007) and the multi-class classification model (multi-sLDA) (Wang et al., 2009). For the sLDA regression model, we fit it using the binary representation (0/1) of the classes, and use a threshold 0.5 to make prediction. For MedLDA^c, to see whether a second-stage max-margin classifier can improve the performance, we also build a method of *MedLDA^c+SVM*, similar to LDA+SVM. For DiscLDA, we fix the transition matrix. Automatically learning the transition matrix can yield slightly better results, as reported by Lacoste-Julien (2009). For all the above methods that use the class label information, they are fit *ONLY* on the training data.

We use the SVM-light (Joachims, 1999), which provides both primal and dual parameters, to build SVM classifiers and to estimate the posterior mean of $\boldsymbol{\eta}$ in MedLDA^c. The parameter C is chosen via 5 fold cross-validation during training from $\{k^2 : k = 1, \cdots, 8\}$. For each model, we run the experiments for 5 times and take the average as the final results. The prediction accuracy of different models with respect to the number of topics is shown in Figure 5(a). For DiscLDA, we

15. Split can be found at `http://people.csail.mit.edu/jrennie/20Newsgroups/`.
16. We use the training/testing split in `http://people.csail.mit.edu/jrennie/20Newsgroups/`.



(a) (b)

Figure 5: Classification accuracy of different models for: (a) binary and (b) multi-class classification on the 20 Newsgroup data.

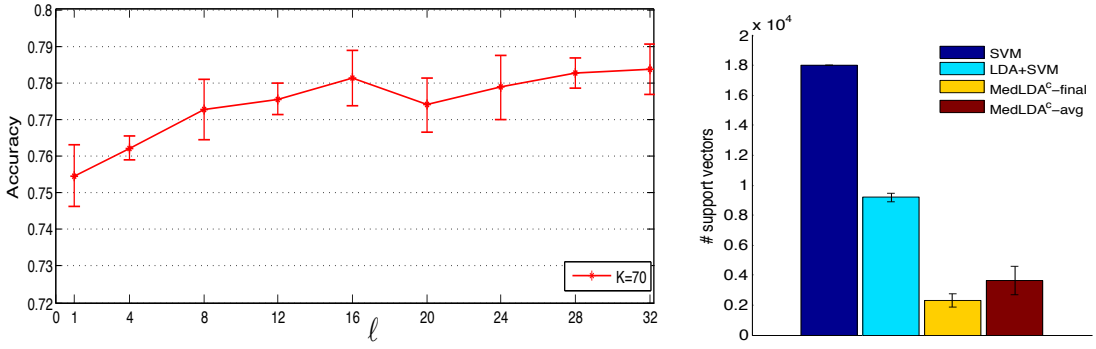
follow Lacoste-Julien et al. (2008) to set $K = 2K_0 + K_1$, where K_0 is the number of class-specific topics and K_1 is the number of shared topics, and $K_1 = 2K_0$. Here, we set $K_0 = 1, \cdots, 8, 10$.

We can see that the max-margin MedLDA^c performs better than the likelihood-based downstream models, include multi-sLDA, sLDA, and the baseline LDA+SVM. The best performances of the two discriminative models (i.e., MedLDA^c and DiscLDA) are comparable. However, MedLDA^c is easier to learn and faster in testing, as we shall see in Section 5.3.2. Moreover, the different approximate inference algorithms used in MedLDA^c (i.e., variational approximation) and DiscLDA (i.e., Monte Carlo sampling methods) can also make the performance different. In our alternative implementation using collapsed variational inference (Teh et al., 2006) method for MedLDA^c (preliminary results in preparation for submission), we were able to achieve slightly better results. However, the collapsed variational method is much more expensive. Finally, since MedLDA^c already integrates the max-margin principle into its training, our conjecture is that the combination of MedLDA^c and SVM does not further improve the performance much on this task. We believe that the slight differences between MedLDA^c and MedLDA^c+SVM are due to the tuning of regularization parameters. For efficiency, we do not change the regularization constant C during training MedLDA^c. The performance of MedLDA^c would be improved if we select a good C in different iterations because the data representation is changing.

Multi-class Classification: We perform multi-class classification on 20 Newsgroups with all the 20 categories. The data set has a balanced distribution over the categories. For the test set, which contains 7505 documents in total, the smallest category has 251 documents and the largest category has 399 documents. For the training set, which contains 11269 documents, the smallest and the largest categories contain 376 and 599 documents, respectively. Therefore, the *naïve baseline* that predicts the most frequent category for all the test documents has the classification accuracy 0.0532.

We compare MedLDA^c with LDA+SVM, multi-sLDA, DiscLDA, and the standard multi-class SVM built on raw text. We use the SVM^{struct} package with a cost function as $\Delta\ell_d(y) \triangleq \ell\mathbb{I}(y \neq y_d)$ to solve the sub-step of learning $q(\boldsymbol{\eta})$ and build the SVM classifiers for LDA+SVM. The parameter ℓ is selected with 5 fold cross-validation.[17] The average results as well as standard deviations over

17. The traditional $0/1$ cost does not yield the best results. In most cases, the selected ℓ's are around 16.



(a) (b)

Figure 6: (a) Sensitivity to the cost parameter ℓ for the MedLDA^c; and (b) the number of support vectors for n-slack multi-class SVM, LDA+SVM, and n-slack MedLDA^c. For MedLDA^c, we show both the number of support vectors at the final iteration and the average number during training.

5 randomly initialized runs are shown in Figure 5(b). For DiscLDA, we use the same equation as in Lacoste-Julien et al. (2008) to set the number of topics and set $K_0 = 1, \cdots, 5$. We can see that all the supervised topic models discover more predictive topical representations for classification, and the discriminative max-margin MedLDA^c and DiscLDA perform comparably, slightly better than the standard multi-class SVM (about 0.013 ± 0.003 improvement in accuracy). However, as we have stated and will show in Section 5.3.2, MedLDA^c is faster in testing than DiscLDA. As we shall see shortly, MedLDA^c needs much fewer support vectors than standard SVM.

Figure 6(a) shows the multi-class classification accuracy on the 20 Newsgroups data set for MedLDA^c with 70 topics. We show the results with ℓ manually set at $1, 4, 8, 12, \cdots, 32$. We can see that although the default $0/1$-cost works well for MedLDA^c, we can get better accuracy if we use a larger cost for penalizing wrong predictions. The performance is quite stable when ℓ is set to be larger than 8. The reason why ℓ affects the performance is that ℓ as well as C control: 1) the scale of the posterior mean of $\boldsymbol{\eta}$ and the Lagrangian multipliers $\boldsymbol{\mu}$, whose dot-product regularizes the topic mixing proportions in Equation (21); and 2) the goodness of fit of the MED large-margin classifier on the data (see Joachims et al., 2009, for another practical example that uses $0/\ell$-cost, where ℓ is set at 100). For practical reasons, we only try a small subset of candidate C values in parameter search, which can also influence the difference on performance in Figure 6(a). Performing very careful parameter search on C could possibly shrink the difference. Finally, for a small ℓ (e.g., 1 for the standard $0/1$-cost), we usually need a large C in order to obtain good performance. But our empirical experience with SVM^{struct} shows that the multi-class SVM with a larger C (and smaller ℓ) is typically more expensive to train than the SVM with a larger ℓ (and smaller C). That is one reason why we choose to use a large ℓ.

Figure 6(b) shows the number of support vectors for MedLDA^c, LDA+SVM, and the multi-class SVM built on raw text features, which are high-dimensional ($\sim$60,000 dimension for 20 Newsgroup data) and sparse. Here we consider the traditional n-slack formulation of multi-class SVM and n-slack MedLDA^c using the SVM^{struct} package, where a support vector corresponds to a document-label pair. For MedLDA^c and LDA+SVM, we set $K = 70$. For MedLDA^c, we report both the number of support vectors at the final iteration and the average number of support vectors over all iterations.

We can see that both MedLDA^c and LDA+SVM generally need much fewer support vectors than the standard SVM on raw text. The major reason is that both MedLDA^c and LDA+SVM uses a much lower dimensional and more compact representation for each document. Moreover, MedLDA^c needs (about 4 times) fewer support vectors than LDA+SVM. This could be because MedLDA^c make use of both text contents and the supervising class labels in the training data and its estimated topics tend to be more discriminative when being used to infer the latent topical representations of documents, that is, using these latent representations by MedLDA^c, the documents in different categories are more likely to be well-separated, and therefore the max-margin classifier is simpler (i.e., needs fewer support vectors). This observation is consistent with what we have observed on the per-class distributions over topics in Figure 3. Finally, we observed that about 32% of the support vectors in MedLDA^c are also the support vectors in multi-class SVM on the raw features.

5.2.2 REGRESSION

We first evaluate MedLDA^r on the movie review data set used by Blei and McAuliffe (2007), which contains 5006 documents and comprises 1.6M words, with a 5000-term vocabulary chosen by tf-idf. The data set was compiled from the one provided by Pang and Lee (2005). As Blei and McAuliffe (2007) did, we take logs of the response values to make them approximately normal. We compare MedLDA^r with unsupervised LDA, supervised sLDA, MedLDA^r_p—a MedLDA regression model which uses unsupervised LDA as the underlying topic model (Please see Appendix B for details), and the linear SVR that uses the empirical word frequency as input features. For LDA, we use its low dimensional representation of documents as input features to a linear SVR and denote this method by *LDA+SVR*. The evaluation criterion is predictive R^2 (pR^2), which is defined as one minus the mean squared error divided by the data variance (Blei and McAuliffe, 2007), specifically,

$$pR^2 = 1 - \frac{\sum_{d=1}^{D}(y_d - \hat{y}_d)^2}{\sum_{d=1}^{D}(y_d - \bar{y})^2},$$

where y_d and $\hat{y}_d$ are the true and estimated response values of document d, respectively; and $\bar{y}$ is the mean of true response values on the whole data set. When we report pR^2, by default it is computed on the testing data set. Note that the *naïve baseline* that predicts the mean response value for all documents (i.e., $\forall d,\ \hat{y}_d = \bar{y}$) will have 0 on pR^2. Any method that have a positive pR^2 performs better than the naïve baseline.

Figure 7 shows the average results as well as standard deviations over 5 randomly initialized runs, together with the per-word likelihood. For MedLDA and SVR, we fix the precision $\varepsilon = 1e^{-3}$ and select C via cross-validation during training. We can see that the supervised MedLDA and sLDA can get better results than unsupervised LDA, which ignores supervised responses during discovering topical representations, and the linear SVR regression model. By using max-margin learning, MedLDA^r can get slightly better results than the likelihood-based sLDA, especially when the number of topics is small (e.g., ≤ 15). Indeed, when the number of topics is small, the latent representation of sLDA alone does not result in a highly separable problem, thus the integration of max-margin training helps in discovering a more discriminative latent representation using the same number of topics. In fact, the number of support vectors (i.e., documents that have at least one non-zero lagrange multiplier) decreases dramatically at $T = 15$ and stays nearly the same for $T > 15$, which with reference to Equation (14) explains why the relative improvement over sLDA decreased as T increases. This behavior suggests that MedLDA^r can discover more predictive latent structures for *difficult*, non-separable regression problems.

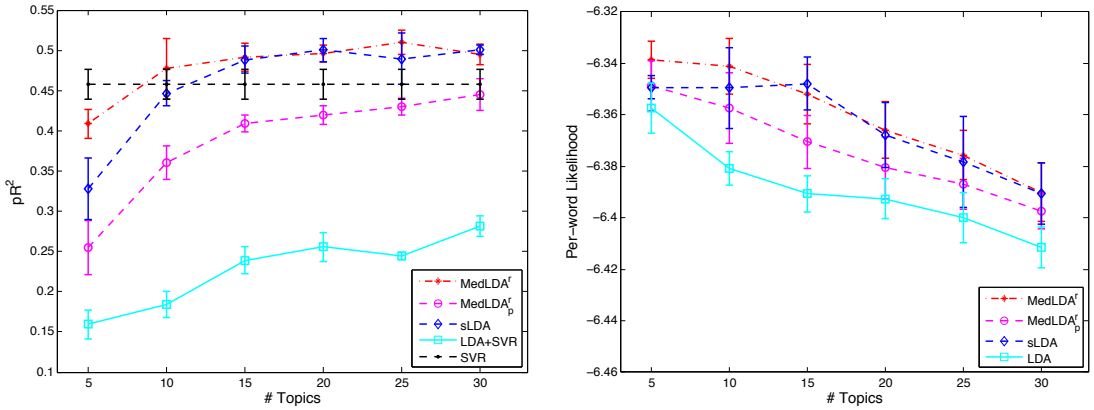


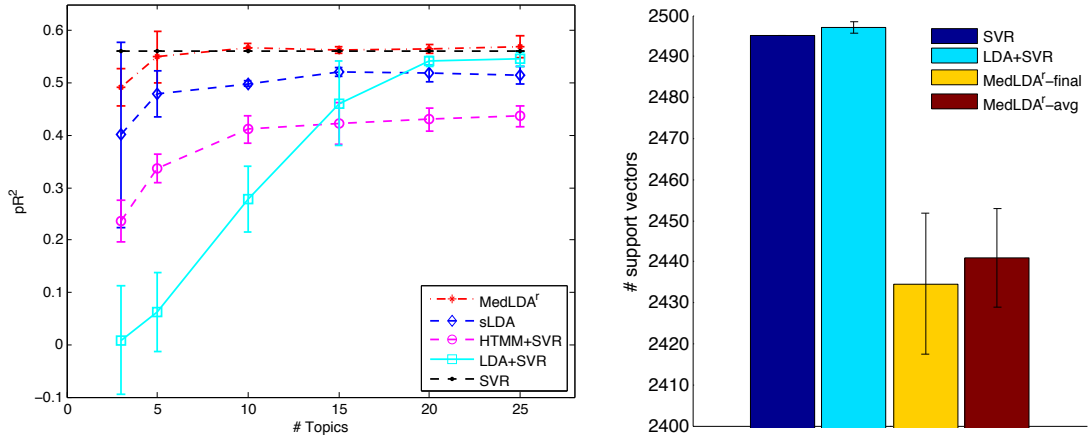
Figure 7: Predictive R^2 (left) and per-word likelihood (right) of different models on the movie review data set.

For the two variants of MedLDA regression models, we can see an obvious improvement of MedLDA^r over MedLDA^r_p. This is because for MedLDA^r_p, the update rule of ϕ does not have the third and fourth terms of Equation (14). Those terms make the max-margin estimation and latent topic discovery attached more tightly.

We also build another real data set of hotel review rating[18] by randomly crawling hotel reviews from TripAdvisor,[19] where each review is associated with a global rating score and five aspect rating scores for the aspects[20]—*Value*, *Rooms*, *Location*, *Cleanliness*, and *Service*. This data set is very interesting and can be used for many data mining tasks, for example, extracting the textual mentions of each aspect. Also, the rich features in reviews can be exploited to discover interesting latent structures with a conditional topic model (Zhu and Xing, 2010). In these experiments, we focus on predicting the global rating scores for reviews. To avoid too short and too long reviews, we only keep those reviews whose character length is between 1500 and 6000. On TripAdvisor, the global ratings rank from 1 to 5. We randomly select 1000 reviews for each rating and the data set consists of 5000 reviews in total. We uniformly partition it into training and testing sets. By removing a standard list of stopping words and those terms whose count frequency is less than 5, we build a dictionary with 12000 terms. Similarly, we take logarithm to make the response approximately normal. Figure 8(a) shows the predictive R^2 of different methods. Here, we also compare with the hidden topic Markov model (HTMM) (Gruber et al., 2007), which assumes the words in the same sentence have the same topic assignment. We use HTMM to discover latent representations of documents and use SVR to do regression. On this data set, we see a clear improvement of the supervised MedLDA^r compared to sLDA. The performance of unsupervised LDA (with a combination with SVR) is generally very unstable. The HTMM is more robust but its performance is worse than those of the supervised topic models. Finally, a linear SVR on empirical word frequency achieves a pR^2 of about 0.56, comparable to the best performance that can be achieved by MedLDA^r.

Figure 8(b) shows the number of support vectors for MedLDA^r, the standard SVR built on empirical word frequency, and the two-stage approach LDA+SVR. For MedLDA^r, we report both

18. The data set is available at `http://www.ml-thu.net/~jun/ReviewData.htm`.
19. TripAdvisor can be found at `http://www.tripadvisor.com/`.
20. The website is subject to change. Our data set was built in December, 2009.



(a) (b)

Figure 8: (a) Predictive R^2 of different models on the hotel review data set; and (b) the number of support vectors for SVR, LDA+SVR, and MedLDA^r. For MedLDA^r, we show both the number of support vectors at the final iteration and the average number during training.

the number of support vectors at the last iteration and the average number of support vectors during training. Here, we set $K = 10$ for LDA and MedLDA^r. Again, we can see that MedLDA^r needs fewer support vectors than SVR and LDA+SVR. In contrast, LDA+SVR needs about the same number of support vectors as SVR. This observation suggests that the topical representations by the supervised MedLDA^r are more suitable for learning a simple max-margin predictor, which is consistent with what we have observed in the classification case.

5.2.3 WHEN AND WHY SHOULD MEDLDA BE PREFERRED TO SVM? A DISCUSSION AND SIMULATION STUDY

The above results show that the MedLDA classification model works comparably or slightly better than the SVM classifiers built on raw input features; and for the two regression problems, MedLDA outperforms the support vector regression model (i.e., SVR) on one data set while they are comparable on the other data set. These results raise the question "when should we choose MedLDA?" Our answers are as follows.

First of all, MedLDA is a topic model. Besides making prediction on unseen data, one major function of MedLDA is that it can discover semantic patterns underlying complex data, and facilitate dimensionality reduction (and compression) of data. In contrast, SVM models are more like black box machines which take raw input features and find good decision boundaries or regression curves; but they are incapable of discovering or considering hidden structures of complex data, and performing dimensionality reduction.[21] Our main goal of including SVM/SVR into our comparison of predictive accuracy is indeed to demonstrate that dimensionality reduction and information extraction from raw data via MedLDA does not cause serious loss (if at all) predictive information, which is not the case for many alternative probabilistic or non-probabilistic information extractors

21. Some strategies like sparse feature selection can be incorporated to make an SVM more interpretable in the original feature space. But this is beyond the scope of this paper.

(e.g., LDA or LSI). As an integration of SVM with LDA, MedLDA performs both predictive and exploratory tasks simultaneously. So, the first selection rule is: *if we want to disclose some underlying patterns and extract a lower dimensional semantic-preserving representation of raw data besides doing prediction, MedLDA should be preferred to SVM*.

Second, even if our goal is focusing on prediction performance, MedLDA should also be considered as one competitive alternative. As shown in the above experiments, our simulation experiments below, as well as the follow-up works (Yang et al., 2010; Wang and Mori, 2011; Li et al., 2011), depending on the data and problems, max-margin supervised topic models can outperform SVM models, or they are comparable if no gains on predictive performance are obtained. There are several possible reasons for the comparable (not dramatically superior) classification performance we obtained on the 20 Newsgroups data:

(1) The fully factorized mean field inference method could potentially lead to inaccurate estimates. We have tried more sophisticated inference methods such as collapsed variational inference and collapsed Gibbs sampling,[22] both of which could lead to superior prediction performance (e.g., about 4 percent improvement over SVM on multi-class classification accuracy);

(2) The much lower dimensional topical representations could be too compact, compared to the original high-dimensional inputs. A clever combination (e.g., concatenation with appropriate re-scaling of different features) of the discovered latent topical representations and the original input features could potentially improve the performance, as demonstrated in Wang and Mori (2011) for image classification.

To further substantiate the claimed advantages of MedLDA over SVM for admixed (i.e., multi-topical) data such as text and image, we conduct some simulation experiments to empirically study when MedLDA can perform well. We generate the observed word counts from an LDA model with K topics. The Dirichlet parameters are $\boldsymbol{\alpha} = (1, \ldots, 1)$. For the topics, we randomly draw $\beta_{kn} \propto \text{Beta}(1,1)$, where $\propto$ means that we need to normalize $\boldsymbol{\beta}_k$ to be a distribution over the terms in a given vocabulary. We consider three different settings of binary classification with a vocabulary of 500 terms. The document lengths for each setting are randomly draw from a Poisson distribution, whose mean parameter is L, that is,

$$\forall d,\ N_d \sim \text{Poisson}(L).$$

(1) *Setting 1:* We set $K = 40$. We randomly draw the class label for document d from a distribution model

$$p(y_d = 1|\boldsymbol{\theta}_d) = \frac{1}{1+\exp\{-\boldsymbol{\eta}^\top \boldsymbol{\theta}_d\}},\ \text{where } \boldsymbol{\eta}_k \sim \mathcal{N}(0, 0.1).$$

In other words, the class labels are solely influenced by the latent topic representations. Therefore, the true model that generates the labeled data follows the assumptions of sLDA and MedLDA. We set $L = 25, 50, 150, 300, 500$.

(2) *Setting 2:* We set $K = 150$. We randomly draw the class label for document d from a distribution model

$$p(y_d = 1|\boldsymbol{\theta}_d) = \frac{1}{1+\exp\{-(\boldsymbol{\eta}_1^\top \boldsymbol{\theta}_d + \boldsymbol{\eta}_2^\top \mathbf{w}_d)\}},\ \text{where } \boldsymbol{\eta}_{ij} \sim \mathcal{N}(0, 0.1),\ i = 1, 2.$$

22. Sampling methods for MedLDA can be developed by using Lagrangian methods. But a full discussion on this topic is beyond the scope.

In other words, the true model that generates the labeled data does not follow the assumptions of sLDA. The class labels are influenced by the observed word counts. In fact, due to the law of conservation of belief (i.e., the total probability mass of a distribution must sum to one), the influence of $\boldsymbol{\theta}$ would be generally weaker than that of $\mathbf{w}$ in determining the true class labels. We set $L = 50, 100, 150, 200, 250$.

(3) *Setting 3:* Similar as in setting 2, but we improve the influence of $\boldsymbol{\theta}$ on class labels by using larger weights $\boldsymbol{\eta}_1$. Specifically, we sample the weights

$$\boldsymbol{\eta}_{1j} \sim K \times \mathcal{N}(0, 0.1) \text{ and } \boldsymbol{\eta}_{2j} \sim \mathcal{N}(0, 0.1).$$

We set $L = 50, 100, 150, 200, 250, 300, 350$.

In summary, the first two settings generally represent two extremes where the true model matches the assumptions of MedLDA or SVM, while Setting 3 is somewhat in the middle place between Setting 1 and Setting 2. Since the synthetic words do not have real meanings, below we focus on presenting the prediction performance, rather than visualizing the discovered topic representations.

Figure 9 shows the classification accuracy of MedLDA^c, the SVM classifiers built on word counts, and the MedLDA^c models using both $\boldsymbol{\theta}$ and word counts to learn classifiers[23] at each iteration step of solving for $q(\boldsymbol{\eta})$. We can see that for Setting 1, where the true model that generates the data matches the assumptions of MedLDA (and sLDA models too) well, we can achieve significant improvements compared to the SVM classifiers built on raw input word counts for all settings with various average document lengths. In contrast, for Setting 2, where the true model largely violates the assumptions of MedLDA (in fact, it matches the assumptions of SVM well), we generally do not have much improvements. But still, we can have comparable performance. For the middle ground in Setting 3, we have mixed results. When the average document length is small (e.g., ≤ 250), which means the influence of word counts on class labels is weak, MedLDA^c can improve a lot over SVM. But when the influence of word counts gets bigger (e.g., $L \geq 300$), using the low dimensional topic representations tends to be insufficient to get good performance. Translating to empirical text analysis, MedLDA will be particularly helpful when analyzing short texts, such as abstracts, reviews, users comments, and user status updates, which are nowadays the dominant forms of user texts on social media.

In all the three settings, we can see that a naïve combination of both latent topic representations and input word counts could improve the performance in some cases, or at least it will produce comparable performance with the better model between MedLDA^c and SVM. Finally, comparing the three settings, we can see that for Setting 2, since the true class labels heavily depend on the input word counts, increasing the average document length L generally improves the classification performance of all models. In other words, the classification problems become easier because of more discriminant information is provided as L increases. In contrast, we do not have the similar observations in the other two settings because the true labels are heavily (or solely in Setting 1) determined by $\boldsymbol{\theta}$, whose dimensionality is fixed.

The last reason that we think MedLDA should be considered as an important novel development with one root being from SVM because it presents one of the first successful attempts, in the particular context of Bayesian topic models, towards pushing forward the interface between max-margin learning and Bayesian generative modeling. As further demonstrated in others' work (Yang et al.,

23. We simply concatenate the two types of features without considering the scale difference.

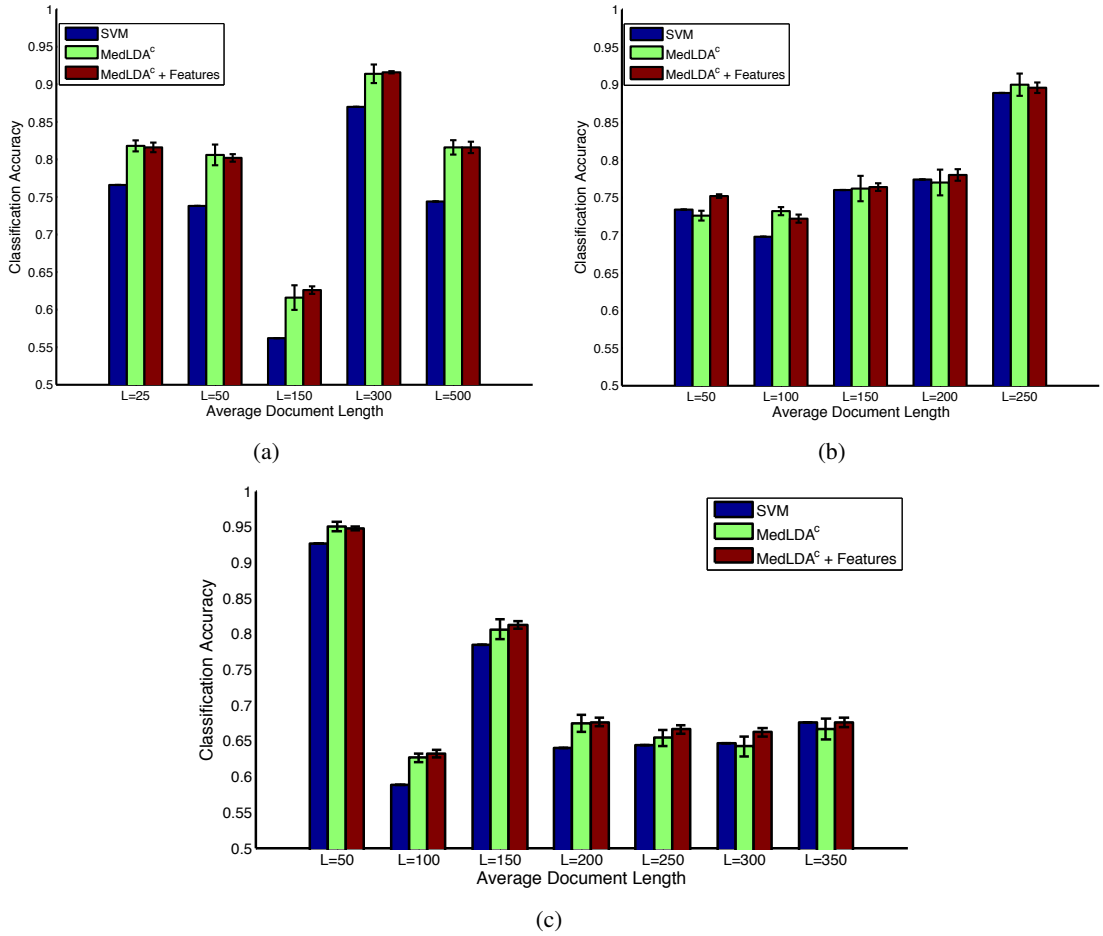


Figure 9: Classification accuracy of different methods in (a) Setting 1; (b) Setting 2; and (c) Setting 3.

2010; Wang and Mori, 2011; Li et al., 2011) as well as our recent work on regularized Bayesian inference (Zhu et al., 2011b,a), the max-margin principle can be a fruitful addition to "regularize" the desired posterior distributions of Bayesian models for performing better prediction in a broad range of scenarios, such as image annotation, classification, multi-task learning, etc.

5.3 Time Efficiency

In this section, we report empirical results on time efficiency in training and testing. All the following results are achieved on a standard desktop with a 2.66GHz Intel processor. We implement all the models in C++ language, without any special optimization of the code.

5.3.1 TRAINING TIME

Figure 10 shows the average training time of different models together with standard deviations on both binary and multi-class classification tasks with 5 randomly initialized runs. Here, we do not compare with DiscLDA because learning the transition matrix is not fully implemented by Lacoste-

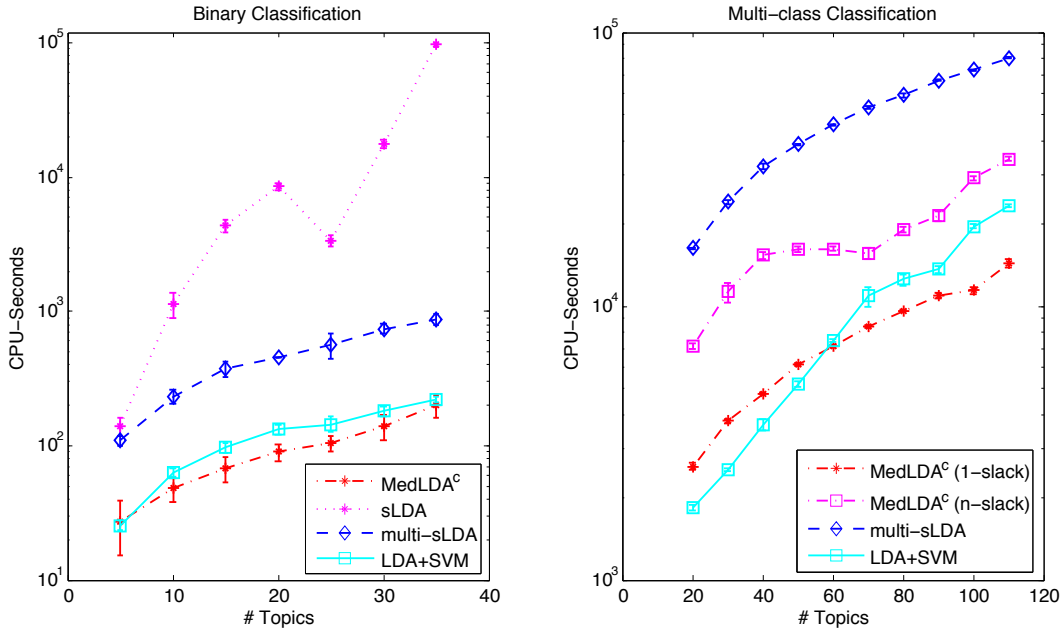
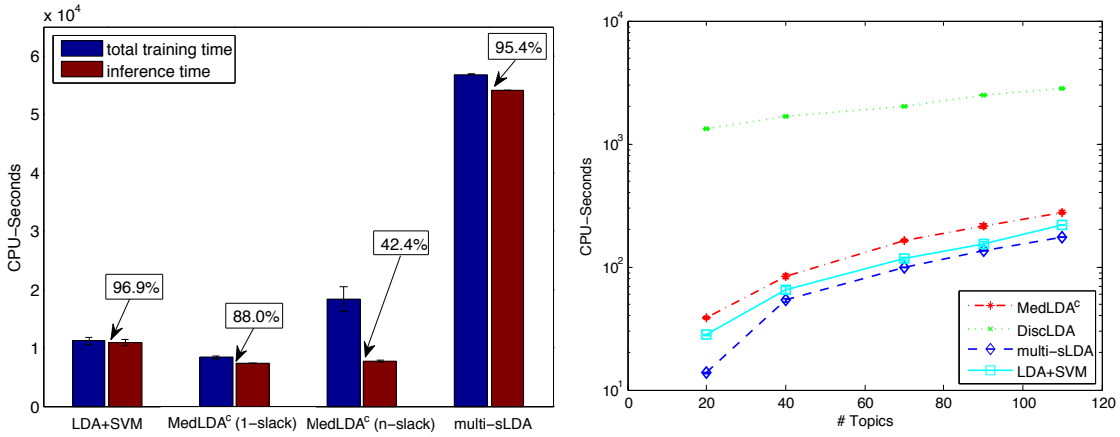


Figure 10: Training time (CPU seconds in log-scale) of different models with respect to the number of topics for both (Left) binary and (Right) multi-class classification.

Julien (2009), but we will compare the testing time with it. From the results, we can see that for binary classification, MedLDA^c is more efficient than multi-class sLDA and is comparable with LDA+SVM. The slowness of multi-class sLDA is because the normalization factor in the distribution model of y strongly couples the topic assignments of different words in the same document. Therefore, the posterior inference is slower than that of unsupervised LDA and MedLDA^c which uses unsupervised LDA as the underlying topic model. For the sLDA regression model, it takes even more training time because of the mismatch between its normal assumption and the non-Gaussian binary response variables, which prolongs the E-step. In contrast, MedLDA^c does not have such a normal assumption.

For multi-class classification, the training time of MedLDA^c is mainly dependent on solving a multi-class SVM problem. Here, we implemented both 1-slack and n-slack versions of multi-class SVM (Joachims et al., 2009) for solving the sub-problem of estimating $q(\boldsymbol{\eta})$ and Lagrangian multipliers in MedLDA^c. As we can see from Figure 10, the MedLDA^c with 1-slack SVM as the sub-solver can be very efficient, comparable to unsupervised LDA+SVM. The MedLDA^c with n-slack SVM solvers is about 3 times slower. Similar to the binary case, for the multi-class supervised sLDA (Wang et al., 2009), because of the normalization factor in the category probability model (i.e., a softmax function), the posterior inference on different topic assignment variables (in the same document) are strongly correlated. Therefore, the inference is (about 10 times) slower than that on unsupervised LDA and MedLDA^c which takes an unsupervised LDA as the underlying topic model. For regression, the training time of MedLDA^r is comparable to that of sLDA, while MedLDA^r_p is more efficient.

We also show the time spent on inference (i.e., E-step) and the ratio it takes over the total training time for different models in Figure 11(a). We can clearly see that the difference between 1-slack



(a) (b)

Figure 11: (a) The inference time (CPU seconds in linear scale) and total training time for learning different models, as well as the ratio of inference time over total training time. For MedLDA^c, we consider both the 1-slack and n-slack formulations; for LDA+SVM, the SVM classifier is by default the 1-slack formulation; and (b) Testing time (CPU seconds in log-scale) of different models with respect to the number of topics for multi-class classification.

MedLDA^c and n-slack MedLDA^c is on the learning of SVMs (i.e., M-step). Both methods have similar inference time. We can also see that for LDA+SVM and multi-sLDA, more than 95% of the training time is spent on inference, which is very expensive for multi-sLDA. Note that LDA+SVM takes a longer inference time than MedLDA^c. This is because we use more data (both training and testing) to learn unsupervised LDA. The SVM classifiers built on raw input word count features are generally much more faster than all the topic models. For instance, it takes about 230 seconds to train a 1-slack multi-class SVM on the 20 Newsgroups training data, or about 1000 seconds to train a n-slack multi-class SVM on the same training set; both are faster than the fastest topic model 1-slack MedLDA^c. This is reasonable because SVM classifiers do not spend time on inferring the latent topic representations.

5.3.2 TESTING TIME

Figure 11(b) shows the average testing time with standard deviation on 20 Newsgroup testing data with 5 randomly initialized runs. We can see that MedLDA^c, multi-class sLDA and unsupervised LDA are comparable in testing time, faster than that of DiscLDA. This is because all the three models of MedLDA^c, multi-class sLDA and LDA are *downstream* models (See the Introduction for definition). In testing, they do exactly the same tasks, that is, to infer the overall latent topical representation and do prediction with a linear model. Therefore, they have comparable testing time. However, DiscLDA is an *upstream* model, for which the prediction task is done with multiple times of doing inference to find the category-dependent latent topical representations. Therefore, in principle, the testing time of an upstream topic model is about $|\mathcal{C}|$ times slower than that of its downstream counterpart model, where $\mathcal{C}$ is the finite set of categories. The results in Figure 11(b)

show that DiscLDA is roughly about 20 times slower than other downstream models. Of course, the different inference algorithms can also make the testing time different.

6. Conclusions and Discussions

We have presented maximum entropy discrimination LDA (MedLDA), a supervised topic model that uses the discriminative max-margin principle to estimate model parameters such as topic distributions underlying a corpus, and infer latent topical vectors of documents. MedLDA integrates the max-margin principle into the process of topic learning and inference via optimizing one single objective function with a set of *expected* margin constraints. The objective function is a tradeoff between the goodness of fit of an underlying topic model and the prediction accuracy of the resultant topic vectors on a max-margin classifier. We provide empirical evidence as well as theoretical insights, which appear to demonstrate that this integration could yield predictive topical representations that are suitable for prediction tasks, such as regression and classification. We also present a general formulation of learning maximum entropy discrimination topic models, which allows any form of likelihood based topic models to be discriminatively trained. Although the general max-margin framework can be approximately solved with different methods, we concentrate on developing efficient variational methods for MedLDA in this paper. Our empirical results on movie review, hotel review and 20 Newsgroups data sets demonstrate that MedLDA is an attractive supervised topic model, which can achieve state of the art performance for topic discovery and prediction accuracy while needs fewer support vectors than competing max-margin methods that are built on raw text or the topical representations discovered by unsupervised LDA.

MedLDA represents the first step towards integrating the max-margin principle into supervised topic models, and under the general MedTM framework presented in Section 4, several improvements and extensions are in the horizon. Specifically, due to the nature of MedTM's joint optimization formulation, advances in either max-margin training or better variational bounds for inference can be easily incorporated. For instance, the mean field variational upper bound in MedLDA can be improved by using the tighter collapsed variational bound (Teh et al., 2006) that achieves results comparable to collapsed Gibbs sampling (Griffiths and Steyvers, 2004). Moreover, as the experimental results suggest, incorporation of a more expressive underlying topic model enhances the overall performance. Therefore, we plan to integrate and use other underlying topic models like the fully generative sLDA model in the classification case. However, as we have stated, the challenge in developing fully supervised MedLDA classification model lies in the hard posterior inference caused by the normalization factor in the category distribution model. Finally, advance in max-margin training would also results in more efficient training.

Acknowledgments

We thank David Blei for answering questions about implementing sLDA, Chong Wang for sharing his implementation of multi-class sLDA, Simon Lacoste-Julien for discussions on DiscLDA and feedbacks on our implementation of MedLDA, and the anonymous reviewers for valuable comments. This work was done while J.Z. was visiting CMU under a support from NSF DBI-0546594 and DBI-0640543 awarded to E.X.; J.Z. is supported by National Key Foundation R&D Projects 2012CB316301, Basic Research Foundation of Tsinghua National Laboratory for Infor-

mation Science and Technology (TNList), a Starting Research Fund from Tsinghua University, No. 553420003, and the 221 Basic Research Plan for Young Faculties at Tsinghua University.

Appendix A. Proof of Corollary 4

In this section, we prove the corollary 4.

Proof Since the variational parameters $(\boldsymbol{\gamma}, \boldsymbol{\phi})$ are fixed when solving for $q(\boldsymbol{\eta})$, we can ignore the terms in $\mathcal{L}^{bs}$ that do not depend on $q(\boldsymbol{\eta})$ and get the function

$$\begin{aligned}\mathcal{L}^{bs}_{[q(\boldsymbol{\eta})]} &\triangleq KL(q(\boldsymbol{\eta})\|p_0(\boldsymbol{\eta})) - \sum_d \mathbb{E}_q[\log p(y_d|\bar{Z}_d, \boldsymbol{\eta}, \delta^2)] \\ &= KL(q(\boldsymbol{\eta})\|p_0(\boldsymbol{\eta})) + \frac{1}{2\delta^2}\left(\mathbb{E}_{q(\boldsymbol{\eta})}[\boldsymbol{\eta}^\top \mathbb{E}[AA^\top]\eta - 2\boldsymbol{\eta}^\top \sum_{d=1}^{D} y_d \mathbb{E}[\bar{Z}_d]]\right) + c,\end{aligned}$$

where c is a constant that does not depend on $q(\boldsymbol{\eta})$.

Let $U(\boldsymbol{\xi}, \boldsymbol{\xi}^*) = C\sum_{d=1}^{D}(\xi_d + \xi_d^*)$. Suppose $(q_0(\boldsymbol{\eta}), \boldsymbol{\xi}_0, \boldsymbol{\xi}_0^*)$ is the optimal solution of P1, then we have: for any feasible $(q(\boldsymbol{\eta}), \boldsymbol{\xi}, \boldsymbol{\xi}^*)$,

$$\mathcal{L}^{bs}_{[q_0(\boldsymbol{\eta})]} + U(\boldsymbol{\xi}_0, \boldsymbol{\xi}_0^*) \leq \mathcal{L}^{bs}_{[q(\boldsymbol{\eta})]} + U(\boldsymbol{\xi}, \boldsymbol{\xi}^*).$$

From Corollary 3, we conclude that the optimum predictive parameter distribution is $q_0(\boldsymbol{\eta}) = \mathcal{N}(\boldsymbol{\lambda}_0, \Sigma)$, where $\Sigma = (I + 1/\delta^2 \mathbb{E}[A^\top A])^{-1}$ does not depend on $q(\boldsymbol{\eta})$. Since $q_0(\boldsymbol{\eta})$ is also normal, for any distribution[24] $q(\boldsymbol{\eta}) = \mathcal{N}(\boldsymbol{\lambda}, \Sigma)$, with several steps of algebra it is easy to show that

$$\mathcal{L}^{bs}_{[q(\boldsymbol{\eta})]} = \frac{1}{2}\boldsymbol{\lambda}^\top (I + \frac{1}{\delta^2}\mathbb{E}[A^\top A])\boldsymbol{\lambda} - \boldsymbol{\lambda}^\top (\sum_{d=1}^{D} \frac{y_d}{\delta^2}\mathbb{E}[\bar{Z}_d]) + c' = \frac{1}{2}\boldsymbol{\lambda}^\top \Sigma^{-1}\boldsymbol{\lambda} - \boldsymbol{\lambda}^\top (\sum_{d=1}^{D} \frac{y_d}{\delta^2}\mathbb{E}[\bar{Z}_d]) + c',$$

where c' is another constant that does not depend on $\boldsymbol{\lambda}$.

Thus, we can get: for any $(\boldsymbol{\lambda}, \boldsymbol{\xi}, \boldsymbol{\xi}^*)$, where

$$(\boldsymbol{\lambda}, \boldsymbol{\xi}, \boldsymbol{\xi}^*) \in \{(\boldsymbol{\lambda}, \boldsymbol{\xi}, \boldsymbol{\xi}^*) : \ y_d - \boldsymbol{\lambda}^\top \mathbb{E}[\bar{Z}_d] \leq \varepsilon + \xi_d; \ -y_d + \boldsymbol{\lambda}^\top \mathbb{E}[\bar{Z}_d] \leq \varepsilon + \xi_d^*; \text{ and } \boldsymbol{\xi}, \boldsymbol{\xi}^* \geq 0 \ \forall d\},$$

we have

$$\frac{1}{2}\boldsymbol{\lambda}_0^\top \Sigma^{-1}\boldsymbol{\lambda}_0 - \boldsymbol{\lambda}_0^\top (\sum_{d=1}^{D} \frac{y_d}{\delta^2}\mathbb{E}[\bar{Z}_d]) + U(\boldsymbol{\xi}_0, \boldsymbol{\xi}_0^*) \leq \frac{1}{2}\boldsymbol{\lambda}^\top \Sigma^{-1}\boldsymbol{\lambda} - \boldsymbol{\lambda}^\top (\sum_{d=1}^{D} \frac{y_d}{\delta^2}\mathbb{E}[\bar{Z}_d]) + U(\boldsymbol{\xi}, \boldsymbol{\xi}^*),$$

which means the mean of the optimum posterior distribution under a Gaussian MedLDA is achieved by solving a primal problem as stated in the Corollary.

■

24. Although the feasible set of $q(\boldsymbol{\eta})$ in P1 is much richer than the set of normal distributions with the covariance matrix Σ, Corollary 3 shows that the solution is a restricted normal distribution. Thus, it suffices to consider only these normal distributions in order to learn the mean of the optimum distribution.

Appendix B. Max-Margin Learning of the Vanilla LDA for Regression

In Section 3.1, we have presented the MedLDA regression model that uses supervised sLDA (Blei and McAuliffe, 2007) to discover latent topic assignments Z and document-level topical representations $\boldsymbol{\theta}$. The same principle can be applied to perform joint maximum likelihood estimation and max-margin training for unsupervised LDA (Blei et al., 2003), which does not directly model side information such as user ratings y. In this section, we present this MedLDA model, which will be referred to as *MedLDA*^r_p. As in MedLDA^c, we assume that the supervised side information y is given, even though not included in the joint likelihood function defined in LDA.[25]

A naïve approach to using unsupervised LDA for supervised prediction tasks (e.g., regression) is a two-stage procedure: 1) using unsupervised LDA to discover the latent topical representations of documents; and 2) feeding the low-dimensional topical representations into a regression model (e.g., SVR) for training and testing. This de-coupled approach can be rather sub-optimal because the side information of documents (e.g., rating scores of movie reviews) is not used in discovering the low-dimensional representations and thus can result in a sub-optimal representation for prediction tasks. Below, we present MedLDA^r_p, which integrates an unsupervised LDA for discovering topics with the SVR for regression. The inter-play between topic discovery and supervised prediction will result in more discriminative latent topical representations, similar as in MedLDA^r.

When the underlying topic model is unsupervised LDA, the likelihood is $p(\mathbf{W}|\boldsymbol{\alpha},\boldsymbol{\beta})$, the same as in MedLDA^c. For regression, we apply the ε-insensitive support vector regression (SVR) (Smola and Schölkopf, 2003) approach as before. Again, we learn a distribution $q(\boldsymbol{\eta})$. The prediction rule is the same as in Equation (6). The integrated learning problem is

$$\text{P6(MedLDA}^r_p\text{)}: \quad \min_{q,q(\boldsymbol{\eta}),\boldsymbol{\alpha},\boldsymbol{\beta},\boldsymbol{\xi},\boldsymbol{\xi}^*} \quad \mathcal{L}^u(q;\boldsymbol{\alpha},\boldsymbol{\beta}) + KL(q(\boldsymbol{\eta})||p_0(\boldsymbol{\eta})) + C\sum_{d=1}^{D}(\xi_d+\xi^*_d)$$
$$\forall d,\ \text{s.t.}: \quad \begin{cases} y_d - \mathbb{E}[\boldsymbol{\eta}^\top \bar{Z}_d] \le \varepsilon + \xi_d \\ -y_d + \mathbb{E}[\boldsymbol{\eta}^\top \bar{Z}_d] \le \varepsilon + \xi^*_d \\ \xi_d, \xi^*_d \ge 0 \end{cases},$$

where the KL-divergence is a regularizer that biases the estimate of $q(\boldsymbol{\eta})$ towards the prior. In MedLDA^r, this KL-regularizer is implicitly contained in the variational bound $\mathcal{L}^{bs}$ as shown in Equation (7). The constrained problem is equivalent to the "unconstrained" problem by removing slack variables:

$$\min_{q,q(\boldsymbol{\eta}),\boldsymbol{\alpha},\boldsymbol{\beta}} \mathcal{L}^u(q;\boldsymbol{\alpha},\boldsymbol{\beta}) + KL(q(\boldsymbol{\eta})||p_0(\boldsymbol{\eta})) + C\sum_{d=1}^{D}\max(0, |y_d - \mathbb{E}[\boldsymbol{\eta}^\top \bar{Z}_d]| - \varepsilon) \tag{22}$$

Variational Algorithm: For MedLDA^r_p, the unconstrained optimization problem (22) can be similarly solved with a coordinate-descent algorithm as in the case of MedLDA^r. Specifically, we assume that $q(\{\boldsymbol{\theta}_d, \mathbf{z}_d\}) = \prod_{d=1}^{D} q(\boldsymbol{\theta}_d|\boldsymbol{\gamma}_d)\prod_{n=1}^{N} q(z_{dn}|\boldsymbol{\phi}_{dn})$, where the variational parameters $\boldsymbol{\gamma}$ and $\boldsymbol{\phi}$ have the same meanings as in MedLDA^r. Then, we alternately solve for each variable and get a variational algorithm which is similar to that of MedLDA^r.

25. One could argue that this design is unreasonable because with y one should only consider sLDA. But we study fitting the vanilla LDA using y in an indirect way described below because of the popularity and historical importance of this scheme in many applied domains.

Solve for $(\boldsymbol{\alpha}, \boldsymbol{\beta})$ *and* $q(\boldsymbol{\eta})$: the update rules of $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ are the same as in the MedLDA^r. The parameter δ^2 is not used here. By using Lagrangian methods, we get that

$$q(\boldsymbol{\eta}) = \frac{p_0(\boldsymbol{\eta})}{Z} \exp\left(\boldsymbol{\eta}^\top \sum_{d=1}^{D} (\hat{\mu}_d - \hat{\mu}_d^*)\mathbb{E}[\bar{Z}_d]\right)$$

and the dual problem is the same as D2. Again, we can choose different priors to introduce some regularization effects. For the standard normal prior: $p_0(\boldsymbol{\eta}) = \mathcal{N}(0, I)$, the posterior is also a normal: $q(\boldsymbol{\eta}) = \mathcal{N}(\boldsymbol{\lambda}, I)$, where $\boldsymbol{\lambda} = \sum_{d=1}^{D}(\hat{\mu}_d - \hat{\mu}_d^*)\mathbb{E}[\bar{Z}_d]$ is the mean. This identity covariance matrix is much simpler than the covariance matrix Σ as in *MedLDA*^{r}, which depends on the latent topical representation Z. Since I is independent of Z, the prediction model in *MedLDA*^r_p is less affected by the latent topical representations. Together with the simpler update rule (23), we can conclude that the coupling between the max-margin estimation and the discovery of latent topical representations in *MedLDA*^r_p is looser than that of *MedLDA*^{r}. The looser coupling will lead to inferior empirical performance as we show in Section 5.2.

For the standard normal prior, the dual problem is a QP problem:

$$\begin{aligned} \max_{\boldsymbol{\mu}, \boldsymbol{\mu}^*} \quad & -\frac{1}{2}\|\boldsymbol{\lambda}\|_2^2 - \varepsilon \sum_{d=1}^{D}(\mu_d + \mu_d^*) + \sum_{d=1}^{D} y_d(\mu_d - \mu_d^*) \\ \forall d, \text{ s.t. : } \quad & \mu_d, \mu_d^* \in [0, C], \end{aligned}$$

Similarly, we can derive its primal form, which is as a standard SVR problem:

$$\begin{aligned} \min_{\boldsymbol{\lambda}, \xi, \xi^*} \quad & \frac{1}{2}\|\boldsymbol{\lambda}\|_2^2 + C\sum_{d=1}^{D}(\xi_d + \xi_d^*) \\ \text{s.t. } \forall d : \quad & \begin{cases} y_d - \boldsymbol{\lambda}^\top \mathbb{E}[\bar{Z}_d] \le \varepsilon + \xi_d \\ -y_d + \boldsymbol{\lambda}^\top \mathbb{E}[\bar{Z}_d] \le \varepsilon + \xi_d^* \\ \xi_d, \xi_d^* \ge 0. \end{cases} \end{aligned}$$

Now, we can leverage recent developments in support vector regression (e.g., the public SVM-light package) to solve either the dual problem or the primal problem.

Solve for $q(\{\boldsymbol{\theta}_d, \mathbf{z}_d\})$: We have the same update rule for $\boldsymbol{\gamma}$ as in MedLDA^r. By using the similar one-step approximation strategy, we have:

$$\boldsymbol{\phi}_{dn} \propto \exp\left(\mathbb{E}[\log \boldsymbol{\theta}_d | \boldsymbol{\gamma}_d] + \log p(w_{dn} | \boldsymbol{\beta}) + \frac{\mathbb{E}[\boldsymbol{\eta}]}{N}(\hat{\mu}_d - \hat{\mu}_d^*)\right), \tag{23}$$

Again, we can see that how the max-margin constraints in P6 regularize the procedure of discovering latent topical representations through the last term in Equation (23). Specifically, for a document d, which lies around the decision boundary, that is, a support vector, either $\hat{\mu}_d$ or $\hat{\mu}_d^*$ is non-zero, and the last term biases $\boldsymbol{\phi}_{dn}$ towards a distribution that favors a more accurate prediction on the document. However, compared to Equation (14), we can see that Equation (23) is simpler and does not have the complex third and fourth terms of Equation (14). This simplicity suggests that the latent topical representation is less affected by the max-margin estimation (i.e., the prediction model's parameters).

References

Edoardo M. Airoldi, David M. Blei, Stephen E. Fienberg, and Eric P. Xing. Mixed membership stochastic blockmodels. *Journal of Machine Learning Research*, (9):1981–2014, 2008.

David Blei and John Lafferty. Correlated topic models. In Y. Weiss, B. Schölkopf, and J. Platt, editors, *Advances in Neural Information Processing Systems (NIPS)*, pages 147–154, Cambridge, MA, 2005. MIT Press.

David Blei and Jon D. McAuliffe. Supervised topic models. In J.C. Platt, D. Koller, Y. Singer, and S. Roweis, editors, *Advances in Neural Information Processing Systems (NIPS)*, pages 121–128, Cambridge, MA, 2007. MIT Press.

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

Gal Chechik and Naftali Tishby. Extracting relevant structures with side information. In S. Thrun S. Becker and K. Obermayer, editors, *Advances in Neural Information Processing Systems (NIPS)*, pages 857–864, Cambridge, MA, 2002. MIT Press.

Ning Chen, Jun Zhu, and Eric P. Xing. Predictive subspace learning for multi-view data: a large margin approach. In J. Lafferty, C. K. I. Williams, J. Shawe-Taylor, R.S. Zemel, and A. Culotta, editors, *Advances in Neural Information Processing Systems (NIPS)*, pages 361–369, 2010.

Koby Crammer and Yoram Singer. On the algorithmic implementation of multiclass kernel-based vector machines. *Journal of Machine Learning Research*, (2):265–292, 2001.

Paul Damien and Stephen G. Walker. Sampling truncated Normal, Beta, and Gamma densities. *Journal of Computational and Graphical Statistics*, 10(2):206–215, 2001.

Li Fei-Fei and Pietro Perona. A Bayesian hierarchical model for learning natural scene categories. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 524–531, San Diego, CA, 2005.

Pedro Felzenszwalb, Ross Girshick, David McAllester, and Deva Ramanan. Object detection with discriminatively trained part based models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(9):1627 – 1645, 2010.

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

William E. Griffiths. A Gibbs sampler for the parameters of a truncated multivariate normal distribution. *No 856, Department of Economics, University of Melbourne*, 2002.

Amit Gruber, Michal Rosen-Zvi, and Yair Weiss. Hidden topic Markov models. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, pages 163–170, San Juan, Puerto Rico, 2007.

Xuming He and Richard S. Zemel. Learning hybrid models for image annotation with partially labeled data. In D. Koller, D. Schuurmans, Y. Bengio, and L. Bottou, editors, *Advances in Neural Information Processing Systems (NIPS)*, pages 625–632, 2008.

Tommi Jaakkola, Marina Meila, and Tony Jebara. Maximum entropy discrimination. In *Advances in Neural Information Processing Systems (NIPS)*, pages 470–476, Denver, Colorado, 1999.

Tony Jebara. *Discriminative, Generative and Imitative Learning*. PhD thesis, Media Laboratory, MIT, Dec 2001.

Thorsten Joachims. Making large-scale SVM learning practical. *Advances in Kernel Methods–Support Vector Learning, MIT-Press*, 1999.

Thorsten Joachims, Thomas Finley, and Chun-Nam Yu. Cutting-plane training of structural SVMs. *Machine Learning Journal*, 77(1):27–59, 2009.

Michael I. Jordan, Zoubin Ghahramani, Tommis Jaakkola, and Lawrence K. Saul. *An Introduction to Variational Methods for Graphical Models*. M. I. Jordan (Ed.), Learning in Graphical Models, Cambridge: MIT Press, Cambridge, MA, 1999.

Simon Lacoste-Julien. *Discriminative Machine Learning with Structure*. PhD thesis, EECS Department, University of California, Berkeley, Jan 2009. URL `http://www.eecs.berkeley.edu/Pubs/TechRpts/2010/EECS-2010-4.html`.

Simon Lacoste-Julien, Fei Sha, and Michael I. Jordan. DiscLDA: Discriminative learning for dimensionality reduction and classification. In D. Koller, D. Schuurmans, Y. Bengio, and L. Bottou, editors, *Advances in Neural Information Processing Systems (NIPS)*, pages 897–904, 2008.

Dingcheng Li, Swapna Somasundaran, and Amit Chakraborty. A combination of topic models with max-margin learning for relation detection. In *ACL TextGraphs-6 Workshop*, 2011.

Li-Jia Li, Richard Socher, and Fei-Fei Li. Towards total scene understanding: Classification, annotation and segmentation in an automatic framework. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2036–2043, Miami, Florida, 2009.

David Mimno and Andrew McCallum. Topic models conditioned on arbitrary features with Dirichlet-multinomial regression. In *International Conference on Uncertainty in Artificial Intelligence (UAI)*, pages 411–418, Corvallis, Oregon, 2008.

Bo Pang and Lillian Lee. Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales. In *Proceddings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 115–124, Ann Arbor, Michigan, 2005.

Dmitry Pavlov, Alexandrin Popescul, David M. Pennock, and Lyle H. Ungar. Mixtures of conditional maximum entropy models. In *International Conference on Machine Learning (ICML)*, Washington, DC USA, 2003.

Daniel Ramage, David Hall, Ramesh Nallapati, and Christopher D. Manning. Labeled LDA: A supervised topic model for credit attribution in multi-labeled corpora. In *Proceddings of the 2009 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 248–256, Singapore, 2009.

Gabriel Rodriguez-Yam, Richard Davis, and Louis Scharf. Efficient Gibbs sampling of truncated multivariate normal with application to constrained linear regression. *Technical Report, Department of Statistics, Columbia University*, 2004.

Bryan C. Russell, Antonio Torralba, Kevin P. Murphy, and William T. Freeman. Labelme: a database and web-based tool for image annotation. *International Journal of Computer Vision*, 77(1-3):157–173, 2008.

Ruslan Salakhutdinov and Geoffrey Hinton. Replicated softmax: an undirected topic model. In Y. Bengio, D. Schuurmans, J. Lafferty, C. K. I. Williams, and A. Culotta, editors, *Advances in Neural Information Processing Systems (NIPS)*, pages 1607–1614, Vancouver, B.C., Canada, 2009.

Edward Schofield. *Fitting Maximum-Entropy Models on Large Sample Spaces*. PhD thesis, Department of Computing, Imperial College London, Jan 2007.

Alex J. Smola and Bernhard Schölkopf. A tutorial on support vector regression. *Statistics and Computing*, 14(3):199–222, 2003.

Rion Snow, Brendan O'Connor, Daniel Jurafsky, and Andrew Ng. Cheap and fast – but is it good? evaluating non-expert annotations for natural language tasks. In *Proceddings of the 2008 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 254–263, Honolulu, Hawaii, 2008.

Erik Sudderth, Antonio Torralba, William Freeman, and Alan Willsky. Learning hierarchical models of scenes, objects, and parts. In *IEEE International Conference on Computer Vision (ICCV)*, pages 1331–1338, Beijing, China, 2005.

Ben Taskar, Carlos Guestrin, and Daphne Koller. Max-margin Markov networks. In Sebastian Thrun, Lawrence Saul, and Bernhard Schölkopf, editors, *Advances in Neural Information Processing Systems (NIPS)*, Cambridge, MA, 2003. MIT Press.

Yee Whye Teh, David Newman, and Max Welling. A collapsed variational Bayesian inference algorithm for latent Dirichlet allocation. In B. Schölkopf, J. Platt, and T. Hoffman, editors, *Advances in Neural Information Processing Systems (NIPS)*, pages 1353–1360, Cambridge, MA, 2006. MIT Press.

Ivan Titov and Ryan McDonald. A joint model of text and aspect ratings for sentiment summarization. In *Proceddings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 308–316, Columbus, Ohio, 2008.

Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-SNE. *Journal of Machine Learning Research*, (9):2579–2605, 2008.

Vladimir Vapnik. *Statistical Learning Theory*. John Wiley and Sons, New York, 1998.

Chong Wang, David Blei, and Li Fei-Fei. Simultaneous image classification and annotation. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1903–1910, Miami, Florida, 2009.

Yang Wang and G. Mori. Max-margin latent Dirichlet allocation for image classification and annotation. In *British Machine Vision Conference (BMVC)*, 2011.

Max Welling, Michal Rosen-Zvi, and Geoffrey Hinton. Exponential family harmoniums with an application to information retrieval. In Lawrence K. Saul, Yair Weiss, and Léon Bottou, editors, *Advances in Neural Information Processing Systems (NIPS)*, pages 1481–1488, Cambridge, MA, 2004. MIT Press.

Eric P. Xing, Rong Yan, and Alexander G. Hauptmann. Mining associated text and images with dual-wing Harmoniums. In *International Conference on Uncertainty in Artifical Intelligence (UAI)*, pages 633–641, Arlington, Virginia, 2005.

Shuanghong Yang, Jiang Bian, and Hongyuan Zha. Hybrid generative/discriminative learning for automatic image annotation. In *International Conference on Uncertainty in Artifical Intelligence (UAI)*, pages 683–690, Corvallis, Oregon, 2010.

Chun-Nam Yu and Thorsten Joachims. Learning structural SVMs with latent variables. In Léon Bottou and Michael Littman, editors, *International Conference on Machine Learning (ICML)*, pages 1169–1176, Montreal, 2009.

Bing Zhao and Eric P. Xing. HM-BiTAM: Bilingual topic exploration, word alignment, and translation. In J.C. Platt, D. Koller, Y. Singer, and S. Roweis, editors, *Advances in Neural Information Processing Systems (NIPS)*, pages 1689–1696, Cambridge, MA, 2007. MIT Press.

Jun Zhu and Eric P. Xing. Maximum entropy discrimination Markov networks. *Journal of Machine Learning Research*, (10):2531–2569, 2009.

Jun Zhu and Eric P. Xing. Conditional topic random fields. In J. Fürnkranz and T. Joachims, editors, *International Conference on Machine Learning (ICML)*, pages 1239–1246, Haifa, Israel, 2010.

Jun Zhu, Eric P. Xing, and Bo Zhang. Partially observed maximum entropy discrimination Markov networks. In D. Koller, D. Schuurmans, Y. Bengio, and L. Bottou, editors, *Advances in Neural Information Processing Systems (NIPS)*, pages 1977–1984, 2008.

Jun Zhu, Amr Ahmed, and Eric P. Xing. MedLDA: Maximum margin supervised topic models for regression and classification. In Léon Bottou and Michael Littman, editors, *International Conference on Machine Learning (ICML)*, pages 1257–1264, Montreal, 2009.

Jun Zhu, Li-Jia Li, Li Fei-Fei, and Eric P. Xing. Large margin training of upstream scene understanding models. In J. Lafferty, C. K. I. Williams, J. Shawe-Taylor, R.S. Zemel, and A. Culotta, editors, *Advances in Neural Information Processing Systems (NIPS)*, pages 2586–2594, 2010.

Jun Zhu, Ning Chen, and Eric P. Xing. Infinite SVM: a Dirichlet process mixture of large-margin kernel machines. In L. Getoor and T. Scheffer, editors, *International Conference on Machine Learning (ICML)*, pages 617–624, Bellevue, Washington, USA, 2011a.

Jun Zhu, Ning Chen, and Eric P. Xing. Infinite latent SVM for classification and multi-task learning. In J. Shawe-Taylor, R.S. Zemel, P. Bartlett, F.C.N. Pereira, and K.Q. Weinberger, editors, *Advances in Neural Information Processing Systems (NIPS)*, pages 1620–1628, 2011b.

Journal of Machine Learning Research 13 (2012) 2279-2292 Submitted 8/11; Revised 3/12; Published 8/12

Pairwise Support Vector Machines and their Application to Large Scale Problems

Carl Brunner C.BRUNNER@GMX.NET
Andreas Fischer ANDREAS.FISCHER@TU-DRESDEN.DE
Institute for Numerical Mathematics
Technische Universität Dresden
01062 Dresden, Germany

Klaus Luig LUIG@COGNITEC.COM
Thorsten Thies THIES@COGNITEC.COM
Cognitec Systems GmbH
Grossenhainer Str. 101
01127 Dresden, Germany

Editor: Corinna Cortes

Abstract

Pairwise classification is the task to predict whether the examples a, b of a pair (a, b) belong to the same class or to different classes. In particular, interclass generalization problems can be treated in this way. In pairwise classification, the order of the two input examples should not affect the classification result. To achieve this, particular kernels as well as the use of symmetric training sets in the framework of support vector machines were suggested. The paper discusses both approaches in a general way and establishes a strong connection between them. In addition, an efficient implementation is discussed which allows the training of several millions of pairs. The value of these contributions is confirmed by excellent results on the labeled faces in the wild benchmark.

Keywords: pairwise support vector machines, interclass generalization, pairwise kernels, large scale problems

1. Introduction

To extend binary classifiers to multiclass classification several modifications have been suggested, for example the one against all technique, the one against one technique, or directed acyclic graphs, see Duan and Keerthi (2005), Hill and Doucet (2007), Hsu and Lin (2002), and Rifkin and Klautau (2004) for further information, discussions, and comparisons. A more recent approach used in the field of multiclass and binary classification is pairwise classification (Abernethy et al., 2009; Bar-Hillel et al., 2004a,b; Bar-Hillel and Weinshall, 2007; Ben-Hur and Noble, 2005; Phillips, 1999; Vert et al., 2007). Pairwise classification relies on two input examples instead of one and predicts whether the two input examples belong to the same class or to different classes. This is of particular advantage if only a subset of classes is known for training. For later use, a support vector machine (SVM) that is able to handle pairwise classification tasks is called pairwise SVM.

A natural requirement for a pairwise classifier is that the order of the two input examples should not influence the classification result (symmetry). A common approach to enforce this symmetry is the use of selected kernels. For pairwise SVMs, another approach was suggested. Bar-Hillel

et al. (2004a) propose the use of training sets with a symmetric structure. We will discuss both approaches to obtain symmetry in a general way. Based on this, we will provide conditions when these approaches lead to the same classifier. Moreover, we show empirically that the approach of using selected kernels is three to four times faster in training.

A typical pairwise classification task arises in face recognition. There, one is often interested in the interclass generalization, where none of the persons in the training set is part of the test set. We will demonstrate that training sets with many classes (persons) are needed to obtain a good performance in the interclass generalization. The training on such sets is computationally expensive. Therefore, we discuss an efficient implementation of pairwise SVMs. This enables the training of pairwise SVMs with several millions of pairs. In this way, for the labeled faces in the wild database, a performance is achieved which is superior to the current state of the art.

This paper is structured as follows. In Section 2 we give a short introduction to pairwise classification and discuss the symmetry of decision functions obtained by pairwise SVMs. Afterwards, in Section 3.1, we analyze the symmetry of decision functions from pairwise SVMs that rely on symmetric training sets. The new connection between the two approaches for obtaining symmetry is established in Section 3.2. The efficient implementation of pairwise SVMs is discussed in Section 4. Finally, we provide performance measurements in Section 5.

The main contribution of the paper is that we show the equivalence of two approaches for obtaining a symmetric classifier from pairwise SVMs and demonstrate the efficiency and good interclass generalization performance of pairwise SVMs on large scale problems.

2. Pairwise Classification

Let X be an arbitrary set and let m training examples $x_i \in X$ with $i \in M := \{1,\dots,m\}$ be given. The class of a training example might be unknown, but we demand that we know for each pair (x_i,x_j) of training examples whether its examples belong to the same class or to different classes. Accordingly, we define $y_{ij} := +1$ if the examples of the pair (x_i,x_j) belong to the same class and call it a *positive pair*. Otherwise, we set $y_{ij} := -1$ and call (x_i,x_j) a *negative pair*.

In pairwise classification the aim is to decide whether the examples of a pair $(a,b) \in X \times X$ belong to the same class or not. In this paper, we will make use of pairwise decision functions $f : X \times X \to \mathbb{R}$. Such a function predicts whether the examples a,b of a pair (a,b) belong to the same class ($f(a,b) > 0$) or not ($f(a,b) < 0$). Note that neither a,b need to belong to the set of training examples nor the classes of a,b need to belong to the classes of the training examples.

A common tool in machine learning are kernels $k : X \times X \to \mathbb{R}$. Let $\mathcal{H}$ denote an arbitrary real Hilbert space with scalar product $\langle\cdot,\cdot\rangle$. For $\phi : X \to \mathcal{H}$,

$$k(s,t) := \langle \phi(s), \phi(t) \rangle$$

defines a *standard* kernel.

In pairwise classification one often uses *pairwise* kernels $K : (X \times X) \times (X \times X) \to \mathbb{R}$. In this paper we assume that any pairwise kernel is symmetric, that is, it holds that

$$K((a,b),(c,d)) = K((c,d),(a,b))$$

for all $a,b,c,d \in X$, and that it is positive semidefinite (Schölkopf and Smola, 2001). For instance,

$$K_D((a,b),(c,d)) := k(a,c) + k(b,d), \tag{1}$$

$$K_T((a,b),(c,d)) := k(a,c) \cdot k(b,d) \tag{2}$$

are symmetric and positive semidefinite. We call K_D *direct sum pairwise kernel* and K_T *tensor pairwise kernel* (cf. Schölkopf and Smola, 2001).

A natural and desirable property of any pairwise decision function is that it should be symmetric in the following sense

$$f(a,b) = f(b,a) \quad \text{for all } a,b \in X.$$

Now, let us assume that $I \subseteq M \times M$ is given. Then, the pairwise decision function f obtained by a pairwise SVM can be written as

$$f(a,b) := \sum_{(i,j)\in I} \alpha_{ij} y_{ij} K((x_i,x_j),(a,b)) + \gamma \tag{3}$$

with bias $\gamma \in \mathbb{R}$ and $\alpha_{ij} \geq 0$ for all $(i,j) \in I$. Obviously, if K_D (1) or K_T (2) are used, then the decision function is not symmetric in general. This motivates us to call a kernel *K balanced* if

$$K((a,b),(c,d)) = K((a,b),(d,c)) \quad \text{for all } a,b,c,d \in X$$

holds. Thus, if a balanced kernel is used, then (3) is always a symmetric decision function. For instance, the following kernels are balanced

$$K_{DL}((a,b),(c,d)) := \frac{1}{2}\left(k(a,c)+k(a,d)+k(b,c)+k(b,d)\right), \tag{4}$$

$$K_{TL}((a,b),(c,d)) := \frac{1}{2}\left(k(a,c)k(b,d)+k(a,d)k(b,c)\right), \tag{5}$$

$$K_{ML}((a,b),(c,d)) := \frac{1}{4}\left(k(a,c)-k(a,d)-k(b,c)+k(b,d)\right)^2, \tag{6}$$

$$K_{TM}((a,b),(c,d)) := K_{TL}((a,b),(c,d)) + K_{ML}((a,b),(c,d)). \tag{7}$$

Vert et al. (2007) call K_{ML} *metric learning pairwise kernel* and K_{TL} *tensor learning pairwise kernel*. Similarly, we call K_{DL}, which was introduced in Bar-Hillel et al. (2004a), *direct sum learning pairwise kernel* and K_{TM} *tensor metric learning pairwise kernel*. For representing some balanced kernels by projections see Brunner et al. (2011).

3. Symmetric Pairwise Decision Functions and Pairwise SVMs

Pairwise SVMs lead to decision functions of the form (3). As detailed above, if a balanced kernel is used within a pairwise SVM, one always obtains a symmetric decision function. For pairwise SVMs which use K_D (1) as pairwise kernel, it has been claimed that any symmetric set of training pairs leads to a symmetric decision function (see Bar-Hillel et al., 2004a). We call a set of training pairs symmetric, if for any training pair (a,b) the pair (b,a) also belongs to the training set. In Section 3.1 we prove the claim of Bar-Hillel et al. (2004a) in a more general context which includes K_T (2). Additionally, we show in Section 3.2 that under some conditions a symmetric training set leads to the same decision function as balanced kernels if we disregard the SVM bias term γ. Interestingly, the application of balanced kernels leads to significantly shorter training times (see Section 4.2).

3.1 Symmetric Training Sets

In this subsection we show that the symmetry of a pairwise decision function is indeed achieved by means of symmetric training sets. To this end, let $I \subseteq M \times M$ be a symmetric index set, in other words if (i, j) belongs to I then (j, i) also belongs to I. Furthermore, we will make use of pairwise kernels K with

$$K((a,b),(c,d)) = K((b,a),(d,c)) \quad \text{for all } a,b,c,d \in X. \tag{8}$$

As any pairwise kernel is assumed to be symmetric, (8) holds for any balanced pairwise kernel. Note that there are other pairwise kernels that satisfy (8), for instance for the kernels given in Equations 1 and 2.

For $I_R, I_N \subseteq I$ defined by $I_R := \{(i, j) \in I | i = j\}$ and $I_N := I \setminus I_R$ let us consider the dual pairwise SVM

$$\begin{aligned} &\min_{\alpha} \; G(\alpha) \\ \text{s. t.} \quad &0 \leq \alpha_{ij} \leq C \quad \text{for all } (i,j) \in I_N \\ &0 \leq \alpha_{ii} \leq 2C \quad \text{for all } (i,i) \in I_R \\ &\sum_{(i,j)\in I} y_{ij}\alpha_{ij} = 0. \end{aligned} \tag{9}$$

with

$$G(\alpha) := \frac{1}{2} \sum_{(i,j),(k,l)\in I} \alpha_{ij}\alpha_{kl}y_{ij}y_{kl}K((x_i,x_j),(x_k,x_l)) - \sum_{(i,j)\in I} \alpha_{ij}.$$

Lemma 1 *If I is a symmetric index set and if* (8) *holds, then there is a solution $\hat{\alpha}$ of* (9) *with $\hat{\alpha}_{ij} = \hat{\alpha}_{ji}$ for all $(i, j) \in I$.*

Proof By the theorem of Weierstrass there is a solution α^* of (9). Let us define another feasible point $\tilde{\alpha}$ of (9) by

$$\tilde{\alpha}_{ij} := \alpha^*_{ji} \quad \text{for all } (i,j) \in I.$$

For easier notation we set $K_{ij,kl} := K((x_i,x_j),(x_k,x_l))$. Then,

$$2G(\tilde{\alpha}) = \sum_{(i,j),(k,l)\in I} \alpha^*_{ji}\alpha^*_{lk}y_{ij}y_{kl}K_{ij,kl} - 2\sum_{(i,j)\in I} \alpha^*_{ji}.$$

Note that $y_{ij} = y_{ji}$ holds for all $(i, j) \in I$. By (8) we further obtain

$$2G(\tilde{\alpha}) = \sum_{(i,j),(k,l)\in I} \alpha^*_{ji}\alpha^*_{lk}y_{ji}y_{lk}K_{ji,lk} - 2\sum_{(i,j)\in I} \alpha^*_{ji} = 2G(\alpha^*).$$

The last equality holds since I is a symmetric training set. Hence, $\tilde{\alpha}$ is also a solution of (9). Since (9) is convex (cf. Schölkopf and Smola, 2001),

$$\alpha^\lambda := \lambda\alpha^* + (1-\lambda)\tilde{\alpha}$$

solves (9) for any $\lambda \in [0, 1]$. Thus, $\hat{\alpha} := \alpha^{1/2}$ has the desired property. ■

Note that a result similar to Lemma 1 is presented by Wei et al. (2006) for Support Vector Regression. They, however, claim that any solution of the corresponding quadratic program has the described property.

Theorem 2 *If I is a symmetric index set and if* (8) *holds, then any solution α of the optimization problem* (9) *leads to a symmetric pairwise decision function $f : X \times X \to \mathbb{R}$.*

Proof For any solution α of (9) let us define $g_\alpha : X \times X \to \mathbb{R}$ by

$$g_\alpha(a,b) \coloneqq \sum_{(i,j)\in I} \alpha_{ij} y_{ij} K((x_i,x_j),(a,b)).$$

Then, the obtained decision function can be written as $f_\alpha(a,b) = g_\alpha(a,b) + \gamma$ for some appropriate $\gamma \in \mathbb{R}$. If α^1 and α^2 are solutions of (9) then $g_{\alpha^1} = g_{\alpha^2}$ can be derived by means of convex optimization theory. According to Lemma 1 there is always a solution $\hat{\alpha}$ of (9) with $\hat{\alpha}_{ij} = \hat{\alpha}_{ji}$ for all $(i,j) \in I$. Obviously, such a solution leads to a symmetric decision function $f_{\hat{\alpha}}$. Hence, f_α is a symmetric decision function for all solutions α. ■

3.2 Balanced Kernels vs. Symmetric Training Sets

Section 2 shows that one can use balanced kernels to obtain a symmetric pairwise decision function by means of a pairwise SVM. As detailed in Section 3.1 this can also be achieved by symmetric training sets. Now, we show in Theorem 3 that the decision function is the same, regardless whether a symmetric training set or a certain balanced kernel is used. This result is also of practical value, since the approach with balanced kernels leads to significantly shorter training times (see the empirical results in Section 4.2).

Suppose J is a largest subset of a given symmetric index set I satisfying

$$((i,j) \in J \wedge j \neq i) \quad \Rightarrow \quad (j,i) \notin J.$$

Now, we consider the optimization problem

$$\begin{aligned} &\min_{\beta} H(\beta) \\ \text{s.t.} \quad &0 \leq \beta_{ij} \leq 2C \quad \text{for all } (i,j) \in J \\ &\sum_{(i,j)\in J} y_{ij}\beta_{ij} = 0 \end{aligned} \tag{10}$$

with

$$H(\beta) \coloneqq \frac{1}{2} \sum_{(i,j),(k,l)\in J} \beta_{ij}\beta_{kl} y_{ij} y_{kl} \hat{K}_{ij,kl} - \sum_{(i,j)\in J} \beta_{ij}$$

and

$$\hat{K}_{ij,kl} \coloneqq \frac{1}{2}\left(K_{ij,kl} + K_{ji,kl}\right), \tag{11}$$

where K is an arbitrary pairwise kernel. Obviously, $\hat{K}$ is a balanced kernel. For instance, if $K = K_D$ (1) then $\hat{K} = K_{DL}$ (4) or if $K = K_T$ (2) then $\hat{K} = K_{TL}$ (5). The assumed symmetry of K yields

$$\hat{K}_{ij,kl} = \hat{K}_{ij,lk} = \hat{K}_{ji,kl} = \hat{K}_{ji,lk} = \hat{K}_{kl,ij} = \hat{K}_{lk,ij} = \hat{K}_{kl,ji} = \hat{K}_{lk,ji}. \tag{12}$$

Note that (12) holds not only for kernels given by (11) but for any balanced kernel.

Theorem 3 *Let the functions $g_\alpha : X \times X \to \mathbb{R}$ and $h_\beta : X \times X \to \mathbb{R}$ be defined by*

$$g_\alpha(a,b) \coloneqq \sum_{(i,j)\in I} \alpha_{ij} y_{ij} K((x_i,x_j),(a,b)),$$
$$h_\beta(a,b) \coloneqq \sum_{(i,j)\in J} \beta_{ij} y_{ij} \hat{K}((x_i,x_j),(a,b)),$$

where I is a symmetric index set and J is defined as above. Additionally, let K fulfill (8) *and $\hat{K}$ be given by* (11)*. Then, for any solution α^* of* (9) *and for any solution β^* of* (10) *it holds that $g_{\alpha^*} = h_{\beta^*}$.*

Proof By means of convex optimization theory it can be derived that g_α is the same function for any solution α. The same holds for h_β and any solution β. Hence, due to Lemma 1 we can assume that α^* is a solution of (9) with $\alpha^*_{ij} = \alpha^*_{ji}$. For $J_R \coloneqq I_R$ and $J_N \coloneqq J \setminus J_R$ we define $\bar{\beta}$ by

$$\bar{\beta}_{ij} \coloneqq \begin{cases} \alpha^*_{ij} + \alpha^*_{ji} & \text{if } (i,j) \in J_N, \\ \alpha^*_{ii} & \text{if } (i,j) \in J_R. \end{cases}$$

Obviously, $\bar{\beta}$ is a feasible point of (10). Then, by (11) and by $\alpha^*_{ij} = \alpha^*_{ji}$ we obtain for

$$\begin{aligned} (i,j) \in J_N: \qquad \bar{\beta}_{ij}\hat{K}_{ij,kl} &= \frac{\bar{\beta}_{ij}}{2}(K_{ij,kl} + K_{ji,kl}) = \frac{\alpha^*_{ij} + \alpha^*_{ji}}{2}\left(K_{ij,kl} + K_{ji,kl}\right) \\ &= \alpha^*_{ij}K_{ij,kl} + \alpha^*_{ji}K_{ji,kl}, \qquad (13) \\ (i,i) \in J_R: \qquad \bar{\beta}_{ii}\hat{K}_{ii,kl} &= \frac{\bar{\beta}_{ii}}{2}(K_{ii,kl} + K_{ii,kl}) = \alpha^*_{ii}K_{ii,kl}. \end{aligned}$$

Then, $y_{ij} = y_{ji}$ implies

$$h_{\bar{\beta}} = g_{\alpha^*}. \qquad (14)$$

In a second step we prove that $\bar{\beta}$ is a solution of problem (10). By using $y_{kl} = y_{lk}$, the symmetry of K, (13), (12), and the definition of $\bar{\beta}$ one obtains

$$\begin{aligned} &2G(\alpha^*) + 2\sum_{(i,j)\in I} \alpha^*_{ij} \\ &= \sum_{(i,j)\in I} \alpha^*_{ij} y_{ij} \left(\sum_{(k,l)\in J_N} y_{kl}\left(\alpha^*_{kl}K_{ij,kl} + \alpha^*_{lk}K_{ij,lk}\right) + \sum_{(k,k)\in J_R} y_{kk}\alpha^*_{kk}K_{ij,kk} \right) \\ &= \sum_{(i,j)\in J_N \cup J_R} \alpha^*_{ij} y_{ij} \sum_{(k,l)\in J} \bar{\beta}_{kl} y_{kl} \hat{K}_{ij,kl} + \sum_{(i,j)\in J_N} \alpha^*_{ji} y_{ji} \sum_{(k,l)\in J} \bar{\beta}_{kl} y_{kl} \hat{K}_{ji,kl} \\ &= \sum_{(i,j)\in J_N} \bar{\beta}_{ij} y_{ij} \sum_{(k,l)\in J} \bar{\beta}_{kl} y_{kl} \hat{K}_{ij,kl} + \sum_{(i,i)\in J_R} \bar{\beta}_{ii} y_{ii} \sum_{(k,l)\in J} \bar{\beta}_{kl} y_{kl} \hat{K}_{ii,kl} \\ &= 2H(\bar{\beta}) + 2\sum_{(i,j)\in J} \bar{\beta}_{ij}. \end{aligned}$$

Then, the definition of $\bar{\beta}$ implies

$$G(\alpha^*) = H(\bar{\beta}). \qquad (15)$$

Now, let us define $\bar{\alpha}$ by

$$\bar{\alpha}_{ij} := \begin{cases} \beta^*_{ij}/2 & \text{if } (i,j) \in J_N, \\ \beta^*_{ji}/2 & \text{if } (j,i) \in J_N, \\ \beta^*_{ii} & \text{if } (i,j) \in J_R. \end{cases}$$

Obviously, $\bar{\alpha}$ is a feasible point of (9). Then, by (8) and (11) we obtain for

$$(k,l) \in J_N: \quad \bar{\alpha}_{kl} K_{ij,kl} + \bar{\alpha}_{lk} K_{ij,lk} = \frac{\beta^*_{kl}}{2}(K_{ij,kl} + K_{ij,lk}) = \beta^*_{kl} \hat{K}_{ij,kl},$$

$$(k,k) \in J_R: \quad \bar{\alpha}_{kk} K_{ij,kk} = \frac{\beta^*_{kk}}{2}(K_{ij,kk} + K_{ij,kk}) = \beta^*_{kk} \hat{K}_{ij,kk}.$$

This, (12), and $y_{kl} = y_{lk}$ yield

$$\begin{aligned} & 2H(\beta^*) + 2 \sum_{(i,j)\in J} \beta^*_{ij} \\ &= \sum_{(i,j)\in J} \beta^*_{ij} y_{ij} \left(\sum_{(k,l)\in J_N} \beta^*_{kl} y_{kl} \frac{1}{2} \left(\hat{K}_{ij,kl} + \hat{K}_{ji,kl} \right) + \sum_{(k,k)\in J_R} \beta^*_{kk} y_{kk} \frac{1}{2} \left(\hat{K}_{ij,kk} + \hat{K}_{ji,kk} \right) \right) \\ &= \frac{1}{2} \sum_{(i,j)\in J} \beta^*_{ij} y_{ij} \left(\sum_{(k,l)\in I} \bar{\alpha}_{kl} y_{kl} \left(K_{ij,kl} + K_{ji,kl} \right) \right). \end{aligned}$$

Then, the definition of $\bar{\alpha}$ provides $\beta^*_{ij} = \bar{\alpha}_{ij} + \bar{\alpha}_{ji}$ for $(i,j) \in J_N$ and $\bar{\alpha}_{ij} = \bar{\alpha}_{ji}$. Thus,

$$2H(\beta^*) + 2 \sum_{(i,j)\in J} \beta^*_{ij} = \sum_{(i,j)\in I} \bar{\alpha}_{ij} y_{ij} \left(\sum_{(k,l)\in I} \bar{\alpha}_{kl} y_{kl} K_{ij,kl} \right) = 2G(\bar{\alpha}) + 2 \sum_{(i,j)\in I} \bar{\alpha}_{ij}$$

follows. This implies $G(\bar{\alpha}) = H(\beta^*)$. Now, let us assume that $\bar{\beta}$ is not a solution of (10). Then, $H(\beta^*) < H(\bar{\beta})$ holds and, by (15), we have

$$G(\alpha^*) = H(\bar{\beta}) > H(\beta^*) = G(\bar{\alpha}).$$

This is a contradiction to the optimality of α^*. Hence, $\bar{\beta}$ is a solution of (10) and $h_{\beta^*} = h_{\bar{\beta}}$ follows. Then, with (14) we have the desired result. ■

4. Implementation

One of the most widely used techniques for solving SVMs efficiently is the sequential minimal optimization (SMO) (Platt, 1999). A well known implementation of this technique is LIBSVM (Chang and Lin, 2011). Empirically, SMO scales quadratically with the number of training points (Platt, 1999). Note that in pairwise classification the training points are the training pairs. If all possible training pairs are used, then the number of training pairs grows quadratically with the number m of training examples. Hence, the runtime of LIBSVM would scale quartically with m. In Section 4.1 we discuss how the costs for evaluating pairwise kernels, which can be expressed by standard kernels, can be drastically reduced. In Section 3 we discussed that one can either use balanced kernels or symmetric training sets to enforce the symmetry of a pairwise decision function. Additionally, we showed that both approaches lead to the same decision function. Section 4.2 compares the needed training times of the approach with balanced kernels and the approach with symmetric training sets.

4.1 Caching the Standard Kernel

In this subsection balanced kernels are used to enforce the symmetry of the pairwise decision function. Kernel evaluations are crucial for the performance of LIBSVM. If we could cache the whole kernel matrix in RAM we would get a huge increase of speed. Today, this seems impossible for significantly more than 125,250 training pairs as storing the (symmetric) kernel matrix for this number of pairs in double precision needs approximately 59GB. Note that training sets with 500 training examples already result in 125,250 training pairs. Now, we describe how the costs of kernel evaluations can be drastically reduced. For example, let us select the kernel K_{TL} (5) with an arbitrary standard kernel. For a single evaluation of K_{TL} the standard kernel has to be evaluated four times with vectors of X. Afterwards, four arithmetic operations are needed.

It is easy to see that each standard kernel value is used for evaluating many different elements of the kernel matrix. In general, it is possible to cache the standard kernel values for all training examples. For example, to cache the standard kernel values for 10,000 examples one needs 400MB. Thus, each kernel evaluation of K_{TL} costs four arithmetic operations only. This does not depend on the chosen standard kernel.

Table 1 compares the training times with and without caching the standard kernel values. For these measurements examples from the double interval task (cf. Section 5.1) are used where each class is represented by 5 examples, K_{TL} is chosen as pairwise kernel with a linear standard kernel, a cache size of 100MB is selected for caching pairwise kernel values, and all possible pairs are used for training. In Table 1a the training set of each run consists of $m = 250$ examples of 50 classes with different dimensions n. Table 1b shows results for different numbers m of examples of dimension $n = 500$. The speedup factor by the described caching technique is up to 100.

Dimension n of examples	Standard kernel (time in mm:ss) not cached	cached
200	2:08	0:07
400	4:31	0:07
600	6:24	0:07
800	9:41	0:08
1000	11:27	0:09

(a) Different dimensions n of examples

Number m of examples	Standard kernel (time in hh:mm) not cached	cached
200	0:04	0:00
400	1:05	0:01
600	4:17	0:02
800	12:40	0:06
1000	28:43	0:13

(b) Different numbers m of examples

Table 1: Training time with and without caching the standard kernel

4.2 Balanced Kernels vs. Symmetric Training Sets

Theorem 3 shows that pairwise SVMs which use symmetric training sets and pairwise SVMs with balanced kernels lead to the same decision function. For symmetric training sets the number of training pairs is nearly doubled compared to the number in the case of balanced kernels. Simultaneously, (11) shows that evaluating a balanced kernel is computationally more expensive compared to the corresponding non balanced kernel.

Table 2 compares the needed training time of both approaches. There, examples from the double interval task (cf. Section 5.1) of dimension $n = 500$ are used where each class is represented by 5 examples, K_T and its balanced version K_{TL} with linear standard kernels are chosen as pairwise kernel, a cache size of 100MB is selected for caching the pairwise kernel values, and all possible pairs are used for training. It turns out, that the approach with balanced kernels is three to four times faster than using symmetric training sets. Of course, the technique of caching the standard kernel values as described in Section 4.1 is used within all measurements.

Number m of examples	Symmetric training set (t in hh:mm)	Balanced kernel (t in hh:mm)
500	0:03	0:01
1000	0:46	0:17
1500	3:26	0:56
2000	9:44	2:58
2500	23:15	6:20

Table 2: Training time for symmetric training sets and for balanced kernels

5. Classification Experiments

In this section we will present results of applying pairwise SVMs to one synthetic data set and to one real world data set. Before we come to those data sets in Sections 5.1 and 5.2 we introduce K_{TL}^{lin} and K_{TL}^{poly}. Those kernels denote K_{TL} (5) with linear standard kernel and homogenous polynomial standard kernel of degree two, respectively. The kernels K_{ML}^{lin}, K_{ML}^{poly}, K_{TM}^{lin}, and K_{TM}^{poly} are defined analogously. In the following, detection error trade-off curves (DET curves cf. Gamassi et al., 2004) will be used to measure the performance of a pairwise classifier. Such a curve shows for any false match rate (FMR) the corresponding false non match rate (FNMR). A special point of interest of such a curve is the (approximated) equal error rate (EER), that is the value for which FMR=FNMR holds.

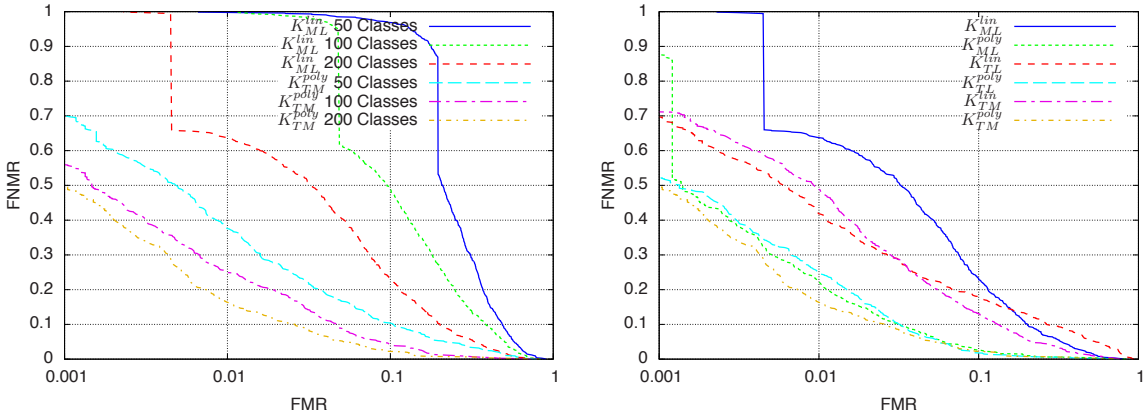
5.1 Double Interval Task

Let us describe the *double interval task* of dimension n. To get such an example $x \in \{-1,1\}^n$ one draws $i,j,k,l \in \mathbb{N}$ so that $2 \le i \le j, j+2 \le k \le l \le n$ and defines

$$x_p := \begin{cases} 1 & p \in \{i,\ldots,j\} \cup \{k,\ldots,l\}, \\ -1 & \text{otherwise.} \end{cases}$$

The class c of such an example is given by $c(x) := (i,k)$. Note that the pair (j,l) does not influence the class. Hence, there are $(n-3)(n-2)/2$ classes.

For our measurements we selected $n = 500$ and tested all kernels in (4)–(7) with a linear standard kernel and a homogenous polynomial standard kernel of degree two, respectively. We created a test set consisting of 750 examples of 50 classes so that each class is represented by 15 examples. Any training set was generated in such a way that the set of classes in the training set is disjoint from the



(a) Different class numbers in training

(b) Different kernels for 200 classes in training

Figure 1: DET curves for double interval task

set of classes in the test set. We created training sets consisting of 50 classes and different numbers of examples per class. For training all possible training pairs were used.

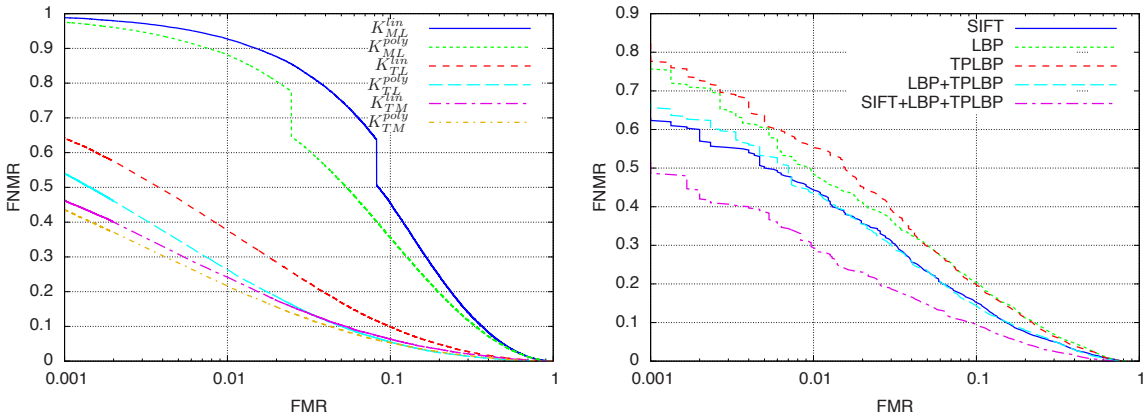
We observed that an increasing number of examples per class improves the performance independently of the other parameters. As a trade-off between the needed training time and performance of the classifier, we decided to use 15 examples per class for the measurements. Independently of the selected kernel, a penalty parameter C of 1,000 turned out to be a good choice. The kernel K_{DS} led to a bad performance regardless of the standard kernel chosen. Therefore, we omit results for K_{DS}.

Figure 1a shows that an increasing number of classes in the training set improves the performance significantly. This holds for all kernels mentioned above. Here, we only present results for K^{lin}_{ML} and K^{poly}_{TM}. Figure 1b shows the DET curves for different kernels where the training set consists of 200 classes. In particular, any of the pairwise kernels which uses a homogeneous polynomial of degree 2 as standard kernel leads to better results than its corresponding counterpart with a linear standard kernel. For FMRs smaller than 0.07 K^{poly}_{TM} leads to the best results, whereas for larger FMRs the DET curves of K^{poly}_{ML}, K^{poly}_{TL}, and K^{poly}_{TM} intersect.

5.2 Labeled Faces in the Wild

In this subsection we will present results of applying pairwise SVMs to the labeled faces in the wild (LFW) data set (Huang et al., 2007). This data set consists of 13,233 images of 5,749 persons. Several remarks on this data set are in order. Huang et al. (2007) suggest two protocols for performance measurements. Here, the unrestricted protocol is used. This protocol is a fixed tenfold cross validation where each test set consists of 300 positive pairs and 300 negative pairs. Moreover, any person (class) in a training set is not part of the corresponding test set.

There are several feature vectors available for the LFW data set. For the presented measurements we mainly followed Li et al. (2012) and used the scale-invariant feature transform (SIFT)-based feature vectors for the funneled version (Guillaumin et al., 2009) of LFW. In addition, the aligned images (Wolf et al., 2009) are used. For this, the aligned images are cropped to 80×150 pixels and are then normalized by passing them through a log function (cf. Li et al., 2012). Afterwards, the



(a) View 1 partition, different kernels, added up decision function values of SIFT, LPB, and TPLBP feature vectors

(b) Unrestricted protocol, K_{TM}^{poly}, different feature vectors, "+" stands for adding up the corresponding decision function values

Figure 2: DET curves for LFW data set

local binary patterns (LBP) (Ojala et al., 2002) and three-patch LBP (TPLBP) (Wolf et al., 2008) are extracted. In contrast to Li et al. (2012), the pose is neither estimated nor swapped and no PCA is applied to the data. As the norm of the LBP feature vectors is not the same for all images we scaled them to Euclidean norm 1.

For model selection, the View 1 partition of the LFW database is recommended (Huang et al., 2007). Using all possible pairs of this partition for training and for testing, we obtained that a penalty parameter C of 1,000 is suitable. Moreover, for each used feature vector, the kernel K_{TM}^{poly} leads to the best results among all used kernels and also if sums of decision function values belonging to SIFT, LBP, and TPLBP feature vectors are used. For example, Figure 2a shows the performance of different kernels, where the decision function values corresponding to SIFT, LBP, and TPLBP feature vectors are added up.

Due to the speed up techniques presented in Section 4 we were able to train with large numbers of training pairs. However, if all pairs were used for training, then any training set would consist of approximately 50,000,000 pairs and the training would still need too much time. Hence, whereas in any training set all positive training pairs were used, the negative training pairs were randomly selected in such a way that any training set consists of 2,000,000 pairs. The training of such a model took less than 24 hours on a standard PC. In Figure 2b we present the average DET curves obtained for K_{TM}^{poly} and feature vectors based on SIFT, LBP, and TPLBP. Inspired by Li et al. (2012), we determined two further DET curves by adding up the decision function values. This led to very good results. Furthermore, we concatenated the SIFT, LBP, and TPLBP feature vectors. Surprisingly, the training of some of those models needed longer than a week. Therefore, we do not present these results.

In Table 3 the mean equal error rate (EER) and the standard error of the mean (SEM) obtained from the tenfold cross validation are provided for several types of feature vectors. Note, that many of our results are comparable to the state of the art or even better. The current state of the art can be found on the homepage of Huang et al. (2007) and in the publication of Li et al. (2012). If only SIFT-based feature vectors are used, then the best known result is 0.125 ± 0.0040 (EER $\pm$ SEM). With

pairwise SVMs we achieved the same EER but a slightly higher SEM 0.1252 ± 0.0062. If we add up the decision function values corresponding to the LBP and TPLBP feature vectors, then our result 0.1210 ± 0.0046 is worse compared to the state of the art 0.1050 ± 0.0051. One possible reason for this fact might be that we did not swap the pose. Finally, for the added up decision function values corresponding to SIFT, LBP and TPLBP feature vectors, our performance 0.0947 ± 0.0057 is better than 0.0993 ± 0.0051. Furthermore, it is worth noting that our standard errors of the mean are comparable to the other presented learning algorithms although most of them use a PCA to reduce noise and dimension of the feature vectors. Note that the results of the commercial system are not directly comparable since it uses outside training data (for reference see Huang et al., 2007).

		SIFT	LBP	TPLBP	L+T	S+L+T	CS
Pairwise	Mean EER	0.1252	0.1497	0.1452	0.1210	0.0947	-
SVM	SEM	0.0062	0.0052	0.0060	0.0046	0.0057	-
State of	Mean EER	0.1250	0.1267	0.1630	0.1050	0.0993	0.0870
the Art	SEM	0.0040	0.0055	0.0070	0.0051	0.0051	0.0030

Table 3: Mean EER and SEM for LFW data set. S=SIFT, L=LBP, T=TPLBP, +=adding up decision function values, CS=Commercial system `face.com r2011b`

6. Final Remarks

In this paper we suggested the SVM framework for handling large pairwise classification problems. We analyzed two approaches to enforce the symmetry of the obtained classifiers. To the best of our knowledge, we gave the first proof that symmetry is indeed achieved. Then, we proved that for each parameter set of one approach there is a corresponding parameter set of the other one such that both approaches lead to the same classifier. Additionally, we showed that the approach based on balanced kernels leads to shorter training times.

We discussed details of the implementation of a pairwise SVM solver and presented numerical results. Those results demonstrate that pairwise SVMs are capable of successfully treating large scale pairwise classification problems. Furthermore, we showed that pairwise SVMs compete very well for a real world data set.

We would like to underline that some of the discussed techniques could be transferred to other approaches for solving pairwise classification problems. For example, most of the results can be applied easily to One Class Support Vector Machines (Schölkopf et al., 2001).

Acknowledgments

We would like to thank the unknown referees for their valuable comments and suggestions.

References

J. Abernethy, F. Bach, T. Evgeniou, and J.-P. Vert. A new approach to collaborative filtering: Operator estimation with spectral regularization. *Journal of Machine Learning Research*, 10:803–826, 2009.

A. Bar-Hillel and D. Weinshall. Learning distance function by coding similarity. In Z. Ghahramani, editor, *Proceedings of the 24th International Conference on Machine Learning (ICML '07)*, pages 65–72. ACM, 2007.

A. Bar-Hillel, T. Hertz, and D. Weinshall. Boosting margin based distance functions for clustering. In C. E. Brodley, editor, *In Proceedings of the 21st International Conference on Machine Learning (ICML '04)*, pages 393–400. ACM, 2004a.

A. Bar-Hillel, T. Hertz, and D. Weinshall. Learning distance functions for image retrieval. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR '04)*, volume 2, pages 570–577. IEEE Computer Society Press, 2004b.

A. Ben-Hur and W. Stafford Noble. Kernel methods for predicting protein–protein interactions. *Bioinformatics*, 21(1):38–46, 2005.

C. Brunner, A. Fischer, K. Luig, and T. Thies. Pairwise kernels, support vector machines, and the application to large scale problems. Technical Report MATH-NM-04-2011, Institute of Numerical Mathematics, Technische Universität Dresden, October 2011. URL `http://www.math.tu-dresden.de/~fischer`.

C.-C. Chang and C.-J. Lin. LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2(3):1–26, 2011. URL `http://www.csie.ntu.edu.tw/~cjlin/libsvm (August 2011)`.

K. Duan and S. S. Keerthi. Which is the best multiclass SVM method? An empirical study. In N. C. Oza, R. Polikar, J. Kittler, and F. Roli, editors, *Proceedings of the 6th International Workshop on Multiple Classifier Systems*, pages 278–285. Springer, 2005.

M. Gamassi, M. Lazzaroni, M. Misino, V. Piuri, D. Sana, and F. Scotti. Accuracy and performance of biometric systems. In *Proceedings of the 21th IEEE Instrumentation and Measurement Technology Conference (IMTC '04)*, pages 510–515. IEEE, 2004.

M. Guillaumin, J. Verbeek, and C. Schmid. Is that you? Metric learning approaches for face identification. In *Proceedings of the 12th International Conference on Computer Vision (ICCV '09)*, pages 498–505, 2009. URL `http://lear.inrialpes.fr/pubs/2009/GVS09 (August 2011)`.

S. I. Hill and A. Doucet. A framework for kernel-based multi-category classification. *Journal of Artificial Intelligence Research*, 30(1):525–564, 2007.

C.-W. Hsu and C.-J. Lin. A comparison of methods for multiclass support vector machines. *IEEE Transactions on Neural Networks*, 13(2):415–425, 2002.

G. B. Huang, M. Ramesh, T. Berg, and E. Learned-Miller. Labeled faces in the wild: A database for studying face recognition in unconstrained environments. Technical Report 07-49, University of Massachusetts, Amherst, October 2007. URL `http://vis-www.cs.umass.edu/lfw/ (August 2011)`.

P. Li, Y. Fu, U. Mohammed, J. H. Elder, and S. J. D. Prince. Probabilistic models for inference about identity. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34:144–157, 2012.

T. Ojala, M. Pietikäinen, and T. Mäenpää. Multiresolution gray-scale and rotation invariant texture classification with local binary patterns. In *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(7):971–987, 2002. URL `http://www.cse.oulu.fi/MVG/Downloads/LBPMatlab (August 2011)`.

P. J. Phillips. Support vector machines applied to face recognition. In M. S. Kearns, S. A. Solla, and D. A. Cohn, editors, *Advances in Neural Information Processing Systems 11*, pages 803–809. MIT Press, 1999.

J. C. Platt. Fast training of support vector machines using sequential minimal optimization. In B. Schölkopf, C. J. C. Burges, and A. J. Smola, editors, *Advances in Kernel Methods: Support Vector Learning*, pages 185–208. MIT Press, 1999.

R. Rifkin and A. Klautau. In defense of one-vs-all classification. *Journal of Machine Learning Research*, 5:101–141, 2004.

B. Schölkopf and A. J. Smola. *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. MIT Press, 2001.

B. Schölkopf, J. C. Platt, J. Shawe-Taylor, A. J. Smola, and R. C. Williamson. Estimating the support of a high-dimensional distribution. *Neural Computations*, 13(7):1443–1471, 2001.

J. P. Vert, J. Qiu, and W. Noble. A new pairwise kernel for biological network inference with support vector machines. *BMC Bioinformatics*, 8(Suppl 10):S8, 2007.

L. Wei, Y. Yang, R. M. Nishikawa, and M. N. Wernick. Learning of perceptual similarity from expert readers for mammogram retrieval. In *Proceedings of the IEEE International Symposium on Biomedical Imaging (ISBI)*, pages 1356–1359. IEEE, 2006.

L. Wolf, T. Hassner, and Y. Taigman. Descriptor based methods in the wild. In *Faces in Real-Life Images Workshop at the European Conference on Computer Vision (ECCV '08)*, 2008. URL `http://www.openu.ac.il/home/hassner/projects/Patchlbp (August 2011)`.

L. Wolf, T. Hassner, and Y. Taigman. Similarity scores based on background samples. In *Proceedings of the 9th Asian Conference on Computer Vision (ACCV '09)*, volume 2, pages 88–97, 2009.

Journal of Machine Learning Research 13 (2012) 2293-2337 Submitted 7/11; Revised 4/12; Published 8/12

High-Dimensional Gaussian Graphical Model Selection: Walk Summability and Local Separation Criterion

Animashree Anandkumar A.ANANDKUMAR@UCI.EDU
Electrical Engineering and Computer Science
University of California, Irvine
Irvine, CA 92697

Vincent Y. F. Tan TANYFV@I2R.A-STAR.EDU.SG
Data Mining Department
Institute for Infocomm Research
Singapore
Electrical and Computer Engineering, National University of Singapore

Furong Huang FURONGH@UCI.EDU
Electrical Engineering and Computer Science
University of California, Irvine
Irvine, CA 92697

Alan S. Willsky WILLSKY@MIT.EDU
Stochastic Systems Group
Laboratory for Information and Decision Systems
Massachusetts Institute of Technology
Cambridge, MA 02139

Editor: Martin Wainwright

Abstract

We consider the problem of high-dimensional Gaussian graphical model selection. We identify a set of graphs for which an efficient estimation algorithm exists, and this algorithm is based on thresholding of empirical conditional covariances. Under a set of transparent conditions, we establish structural consistency (or *sparsistency*) for the proposed algorithm, when the number of samples $n = \Omega(J_{\min}^{-2} \log p)$, where p is the number of variables and $J_{\min}$ is the minimum (absolute) edge potential of the graphical model. The sufficient conditions for sparsistency are based on the notion of *walk-summability* of the model and the presence of sparse *local vertex separators* in the underlying graph. We also derive novel non-asymptotic necessary conditions on the number of samples required for sparsistency.

Keywords: Gaussian graphical model selection, high-dimensional learning, local-separation property, walk-summability, necessary conditions for model selection

1. Introduction

Probabilistic graphical models offer a powerful formalism for representing high-dimensional distributions succinctly. In an undirected graphical model, the conditional independence relationships among the variables are represented in the form of an undirected graph. Learning graphical models using its observed samples is an important task, and involves both structure and parameter estima-

tion. While there are many techniques for parameter estimation (e.g., expectation maximization), structure estimation is arguably more challenging. High-dimensional structure estimation is NP-hard for general models (Karger and Srebro, 2001; Bogdanov et al., 2008) and moreover, the number of samples available for learning is typically much smaller than the number of dimensions (or variables).

The complexity of structure estimation depends crucially on the underlying graph structure. Chow and Liu (1968) established that structure estimation in tree models reduces to a maximum weight spanning tree problem and is thus computationally efficient. However, a general characterization of graph families for which structure estimation is tractable has so far been lacking. In this paper, we present such a characterization based on the so-called *local separation* property in graphs. It turns out that a wide variety of (random) graphs satisfy this property (with probability tending to one) including large girth graphs, the Erdős-Rényi random graphs (Bollobás, 1985) and the power-law graphs (Chung and Lu, 2006), as well as graphs with short cycles such as the small-world graphs (Watts and Strogatz, 1998) and other hybrid/augmented graphs (Chung and Lu, 2006, Ch. 12). The small world and augmented graphs are especially relevant for modeling data from social networks. Note that these graphs can simultaneously possess many short cycles as well as large node degrees (growing with the number of nodes), and thus, we can incorporate a wide class of graphs for high-dimensional estimation.

Successful structure estimation also relies on certain assumptions on the parameters of the model, and these assumptions are tied to the specific algorithm employed. For instance, for convex-relaxation approaches (Meinshausen and Bühlmann, 2006; Ravikumar et al., 2011), the assumptions are based on certain *incoherence* conditions on the model, which are hard to interpret as well as verify in general. In this paper, we present a set of transparent conditions for Gaussian graphical model selection based on *walk-sum* analysis (Malioutov et al., 2006). Walk-sum analysis has been previously employed to analyze the performance of loopy belief propagation (LBP) and its variants in Gaussian graphical models. In this paper, we demonstrate that walk-summability also turns out to be a natural criterion for efficient structure estimation, thereby reinforcing its importance in characterizing the tractability of Gaussian graphical models.

1.1 Summary of Results

Our main contributions in this work are threefold. We propose a simple local algorithm for Gaussian graphical model selection, termed as conditional covariance threshold test (CMIT) based on a set of conditional covariance thresholding tests. Second, we derive sample complexity results for our algorithm to achieve structural consistency (or sparsistency). Third, we prove a novel non-asymptotic lower bound on the sample complexity required by any learning algorithm to succeed. We now elaborate on these contributions.

Our structure learning procedure is known as the Conditional Covariance Test[1] (CMIT) and is outlined in Algorithm 1. Let $\mathsf{CMIT}(\mathbf{x}^n; \xi_{n,p}, \eta)$ be the output edge set from CMIT given n i.i.d. samples $\mathbf{x}^n$, a threshold $\xi_{n,p}$ (that depends on both p and n) and a constant $\eta \in \mathbb{N}$, which is related to the local vertex separation property (described later). The conditional covariance test proceeds

1. An analogous test is employed for Ising model selection in Anandkumar et al. (2012b) based on conditional mutual information. We later note that conditional mutual information test has slightly worse sample complexity for learning Gaussian models.

Algorithm 1 Algorithm $\mathsf{CMIT}(\mathbf{x}^n; \xi_{n,p}, \eta)$ for structure learning using samples $\mathbf{x}^n$.

Initialize $\widehat{G}_p^n = (V, \emptyset)$.
For each $i, j \in V$, if

$$\min_{\substack{S \subset V \setminus \{i,j\} \\ |S| \leq \eta}} |\widehat{\Sigma}(i,j|S)| > \xi_{n,p}, \tag{1}$$

then add (i, j) to $\widehat{G}_p^n$.
Output: $\widehat{G}_p^n$.

in the following manner. First, the empirical absolute conditional covariances[2] are computed as follows:

$$\widehat{\Sigma}(i,j|S) := \widehat{\Sigma}(i,j) - \widehat{\Sigma}(i,S)\widehat{\Sigma}^{-1}(S,S)\widehat{\Sigma}(S,j),$$

where $\widehat{\Sigma}(\cdot,\cdot)$ are the respective empirical variances. Note that $\widehat{\Sigma}^{-1}(S,S)$ exists when the number of samples satisfies $n > |S|$ (which is the regime under consideration). The conditional covariance is thus computed for each node pair $(i, j) \in V^2$ and the conditioning set which achieves the minimum is found, over all subsets of cardinality at most η; if the minimum value exceeds the threshold $\xi_{n,p}$, then the node pair is declared as an edge. See Algorithm 1 for details.

The computational complexity of the algorithm is $O(p^{\eta+2})$, which is efficient for small η. For the so-called *walk-summable* Gaussian graphical models, the parameter η can be interpreted as an upper bound on the size of local vertex separators in the underlying graph. Many graph families have small η and as such, are amenable to computationally efficient structure estimation by our algorithm. These include Erdős-Rényi random graphs, power-law graphs and small-world graphs, as discussed previously.

We establish that the proposed algorithm has a sample complexity of $n = \Omega(J_{\min}^{-2} \log p)$, where p is the number of nodes (variables) and $J_{\min}$ is the minimum (absolute) edge potential in the model. As expected, the sample complexity improves when $J_{\min}$ is large, that is, the model has strong edge potentials. However, as we shall see, $J_{\min}$ cannot be arbitrarily large for the model to be walk-summable. We derive the minimum sample complexity for various graph families and show that this minimum is attained when $J_{\min}$ takes the maximum possible value.

We also develop novel techniques to obtain necessary conditions for consistent structure estimation of Erdős-Rényi random graphs and other ensembles with non-uniform distribution of graphs. We obtain non-asymptotic bounds on the number of samples n in terms of the expected degree and the number of nodes of the model. The techniques employed are information-theoretic in nature (Cover and Thomas, 2006). We cast the learning problem as a source-coding problem and develop necessary conditions which combine the use of Fano's inequality with the so-called asymptotic equipartition property.

Our sufficient conditions for structural consistency are based on walk-summability. This characterization is novel to the best of our knowledge. Previously, walk-summable models have been extensively studied in the context of inference in Gaussian graphical models. As a by-product of our analysis, we also establish the correctness of loopy belief propagation for walk-summable Gaussian graphical models Markov on locally tree-like graphs (see Section 5 for details). This suggests

2. Alternatively, conditional independence can be tested via sample partial correlations which can be computed via regression or recursion. See Kalisch and Bühlmann (2007) for details.

that walk-summability is a fundamental criterion for tractable learning and inference in Gaussian graphical models.

1.2 Related Work

Given that structure learning of general graphical models is NP-hard (Karger and Srebro, 2001; Bogdanov et al., 2008), the focus has been on characterizing classes of models on which learning is tractable. The seminal work of Chow and Liu (1968) provided an efficient implementation of maximum-likelihood structure estimation for tree models via a maximum weighted spanning tree algorithm. Error-exponent analysis of the Chow-Liu algorithm was studied (Tan et al., 2011a, 2010) and extensions to general forest models were considered by Tan et al. (2011b) and Liu et al. (2011). Learning trees with latent (hidden) variables (Choi et al., 2011) have also been studied recently.

For graphical models Markov on general graphs, alternative approaches are required for structure estimation. A recent paradigm for structure estimation is based on convex relaxation, where an estimate is obtained via convex optimization which incorporates an ℓ_1-based penalty term to encourage sparsity. For Gaussian graphical models, such approaches have been considered in Meinshausen and Bühlmann (2006) and Ravikumar et al. (2011) and d'Aspremont et al. (2008), and the sample complexity of the proposed algorithms have been analyzed. A major disadvantage in using convex-relaxation methods is that the incoherence conditions required for consistent estimation are hard to interpret and it is not straightforward to characterize the class of models satisfying these conditions.

An alternative to the convex-relaxation approach is the use of simple greedy local algorithms for structure learning. The conditions required for consistent estimation are typically more transparent, albeit somewhat restrictive. Bresler et al. (2008) propose an algorithm for structure learning of general graphical models Markov on bounded-degree graphs, based on a series of conditional-independence tests. Abbeel et al. (2006) propose an algorithm, similar in spirit, for learning factor graphs with bounded degree. Spirtes and Meek (1995), Cheng et al. (2002), Kalisch and Bühlmann (2007) and Xie and Geng (2008) propose conditional-independence tests for learning Bayesian networks on directed acyclic graphs (DAG). Netrapalli et al. (2010) proposed a faster greedy algorithm, based on conditional entropy, for graphs with large girth and bounded degree. However, all the works (Bresler et al., 2008; Abbeel et al., 2006; Spirtes and Meek, 1995; Cheng et al., 2002; Netrapalli et al., 2010) require the maximum degree in the graph to be bounded ($\Delta = O(1)$) which is restrictive. We allow for graphs where the maximum degree can grow with the number of nodes. Moreover, we establish a natural tradeoff between the maximum degree and other parameters of the graph (e.g., girth) required for consistent structure estimation.

Necessary conditions for consistent graphical model selection provide a lower bound on sample complexity and have been explored before by Santhanam and Wainwright (2008) and Wang et al. (2010). These works consider graphs drawn uniformly from the class of bounded degree graphs and establish that $n = \Omega(\Delta^k \log p)$ samples are required for consistent structure estimation, in an p-node graph with maximum degree Δ, where k is typically a small positive integer. However, a direct application of these methods yield poor lower bounds if the ensemble of graphs has a highly non-uniform distribution. This is the case with the ensemble of Erdős-Rényi random graphs (Bollobás, 1985). Necessary conditions for structure estimation of Erdős-Rényi random graphs were derived for Ising models by Anandkumar et al. (2012b) based on an information-theoretic covering argument. However, this approach is not directly applicable to the Gaussian setting. We present a novel approach for obtaining necessary conditions for Gaussian graphical model selection based on

the notion of *typicality*. We characterize the set of typical graphs for the Erdős-Rényi ensemble and derive a modified form of Fano's inequality and obtain a non-asymptotic lower bound on sample complexity involving the average degree and the number of nodes.

We briefly also point to a large body of work on high-dimensional covariance selection under different notions of sparsity. Note that the assumption of a Gaussian graphical model Markov on a sparse graph is one such formulation. Other notions of sparsity include Gaussian models with sparse covariance matrices, or having a banded Cholesky factorization. Also, note that many works consider covariance estimation instead of selection and in general, estimation guarantees can be obtained under less stringent conditions. See Lam and Fan (2009), Rothman et al. (2008), Huang et al. (2006) and Bickel and Levina (2008) for details.

1.3 Paper Outline

The paper is organized as follows. We introduce the system model in Section 2. We prove the main result of our paper regarding the structural consistency of conditional covariance thresholding test in Section 3. We prove necessary conditions for model selection in Section 4. In Section 5, we analyze the performance of loopy belief propagation in Gaussian graphical models. Section 7 concludes the paper. Proofs and additional discussion are provided in the appendix.

2. Preliminaries and System Model

We now provide an overview of Gaussian graphical models and the problem of structure learning given samples from the model.

2.1 Gaussian Graphical Models

A Gaussian graphical model is a family of jointly Gaussian distributions which factor in accordance to a given graph. Given a graph $G = (V,E)$, with $V = \{1,\ldots,p\}$, consider a vector of Gaussian random variables $\mathbf{X} = [X_1, X_2, \ldots, X_p]^T$, where each node $i \in V$ is associated with a scalar Gaussian random variable X_i. A Gaussian graphical model Markov on G has a probability density function (pdf) that may be parameterized as

$$f_{\mathbf{X}}(\mathbf{x}) \propto \exp\left[-\frac{1}{2}\mathbf{x}^T\mathbf{J}_G\mathbf{x} + \mathbf{h}^T\mathbf{x}\right], \tag{2}$$

where $\mathbf{J}_G$ is a positive-definite symmetric matrix whose sparsity pattern corresponds to that of the graph G. More precisely,

$$J_G(i,j) = 0 \iff (i,j) \notin G.$$

The matrix $\mathbf{J}_G$ is known as the potential or information matrix, the non-zero entries $J(i,j)$ as the edge potentials, and the vector $\mathbf{h}$ as the potential vector. A model is said to be *attractive* if $J_{i,j} \leq 0$ for all $i \neq j$. The form of parameterization in (2) is known as the information form and is related to the standard mean-covariance parameterization of the Gaussian distribution as

$$\mu = \mathbf{J}^{-1}\mathbf{h}, \quad \mathbf{\Sigma} = \mathbf{J}^{-1},$$

where $\mu := \mathbb{E}[\mathbf{X}]$ is the mean vector and $\mathbf{\Sigma} := \mathbb{E}[(\mathbf{X}-\mu)(\mathbf{X}-\mu)^T]$ is the covariance matrix.

We say that a jointly Gaussian random vector $\mathbf{X}$ with joint pdf $f(\mathbf{x})$ satisfies local Markov property with respect to a graph G if

$$f(x_i|\mathbf{x}_{\mathcal{N}(i)}) = f(x_i|\mathbf{x}_{V\setminus i})$$

holds for all nodes $i \in V$, where $\mathcal{N}(i)$ denotes the set of neighbors of node $i \in V$ and, $V \setminus i$ denotes the set of all nodes excluding i. More generally, we say that $\mathbf{X}$ satisfies the global Markov property, if for all disjoint sets $A, B \subset V$, we have

$$f(\mathbf{x}_A, \mathbf{x}_B|\mathbf{x}_S) = f(\mathbf{x}_A|\mathbf{x}_S) f(\mathbf{x}_B|\mathbf{x}_S).$$

where set S is a *separator*[3] of A and B The local and global Markov properties are equivalent for non-degenerate Gaussian distributions (Lauritzen, 1996).

Our results on structure learning depend on the precision matrix $\mathbf{J}$. Let

$$J_{\min} := \min_{(i,j)\in G} |J(i,j)|,\ J_{\max} := \max_{(i,j)\in G} |J(i,j)|,\ D_{\min} := \min_i J(i,i).$$

Intuitively, models with edge potentials which are "too small" or "too large" are harder to learn than those with comparable potentials. Since we consider the high-dimensional case where the number of variables p grows, we allow the bounds $J_{\min}$, $J_{\max}$, and $D_{\min}$ to potentially scale with p.

The *partial correlation coefficient* between variables X_i and X_j, for $i \neq j$, measures their conditional covariance given all other variables. These are computed by normalizing the off-diagonal values of the information matrix, that is,

$$R(i,j) := \frac{\Sigma(i,j|V\setminus\{i,j\})}{\sqrt{\Sigma(i,i|V\setminus\{i,j\})\Sigma(j,j|V\setminus\{i,j\})}} = -\frac{J(i,j)}{\sqrt{J(i,i)J(j,j)}}. \tag{3}$$

For all $i \in V$, set $R(i,i) = 0$. We henceforth refer to $\mathbf{R}$ as the partial correlation matrix.

An important sub-class of Gaussian graphical models of the form in (19) are the *walk-summable* models (Malioutov et al., 2006). A Gaussian model is said to be α-walk summable if

$$\|\overline{\mathbf{R}}\| \leq \alpha < 1,$$

where $\overline{\mathbf{R}} := [|R(i,j)|]$ denotes the entry-wise absolute value of the partial correlation matrix $\mathbf{R}$ and $\|\cdot\|$ denotes the spectral or 2-norm of the matrix, which for symmetric matrices, is given by the maximum absolute eigenvalue.

In other words, walk-summability means that an attractive model formed by taking the absolute values of the partial correlation matrix of the Gaussian graphical model is also valid (i.e., the corresponding potential matrix is positive definite). This immediately implies that attractive models form a sub-class of walk-summable models. For detailed discussion on walk-summability, see Section A.1.

2.2 Tractable Graph Families

We consider the class of Gaussian graphical models Markov on a graph G_p belonging to some ensemble $\mathcal{G}(p)$ of graphs with p nodes. We consider the high-dimensional learning regime, where both

3. A set $S \subset V$ is a separator for sets A and B if the removal of nodes in S partitions A and B into distinct components.

p and the number of samples n grow simultaneously; typically, the growth of p is much faster than that of n. We emphasize that in our formulation the graph ensemble $\mathcal{G}(p)$ can either be deterministic or random—in the latter, we also specify a probability measure over the set of graphs in $\mathcal{G}(p)$. In the setting where $\mathcal{G}(p)$ is a random-graph ensemble, let $P_{\mathbf{X},G}$ denote the joint probability distribution of the variables $\mathbf{X}$ and the graph $G \sim \mathcal{G}(p)$, and let $f_{\mathbf{X}|G}$ denote the conditional (Gaussian) density of the variables Markov on the given graph G. Let P_G denote the probability distribution of graph G drawn from a random ensemble $\mathcal{G}(p)$. We use the term *almost every* (a.e.) graph G satisfies a certain property $\mathcal{Q}$ if

$$\lim_{p\to\infty} P_G[G \text{ satisfies } \mathcal{Q}] = 1.$$

In other words, the property $\mathcal{Q}$ holds asymptotically almost surely[4] (a.a.s.) with respect to the random-graph ensemble $\mathcal{G}(p)$. Our conditions and theoretical guarantees will be based on this notion for random graph ensembles. Intuitively, this means that graphs that have a vanishing probability of occurrence as $p \to \infty$ are ignored.

We now characterize the ensemble of graphs amenable for consistent structure estimation under our formulation. To this end, we define the concept of local separation in graphs. See Fig. 1 for an illustration. For $\gamma \in \mathbb{N}$, let $B_\gamma(i;G)$ denote the set of vertices within distance γ from i with respect to graph G. Let $H_{\gamma,i} := G(B_\gamma(i))$ denote the subgraph of G spanned by $B_\gamma(i;G)$, but in addition, we retain the nodes not in $B_\gamma(i)$ (and remove the corresponding edges). Thus, the number of vertices in $H_{\gamma,i}$ is p.

Definition 1 (γ-Local Separator) *Given a graph G, a* γ-local separator $S_\gamma(i,j)$ *between i and j, for $(i,j) \notin G$, is a* minimal *vertex separator*[5] *with respect to the subgraph $H_{\gamma,i}$. In addition, the parameter γ is referred to as the* path threshold *for local separation.*

In other words, the γ-local separator $S_\gamma(i,j)$ separates nodes i and j with respect to paths in G of length at most γ. We now characterize the ensemble of graphs based on the size of local separators.

Definition 2 ((η,γ)-Local Separation Property) *An ensemble of graphs satisfies (η,γ)-local separation property if for a.e. G_p in the ensemble,*

$$\max_{(i,j)\notin G_p} |S_\gamma(i,j)| \leq \eta. \tag{4}$$

We denote such a graph ensemble by $\mathcal{G}(p;\eta,\gamma)$.

In Section 3, we propose an efficient algorithm for graphical model selection when the underlying graph belongs to a graph ensemble $\mathcal{G}(p;\eta,\gamma)$ with sparse local separators (i.e., small η, for η defined in (4)). We will see that the computational complexity of our proposed algorithm scales as $O(p^{\eta+2})$. We now provide examples of several graph families satisfying (4).

4. Note that the term a.a.s. does not apply to deterministic graph ensembles $\mathcal{G}(p)$ where no randomness is assumed, and in this setting, we assume that the property $\mathcal{Q}$ holds for every graph in the ensemble.
5. A minimal separator is a separator of smallest cardinality.

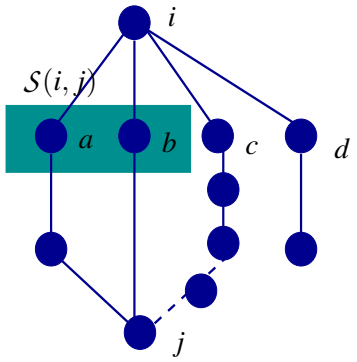


Figure 1: Illustration of l-local separator set $\mathcal{S}(i,j;G,l)$ for the graph shown above with $l=4$. Note that $\mathcal{N}(i)=\{a,b,c,d\}$ is the neighborhood of i and the l-local separator set $\mathcal{S}(i,j;G,l)=\{a,b\}\subset\mathcal{N}(i;G)$. This is because the path along c connecting i and j has a length greater than l and hence node $c\notin\mathcal{S}(i,j;G,l)$.

2.2.1 EXAMPLE 1: BOUNDED-DEGREE

We now show that the local-separation property holds for a rich class of graphs. Any (deterministic or random) ensemble of degree-bounded graphs $\mathcal{G}_{\text{Deg}}(p,\Delta)$ satisfies (η,γ)-local separation property with $\eta=\Delta$ and arbitrary $\gamma\in\mathbb{N}$. If we do not impose any further constraints on $\mathcal{G}_{\text{Deg}}$, the computational complexity of our proposed algorithm scales as $O(p^{\Delta+2})$ (see also Bresler et al., 2008 where the computational complexity is comparable). Thus, when Δ is large, our proposed algorithm and the one in Bresler et al. (2008) are computationally intensive. Our goal in this paper is to relax the usual bounded-degree assumption and to consider ensembles of graphs $\mathcal{G}(p)$ whose maximum degrees may grow with the number of nodes p. To this end, we discuss other structural constraints which can lead to graphs with sparse local separators.

2.2.2 EXAMPLE 2: BOUNDED LOCAL PATHS

Another sufficient condition[6] for the (η,γ)-local separation property in Definition 2 to hold is that there are at most η paths of length at most γ in G between any two nodes (henceforth, termed as the (η,γ)-*local paths property*). In other words, there are at most $\eta-1$ number of overlapping[7] cycles of length smaller than 2γ.

In particular, a special case of the local-paths property described above is the so-called girth property. The *girth* of a graph is the length of the shortest cycle. Thus, a graph with girth g satisfies (η,γ)-local separation property with $\eta=1$ and $\gamma=g/2$. Let $\mathcal{G}_{\text{Girth}}(p;g)$ denote the ensemble of graphs with girth at most g. There are many graph constructions which lead to large girth. For example, the bipartite Ramanujan graph (Chung, 1997, p. 107) and the random Cayley graphs (Gamburd et al., 2009) have large girths.

6. For any graph satisfying (η,γ)-local separation property, the number of vertex-disjoint paths of length at most γ between any two non-neighbors is bounded above by η, by appealing to Menger's theorem for bounded path lengths (Lovász et al., 1978). However, the property of local paths that we describe above is a stronger notion than having sparse local separators and we consider all distinct paths of length at most γ and not just vertex disjoint paths in the formulation.
7. Two cycles are said to overlap if they have common vertices.

The girth condition can be weakened to allow for a small number of short cycles, while not allowing for typical node neighborhoods to contain short cycles. Such graphs are termed as *locally tree-like*. For instance, the ensemble of Erdős-Rényi graphs $\mathcal{G}_{\mathrm{ER}}(p, c/p)$, where an edge between any node pair appears with a probability c/p, independent of other node pairs, is locally tree-like. The parameter c may grow with p, albeit at a controlled rate for tractable structure learning. We make this more precise in Example 3 in Section 3.1. The proof of the following result may be found in Anandkumar et al. (2012a).

Proposition 3 (Random Graphs are Locally Tree-Like) *The ensemble of Erdős-Rényi graphs* $\mathcal{G}_{\mathrm{ER}}(p, c/p)$ *satisfies the* (η, γ)*-local separation property in* (4) *with*

$$\eta = 2,\ \gamma \leq \frac{\log p}{4 \log c}. \tag{5}$$

Thus, there are at most two paths of length smaller than γ between any two nodes in Erdős-Rényi graphs a.a.s, or equivalently, there are no overlapping cycles of length smaller than 2γ a.a.s. Similar observations apply for the more general *scale-free* or *power-law* graphs (Chung and Lu, 2006; Dommers et al., 2010). Along similar lines, the ensemble of Δ-random regular graphs, denoted by $\mathcal{G}_{\mathrm{Reg}}(p, \Delta)$, which is the uniform ensemble of regular graphs with degree Δ has no overlapping cycles of length at most $\Theta(\log_{\Delta-1} p)$ a.a.s. (McKay et al., 2004, Lemma 1).

2.2.3 EXAMPLE 3: SMALL-WORLD GRAPHS

The previous two examples showed local separation holds under two different conditions: bounded maximum degree and bounded number of local paths. The former class of graphs can have short cycles but the maximum degree needs to be constant, while the latter class of graphs can have a large maximum degree but the number of overlapping short cycles needs to be small. We now provide instances which incorporate both these features: large degrees and short cycles, and yet satisfy the local separation property.

The class of hybrid graphs or augmented graphs (Chung and Lu, 2006, Ch. 12) consists of graphs which are the union of two graphs: a "local" graph having short cycles and a "global" graph having small average distances. Since the hybrid graph is the union of these local and global graphs, it has both large degrees and short cycles. The simplest model $\mathcal{G}_{\mathrm{Watts}}(p, d, c/p)$, first studied by Watts and Strogatz (1998), consists of the union of a d-dimensional grid and an Erdős-Rényi random graph with parameter c. It is easily seen that a.e. graph $G \sim \mathcal{G}_{\mathrm{Watts}}(p, d, c/p)$ satisfies (η, γ)-local separation property in (4), with

$$\eta = d + 2,\ \gamma \leq \frac{\log p}{4 \log c}.$$

Similar observations apply for more general hybrid graphs studied in Chung and Lu (2006, Ch. 12).

Thus, we see that a wide range of graphs satisfy the property of having sparse local separators, and that it is possible for graphs with large degrees as well as many short cycles to have this property.

2.2.4 COUNTER-EXAMPLE: DENSE GRAPHS

While the above examples illustrate that a large class of graphs satisfy the local separation criterion, there indeed exist graphs which do not satisfy it. Such graphs tend to be "dense", that is, the

number of edges scales super-linearly in the number of nodes. For instance, the Erdős-Rényi graphs $\mathcal{G}_{\mathrm{ER}}(p, c/p)$ in the dense regime, where the average degree scales as $c = \Omega(p^2)$. In this regime, the node degrees as well as the number of short cycles grow with p. However, there is no simple decomposition into a local and a global graph with desirable properties, as in the previous example of small world graphs. Thus, the size of the local separators also grows with p in this case. Such graphs are hard instances for our framework.

3. Guarantees for Conditional Covariance Thresholding

We now characterize conditions under which the underlying Markov structure can be recovered successfully under conditional covariance thresholding.

3.1 Assumptions

(A1) *Sample Scaling Requirements:* We consider the asymptotic setting where both the number of variables (nodes) p and the number of samples n tend to infinity. We assume that the parameters $(n, p, J_{\min})$ scale in the following fashion:[8]

$$n = \Omega(J_{\min}^{-2} \log p). \tag{6}$$

We require that the number of nodes $p \to \infty$ to exploit the local separation properties of the class of graphs under consideration.

(A2) α-*Walk-summability:* The Gaussian graphical model Markov on $G_p \sim \mathcal{G}(p)$ is α-walk summable a.a.s., that is,

$$\|\overline{\mathbf{R}}_{G_p}\| \leq \alpha < 1, \quad \text{a.e. } G_p \sim \mathcal{G}(p), \tag{7}$$

where α is a constant (i.e., not a function of p), $\overline{\mathbf{R}} := [|R(i,j)|]$ is the entry-wise absolute value of the partial correlation matrix $\mathbf{R}$ and $\|\cdot\|$ denotes the spectral norm.

(A3) *Local-Separation Property:* We assume that the ensemble of graphs $\mathcal{G}(p; \eta, \gamma)$ satisfies the (η, γ)-local separation property with η, γ satisfying:

$$\eta = O(1), \; J_{\min} D_{\min}^{-1} \alpha^{-\gamma} = \omega(1), \tag{8}$$

where α is given by (7) and $D_{\min} := \min_i J(i,i)$ is the minimum diagonal entry of the potential matrix $\mathbf{J}$.

(A4) *Condition on Edge-Potentials:* The minimum absolute edge potential of an α-walk summable Gaussian graphical model satisfies

$$D_{\min}(1-\alpha) \min_{(i,j) \in G_p} \frac{J(i,j)}{K(i,j)} > 1 + \delta, \tag{9}$$

for almost every $G_p \sim \mathcal{G}(p)$, for some $\delta > 0$ (not depending on p) and[9]

$$K(i,j) := \|\mathbf{J}(V \setminus \{i,j\}, \{i,j\})\|^2,$$

8. The notations $\omega(\cdot)$, $\Omega(\cdot)$ refer to asymptotics as the number of variables $p \to \infty$.
9. Here and in the sequel, for $A, B \subset V$, we use the notation $\mathbf{J}(A,B)$ to denote the sub-matrix of $\mathbf{J}$ indexed by rows in A and columns in B.

is the spectral norm of the submatrix of the potential matrix $\mathbf{J}$, and $D_{\min} := \min_i J(i,i)$ is the minimum diagonal entry of $\mathbf{J}$. Intuitively, (9) limits the extent of non-homogeneity in the model and the extent of overlap of neighborhoods. Moreover, this assumption is not required for consistent graphical model selection when the model is attractive ($J_{i,j} \leq 0$ for $i \neq j$).[10]

(A5) *Choice of threshold* $\xi_{n,p}$*:* The threshold $\xi_{n,p}$ for graph estimation under CMIT algorithm is chosen as a function of the number of nodes p, the number of samples n, and the minimum edge potential $J_{\min}$ as follows:

$$\xi_{n,p} = O(J_{\min}),\ \xi_{n,p} = \omega\left(\frac{\alpha^\gamma}{D_{\min}}\right),\ \xi_{n,p} = \Omega\left(\sqrt{\frac{\log p}{n}}\right), \tag{10}$$

where α is given by (7), $D_{\min} := \min_i J(i,i)$ is the minimum diagonal entry of the potential matrix $\mathbf{J}$, and γ is the path-threshold (4) for the (η,γ)-local separation property to hold.

Assumption (A1) stipulates how n, p and $J_{\min}$ should scale for consistent graphical model selection, that is, the sample complexity. The sample size n needs to be sufficiently large with respect to the number of variables p in the model for consistent structure reconstruction. Assumptions (A2) and (A4) impose constraints on the model parameters. Assumption (A3) restricts the class of graphs under consideration. To the best of our knowledge, all previous works dealing with graphical model selection, for example, Meinshausen and Bühlmann (2006), Ravikumar et al. (2011), also impose some conditions for consistent graphical model selection. Assumption (A5) is with regard to the choice of a suitable threshold $\xi_{n,p}$ for thresholding conditional covariances. In the sequel, we compare the conditions for consistent recovery after presenting our main theorem.

3.1.1 EXAMPLE 1: DEGREE-BOUNDED ENSEMBLES

To gain a better understanding of conditions (A1)–(A5), consider the ensemble of graphs $\mathcal{G}_{\mathrm{Deg}}(p;\Delta)$ with bounded degree $\Delta \in \mathbb{N}$. It can be established that for the walk-summability condition in (A2) to hold,[11] we require that for normalized precision matrices ($J(i,i) = 1$),

$$J_{\max} = O\left(\frac{1}{\Delta}\right).$$

See Section A.2 for detailed discussion. When the minimum potential achieves the bound ($J_{\min} = \Theta(1/\Delta)$), a sufficient condition for (A3) to hold is given by

$$\Delta\alpha^\gamma = o(1), \tag{11}$$

where γ is the path threshold for the local-separation property to hold according to Definition 2. Intuitively, we require a larger path threshold γ, as the degree bound Δ on the graph ensemble increases.

Note that (11) allows for the degree bound Δ to grow with the number of nodes as long as the path threshold γ also grows appropriately. For example, if the maximum degree scales as $\Delta = O(\mathrm{poly}(\log p))$ and the path-threshold scales as $\gamma = O(\log\log p)$, then (11) is satisfied. This implies that graphs with fairly large degrees and short cycles can be recovered successfully using our algorithm.

10. The assumption (A5) rules out the possibility that the neighbors are marginally independent. See Section B.3 for details.

11. We can provide improved bounds for random-graph ensembles. See Section A.2 for details.

3.1.2 EXAMPLE 2: GIRTH-BOUNDED ENSEMBLES

The condition in (11) can be specialized for the ensemble of girth-bounded graphs $\mathcal{G}_{\text{Girth}}(p;g)$ in a straightforward manner as

$$\Delta\alpha^{\frac{g}{2}} = o(1), \tag{12}$$

where g corresponds to the *girth* of the graphs in the ensemble. The condition in (12) demonstrates a natural tradeoff between the girth and the maximum degree; graphs with large degrees can be learned efficiently if their girths are large. Indeed, in the extreme case of trees which have infinite girth, in accordance with (12), there is no constraint on node degrees for successful recovery and recall that the Chow-Liu algorithm (Chow and Liu, 1968) is an efficient method for model selection on tree distributions.

3.1.3 EXAMPLE 3: ERDŐS-RÉNYI AND SMALL-WORLD ENSEMBLES

We can also conclude that a.e. Erdős-Rényi graph $G \sim \mathcal{G}_{\text{ER}}(p, c/p)$ satisfies (8) when $c = O(\text{poly}(\log p))$ under the best-possible scaling of $J_{\min}$ subject to the walk-summability constraint in (7) (i.e., $J_{\min}$ achieves the upper bound).

This is because it can be shown that $J_{\min} = O(1/\sqrt{\Delta})$ for walk-summability in (7) to hold. See Section A.2 for details. Noting that a.a.s., the maximum degree Δ for $G \sim \mathcal{G}_{\text{ER}}(p, c/p)$ satisfies

$$\Delta = O\left(\frac{\log p \log c}{\log\log p}\right),$$

from Bollobás (1985, Ex. 3.6) and $\gamma = O(\frac{\log p}{\log c})$ from (5). Thus, the Erdős-Rényi graphs are amenable to successful recovery when the average degree $c = O(\text{poly}(\log p))$. Similarly, for the small-world ensemble $\mathcal{G}_{\text{Watts}}(p, d, c/p)$, when $d = O(1)$ and $c = O(\text{poly}(\log p))$, the graphs are amenable for consistent estimation.

3.2 Consistency of Conditional Covariance Thresholding

Assuming (A1)–(A5), we now state our main result. The proof of this result and the auxiliary lemmata for the proof can be found in Sections B and Section C.

Theorem 4 (Structural consistency of CMIT) *For structure learning of Gaussian graphical models Markov on a graph $G_p \sim \mathcal{G}(p;\eta,\gamma)$, $\mathsf{CMIT}(\mathbf{x}^n;\xi_{n,p},\eta)$ is consistent for a.e. graph G_p. In other words,*

$$\lim_{\substack{n,p\to\infty \\ n=\Omega(J_{\min}^{-2}\log p)}} P\left[\mathsf{CMIT}\left(\{\mathbf{x}^n\};\xi_{n,p},\eta\right) \neq G_p\right] = 0$$

Remarks:

1. *Consistency guarantee:* The CMIT algorithm consistently recovers the structure of Gaussian graphical models asymptotically, with probability tending to one, where the probability measure is with respect to both the random graph (drawn from the ensemble $\mathcal{G}(p;\eta,\gamma)$ and the samples (drawn from $\prod_{i=1}^{n} f(\mathbf{x}_i|G)$).

2. *Analysis of sample complexity:* The above result states that the sample complexity for the CMIT ($n = \Omega(J_{\min}^{-2} \log p)$), which improves when the minimum edge potential $J_{\min}$ is large.[12] This is intuitive since the edges have stronger potentials in this case. On the other hand, $J_{\min}$ cannot be arbitrarily large since the α-walk-summability assumption in (7) imposes an upper bound on $J_{\min}$. The minimum sample complexity (over different parameter settings) is attained when $J_{\min}$ achieves this upper bound. See Section A.2 for details. For example, for any degree-bounded graph ensemble $\mathcal{G}(p,\Delta)$ with maximum degree Δ, the minimum sample complexity is $n = \Omega(\Delta^2 \log p)$, that is, when $J_{\min} = \Theta(1/\Delta)$, while for Erdős-Rényi random graphs, the minimum sample complexity can be improved to $n = \Omega(\Delta \log p)$, that is, when $J_{\min} = \Theta(1/\sqrt{\Delta})$.

3. *Comparison with Ravikumar et al. (2011):* The work by Ravikumar et al. (2011) employs an ℓ_1-penalized likelihood estimator for structure estimation in Gaussian graphical models. Under the so-called incoherence conditions, the sample complexity is $n = \Omega((\Delta^2 + J_{\min}^{-2}) \log p)$. Our sample complexity in (6) is the same in terms of its dependence on $J_{\min}$, and there is no explicit dependence on the maximum degree Δ. Moreover, we have a transparent sufficient condition in terms of α-walk-summability in (7), which directly imposes scaling conditions on $J_{\min}$. It is an open question if the models satisfying incoherence conditions are walk-summable or viceversa. However, for random graph models, we can obtain better guarantees in terms of average degrees while the incoherence conditions are based on maximum degree in the graph. We also present experimental comparison between this method and our developed method in Section 6.

4. *Comparison with Meinshausen and Bühlmann (2006):* The work by Meinshausen and Bühlmann (2006) considers ℓ_1-penalized linear regression for neighborhood selection of Gaussian graphical models and establish a sample complexity of $n = \Omega((\Delta + J_{\min}^{-2}) \log p)$. We note that our guarantees allow for graphs which do not necessarily satisfy the conditions imposed by Meinshausen and Bühlmann (2006). For instance, the assumption of neighborhood stability (assumption 6 in Meinshausen and Bühlmann, 2006) is hard to verify in general, and the relaxation of this assumption corresponds to the class of models with diagonally-dominant covariance matrices. Note that the class of Gaussian graphical models with diagonally-dominant covariance matrices forms a strict sub-class of walk-summable models, and thus satisfies assumption (A2) for the theorem to hold. Thus, Theorem 4 applies to a larger class of Gaussian graphical models compared to Meinshausen and Bühlmann (2006). Furthermore, the conditions for successful recovery in Theorem 4 are arguably more transparent.

5. *Local vs. Global Conditions for Success:* The conditions required for the success of our methods as well as the ℓ_1 penalized MLE of Ravikumar et al. (2011) are global, meaning that the entire model (i.e., all the parameters) need to satisfy the specified conditions for recovering the entire graph. It does not appear straightforward to characterize local conditions for successful recovery under our formulation, that is, when our algorithm may succeed in recovering some parts of the graph, but not others. On the other hand, the ℓ_1 penalized neighborhood selection method of Meinshausen and Bühlmann (2006) provides a separate

12. Note that the sample complexity also implicitly depends on walk-summability parameter α through (8).

set of conditions for recovery of each neighborhood in the graph. However, as discussed above, these conditions appear stronger and more opaque than our conditions.

6. *Comparison with Ising models:* Our above result for learning Gaussian graphical models is analogous to structure estimation of Ising models subject to an upper bound on the edge potentials (Anandkumar et al., 2012b), and we characterize such a regime as a *conditional uniqueness* regime. Thus, walk-summability is the analogous condition for Gaussian models.

Proof Outline: We first analyze the scenario when exact statistics are available. (i) We establish that for any two non-neighbors $(i, j) \notin G$, the minimum conditional covariance in (1) (based on exact statistics) does not exceed the threshold $\xi_{n,p}$. (ii) Similarly, we also establish that the conditional covariance in (1) exceeds the threshold $\xi_{n,p}$ for all neighbors $(i, j) \in G$. (iii) We then extend these results to empirical versions using concentration bounds.

3.2.1 PERFORMANCE OF CONDITIONAL MUTUAL INFORMATION TEST

We now employ the conditional mutual information test, analyzed in Anandkumar et al. (2012b) for Ising models, and note that it has slightly worse sample complexity than using conditional covariances. Using the threshold $\xi_{n,p}$ defined in (10), the conditional mutual information test CMIT is given by the threshold test

$$\min_{\substack{S \subset V \setminus \{i,j\} \\ |S| \leq \eta}} \widehat{I}(X_i; X_j | \mathbf{X}_S) > \xi_{n,p}^2,$$

and node pairs (i, j) exceeding the threshold are added to the estimate $\widehat{G}_p^n$. Assuming (A1)–(A5), we have the following result.

Theorem 5 (Structural consistency of CMIT) *For structure learning of the Gaussian graphical model on a graph $G_p \sim \mathcal{G}(p; \eta, \gamma)$, $\mathsf{CMIT}(\mathbf{x}^n; \xi_{n,p}, \eta)$ is consistent for a.e. graph G_p. In other words,*

$$\lim_{\substack{n,p \to \infty \\ n = \Omega(J_{\min}^{-4} \log p)}} P\left[\mathsf{CMIT}\left(\{\mathbf{x}^n\}; \xi_{n,p}, \eta\right) \neq G_p\right] = 0$$

The proof of this theorem is provided in Section C.3.

Remarks:

1. For Gaussian random variables, conditional covariances and conditional mutual information are equivalent tests for conditional independence. However, from above results, we note that there is a difference in the sample complexity for the two tests. The sample complexity of CMIT is $n = \Omega(J_{\min}^{-4} \log p)$ in contrast to $n = \Omega(J_{\min}^{-2} \log p)$ for CMIT. This is due to faster decay of conditional mutual information on the edges compared to the decay of conditional covariances. Thus, conditional covariances are more efficient for Gaussian graphical model selection compared to conditional mutual information.

4. Necessary Conditions for Model Selection

In the previous sections, we proposed and analyzed efficient algorithms for learning the structure of Gaussian graphical models Markov on graph ensembles satisfying local-separation property. In this section, we study the problem of deriving *necessary* conditions for consistent structure learning.

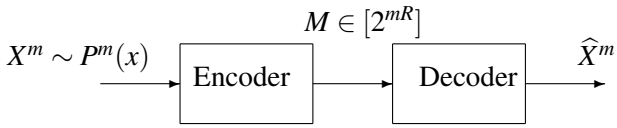


Figure 2: The canonical source coding problem. See Chapter 3 in Cover and Thomas (2006).

For the class of degree-bounded graphs $\mathcal{G}_{\mathrm{Deg}}(p,\Delta)$, necessary conditions on sample complexity have been characterized before (Wang et al., 2010) by considering a certain (limited) set of ensembles. However, a naïve application of such bounds (based on Fano's inequality (Cover and Thomas, 2006, Ch. 2)) turns out to be too weak for the class of Erdős-Rényi graphs $\mathcal{G}_{\mathrm{ER}}(p,c/p)$, where the average degree[13] c is much smaller than the maximum degree.

We now provide necessary conditions on the sample complexity for recovery of Erdős-Rényi graphs. Our information-theoretic techniques may also be applicable to other ensembles of random graphs. This is a promising avenue for future work.

4.1 Setup

We now describe the problem more formally. A graph G is drawn from the ensemble of Erdős-Rényi graphs $G \sim \mathcal{G}_{\mathrm{ER}}(p,c/p)$. The learner is also provided with n conditionally i.i.d. samples $\mathbf{X}^n := (\mathbf{X}_1,\ldots,\mathbf{X}_n) \in (\mathcal{X}^p)^n$ (where $\mathcal{X}=\mathbb{R}$) drawn from the conditional (Gaussian) product probability density function (pdf) $\prod_{i=1}^n f(\mathbf{x}_i|G)$. The task is then to estimate G, a random quantity. The estimate is denoted as $\widehat{G} := \widehat{G}(\mathbf{X}^n)$. It is desired to derive tight necessary conditions on n (as a function of c and p) so that the *probability of error*

$$P_e^{(p)} := P(\widehat{G} \neq G) \to 0 \tag{13}$$

as the number of nodes p tends to infinity. Note that the probability measure P in (13) is associated to *both* the realization of the random graph G and the samples $\mathbf{X}^n$.

The task is reminiscent of source coding (or compression), a problem of central importance in information theory (Cover and Thomas, 2006)—we would like to derive fundamental limits associated to the problem of reconstructing the source G given a compressed version of it $\mathbf{X}^n$ ($\mathbf{X}^n$ is also analogous to the "message"). However, note the important distinction; while in source coding, the source coder can design both the encoder *and* the decoder, our problem mandates that the code is fixed by the conditional probability density $f(\mathbf{x}|G)$. We are only allowed to design the decoder. See comparisons in Figs. 2 and 3.

4.2 Necessary Conditions for Exact Recovery

To derive the necessary condition for learning Gaussian graphical models Markov on sparse Erdős-Rényi graphs $G \sim \mathcal{G}_{\mathrm{ER}}(p,c/p)$, we assume that the strict walk-summability condition with parameter α, according to (7). We are then able to demonstrate the following:

Theorem 6 (Weak Converse for Gaussian Models) *For a walk-summable Gaussian graphical model satisfying* (7) *with parameter* α*, for almost every graph* $G \sim \mathcal{G}_{\mathrm{ER}}(p,c/p)$ *as* $p \to \infty$*, in order*

13. The techniques in this section are applicable when the average degree (c) of $\mathcal{G}_{\mathrm{ER}}(p,c/p)$ ensemble is a function of p, for example, $c = O(\mathrm{poly}(\log p))$.

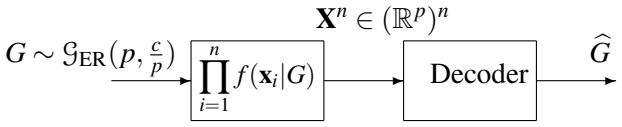


Figure 3: The estimation problem is analogous to source coding: the "source" is $G \sim \mathcal{G}_{\mathrm{ER}}(p, \frac{c}{p})$, the "message" is $\mathbf{X}^n \in (\mathbb{R}^p)^n$ and the "decoded source" is $\widehat{G}$. We are asking what the minimum "rate" (analogous to the number of samples n) are required so that $\widehat{G} = G$ with high probability.

for $P_e^{(p)} \to 0$, we require that

$$n \geq \frac{2}{p \log_2 \left[2\pi e \left(\frac{1}{1-\alpha}+1\right)\right]} \binom{p}{2} H_{\mathrm{b}}\left(\frac{c}{p}\right) \tag{14}$$

for all p sufficiently large.

The proof is provided in Section D.1. By expanding the binary entropy function, it is easy to see that the statement in (14) can be weakened to the necessary condition:

$$n \geq \frac{c \log_2 p}{\log_2 \left[2\pi e \left(\frac{1}{1-\alpha}+1\right)\right]}.$$

The above condition does not involve any asymptotic notation, and also demonstrates the dependence of the sample complexity on p, c and α transparently. Finally, the dependence on α can be explained as follows: any α-walk-summable model is also β-walk-summable for all $\beta > \alpha$. Thus, the class of β-walk-summable models contains the class of α-walk-summable models. This results in a looser bound in (14) for larger α.

4.3 Necessary Conditions for Recovery with Distortion

In this section, we generalize Theorem 6 to the case where we only require estimation of the underlying graph up to a certain edit distance: an error is declared if and only if the estimated graph $\widehat{G}$ exceeds an edit distance (or distortion) D of the true graph. The *edit distance* $d : \mathfrak{G}_p \times \mathfrak{G}_p \to \mathbb{N} \cup \{0\}$ between two undirected graphs $G = (V, E)$ and $G = (V, E')$ is defined as $d(G, G') := |E \triangle E'|$, where $\triangle$ denotes the symmetric difference between the edge sets E and E'. The edit distance can be regarded as a distortion measure between two graphs.

Given an positive integer D, known as the *distortion*, suppose we declare an error if and only if $d(G, G') > D$, then the probability of error is redefined as

$$P_e^{(p)} := P(d(G, \widehat{G}(\mathbf{X}^n)) > D). \tag{15}$$

We derive necessary conditions on n (as a function of p and c) such that the probability of error (15) goes to zero as $p \to \infty$. To ease notation, we define the ratio

$$\beta := D / \binom{p}{2}. \tag{16}$$

Note that β may be a function of p. We do not attempt to make this dependence explicit. The following corollary is based on an idea propounded by Kim et al. (2008) among others.

Corollary 7 (Weak Converse for Discrete Models With Distortion) *For $P_e^{(p)} \to 0$, we must have*

$$n \geq \frac{2}{p \log_2 \left[2\pi e \left(\frac{1}{1-\alpha}+1\right)\right]} \binom{p}{2} \left[H_{\mathrm{b}}\left(\frac{c}{p}\right) - H_{\mathrm{b}}(\beta)\right] \tag{17}$$

for all p sufficiently large.

The proof of this corollary is provided in Section D.7. Note that for (17) to be a useful bound, we need $\beta < c/p$ which translates to an allowed distortion $D < cp/2$. We observe from (17) that because the error criterion has been relaxed, the required number of samples is also reduced from the corresponding lower bound in (14).

4.4 Proof Techniques

Our analysis tools for deriving necessary conditions for Gaussian graphical model selection are information-theoretic in nature. A common and natural tool to derive necessary conditions (also called converses) is to resort to Fano's inequality (Cover and Thomas, 2006, Chapter 2), which (lower) bounds the probability of error $P_e^{(p)}$ as a function of the *equivocation* or *conditional entropy* $H(G|\mathbf{X}^n)$ and the size of the set of all graphs with p nodes. However, a direct and naïve application Fano's inequality results in a trivial lower bound as the set of all graphs, which can be realized by $\mathcal{G}_{\mathrm{ER}}(p, c/p)$ is, loosely speaking, "too large".

To ameliorate such a problem, we employ another information-theoretic notion, known as *typicality*. A *typical set* is, roughly speaking, a set that has small cardinality and yet has high probability as $p \to \infty$. For example, the probability of a set of length-m sequences is of the order $\approx 2^{mH}$ (where H is the entropy rate of the source) and hence those sequences with probability close to this value are called *typical*. In our context, given a graph G, we define the $\bar{d}(G)$ to be the ratio of the number of edges of G to the total number of nodes p. Let $\mathfrak{G}_p$ denote the set of all graphs with p nodes. For a fixed $\varepsilon > 0$, we define the following set of graphs:

$$\mathcal{T}_\varepsilon^{(p)} := \left\{ G \in \mathfrak{G}_p : \left| \frac{\bar{d}(G)}{c} - \frac{1}{2} \right| \leq \frac{\varepsilon}{2} \right\}.$$

The set $\mathcal{T}_\varepsilon^{(p)}$ is known as the ε-*typical set of graphs*. Every graph $G \in \mathcal{T}_\varepsilon^{(p)}$ has an average number of edges that is $\frac{c}{2}\varepsilon$-close in the Erdős-Rényi ensemble. Note that typicality ideas are usually used to derive *sufficient* conditions in information theory (Cover and Thomas, 2006) (*achievability* in information-theoretic parlance); our use of *both* typicality for graphical model selection as well as Fano's inequality to derive converse statements seems novel. Indeed, the proof of the converse of the source coding theorem in Cover and Thomas (2006, Chapter 3) uses only Fano's inequality. We now summarize the properties of the typical set.

Lemma 8 (Properties of $\mathcal{T}_\varepsilon^{(p)}$) *The ε-typical set of graphs has the following properties:*

1. *$P(\mathcal{T}_\varepsilon^{(p)}) \to 1$ as $p \to \infty$.*
2. *For all $G \in \mathcal{T}_\varepsilon^{(p)}$, we have*[14]

$$\exp_2\left[-\binom{p}{2} H_{\mathrm{b}}\left(\frac{c}{p}\right)(1+\varepsilon)\right] \leq P(G) \leq \exp_2\left[-\binom{p}{2} H_{\mathrm{b}}\left(\frac{c}{p}\right)\right].$$

14. We use the notation $\exp_2(\cdot)$ to mean $2^{(\cdot)}$.

3. *The cardinality of the* ε*-typical set can be bounded as*

$$(1-\varepsilon)\exp_2\left[\binom{p}{2}H_{\mathrm{b}}\left(\frac{c}{p}\right)\right] \leq |\mathcal{T}_{\varepsilon}^{(p)}| \leq \exp_2\left[\binom{p}{2}H_{\mathrm{b}}\left(\frac{c}{p}\right)(1+\varepsilon)\right]$$

for all p *sufficiently large.*

The proof of this lemma can be found in Section D.2. Parts 1 and 3 of Lemma 8 respectively say that the set of typical graphs has high probability and has very small cardinality relative to the number of graphs with p nodes $|\mathfrak{G}_p| = \exp_2(\binom{p}{2})$. Part 2 of Lemma 8 is known as the *asymptotic equipartition property*: the graphs in the typical set are almost uniformly distributed.

5. Implications on Loopy Belief Propagation

An active area of research in the graphical model community is that of inference—that is, the task of computing node marginals (or MAP estimates) through efficient distributed algorithms. The simplest of these algorithms is the belief propagation[15] (BP) algorithm, where messages are passed among the neighbors of the graph of the model. It is known that belief propagation (and max-product) is exact on tree models, meaning that correct marginals are computed at all the nodes (Pearl, 1988). On the other hand on general graphs, the generalized version of BP, known as loopy belief propagation (LBP), may not converge and even if it does, the marginals may not be correct. Motivated by the twin problems of convergence and correctness, there has been extensive work on characterizing LBP's performance for different models. As a by-product of our previous analysis on graphical model selection, we now show the asymptotic correctness of LBP on walk-summable Gaussian models when the underlying graph is locally tree-like.

5.1 Background

The belief propagation (BP) algorithm is a distributed algorithm where messages (or beliefs) are passed among the neighbors to draw inferences at the nodes of a graphical model. The computation of node marginals through naïve variable elimination (or Gaussian elimination in the Gaussian setting) is prohibitively expensive. However, if the graph is sparse (consists of few edges), the computation of node marginals may be sped up dramatically by exploiting the graph structure and using distributed algorithms to parallelize the computations.

For the sake of completeness, we now recall the basic steps in LBP, specific to Gaussian graphical models. Given a message schedule which specifies how messages are exchanged, each node j receives information from each of its neighbors (according to the graph), where the message, $m_{i\to j}^t(x_j)$, from i to j, in t^{th} iteration is parameterized as

$$m_{i\to j}^t(x_j) := \exp\left[-\frac{1}{2}\Delta J_{i\to j}^t x_j^2 + \Delta h_{i\to j}^t x_j\right].$$

Each node i prepares message $m_{i\to j}^t(x_j)$ by collecting messages from neighbors of the previous iteration (under parallel iterations), and computing

$$\hat{J}_{i\setminus j}(t) = J(i,i) + \sum_{k\in\mathcal{N}(i)\setminus j}\Delta J_{k\to i}^{t-1}, \quad \hat{h}_{i\setminus j}(t) = h(i) + \sum_{k\in\mathcal{N}(i)\setminus j}\Delta h_{k\to i}(t),$$

15. The variant of the belief propagation algorithm which computes the MAP estimates is known as the max-product algorithm.

where

$$\Delta J^t_{i\to j} = -J(j,i)\hat{J}^{-1}_{i\setminus j}(t)J(j,i), \quad \Delta h^t_{i\to j} = -J(j,i)\hat{J}^{-1}_{i\setminus j}(t)\hat{h}_{k\to i}(t).$$

5.2 Results

Let $\Sigma_{\mathrm{LBP}}(i,i)$ denote the variance at node i at the LBP fixed point.[16] Without loss of generality, we consider the normalized version of the precision matrix

$$\mathbf{J} = \mathbf{I} - \mathbf{R},$$

which can always be obtained from a general precision matrix via normalization. We can then renormalize the variances, computed via LBP, to obtain the variances corresponding to the unnormalized precision matrix.

We consider the following ensemble of locally-tree like graphs. Consider the event that the neighborhood of a node i has no cycles up to graph distance γ, given by

$$\Gamma(i;\gamma,G) := \{B_\gamma(i;G) \text{ does not contain any cycles}\}.$$

We assume a random graph ensemble $\mathcal{G}(p)$ such that for a given node $i \in V$, we have

$$P[\Gamma^c(i;\gamma,G)] = o(1). \tag{18}$$

Proposition 9 (Correctness of LBP) *Given an α-walk-summable Gaussian graphical model on a.e. locally tree-like graph $G \sim \mathcal{G}(p;\gamma)$ with parameter γ satisfying* (18)*, we have*

$$|\Sigma_G(i,i) - \Sigma_{\mathrm{LBP}}(i,i)| \overset{a.a.s.}{=} O(\max(\alpha^\gamma, P[\Gamma^c(i;\gamma,G)])).$$

The proof is given in Section B.4.

Remarks:

1. The class of Erdős-Rényi random graphs, $G \sim \mathcal{G}_{\mathrm{ER}}(p,c/p)$ satisfies (18), with $\gamma = O(\log p/\log c)$ for a node $i \in V$ chosen uniformly at random.
2. Recall that the class of random regular graphs $G \sim \mathcal{G}_{\mathrm{Reg}}(p,\Delta)$ have a girth of $O(\log_{\Delta-1} p)$. Thus, for any node $i \in V$, (18) holds with $\gamma = O(\log_{\Delta-1} p)$.

6. Experiments

In this section we present some experimental results on real and synthetic data. We implement the proposed CMIT method as well the convex relaxation methods, namely, the ℓ_1 penalized maximum likelihood estimate (MLE) (Ravikumar et al., 2011) and ℓ_1 penalized neighborhood selection (Meinshausen and Bühlmann, 2006). We measure the performance of methods using the notion of the edit distance between the true and estimated graphs (for synthetic data). We also compare the penalized likelihood scores of the estimated models using the notion of Bayesian information criterion (BIC) for both synthetic and real data. We implement the proposed CMIT method in MATLAB and the convex relaxation methods using the YALMIP package.[17] We also used CONTEST[18] to generate the synthetic graphs. The data sets, software code and results are available at `http://newport.eecs.uci.edu/anandkumar`.

16. Convergence of LBP on walk-summable models has been established by Malioutov et al. (2006).
17. YALMIP is available at `http://users.isy.liu.se/johanl/yalmip/pmwiki.php?n=Main.Download`.
18. CONTEST is at `http://www.maths.strath.ac.uk/research/groups/numerical_analysis/contest`.

6.1 Data Sets

Synthetic data: In order to evaluate the performance of our algorithm in terms of error in reconstructing the graph structure, we generated samples from Gaussian graphical models for different graphs. These include a single cycle graph with $p = 80$ nodes, an Erdős-Rényi (ER) graph $G \sim \mathcal{G}_{\mathrm{ER}}(p, c/p)$ with average degree $c = 1.2$ and Watts-Strogatz model $\mathcal{G}_{\mathrm{Watts}}(p, d, c/p)$ with degree of local graph $d = 2$ and average degree of the global graph $c = 1.2$. Given the graph structure G, we generate the potential matrix $\mathbf{J}_G$ whose sparsity pattern corresponds to that of G. We set the diagonal elements in $\mathbf{J}_G$ to unity and nonzero off-diagonal entries are picked uniformly[19] from $[0, 0.1]$. We set the potential vector $\mathbf{h}$ to be $\mathbf{0}$ without loss of generality. We let the number of samples be $n \in \{10^2, 5\times 10^2, 10^3, 5\times 10^3, 10^4\}$. Note that for synthetic data, we know η, the size of local separators for non-neighboring node pairs in the graph, and we incorporate it in the implementation of the CMIT algorithm. We present edit distance results for CMIT and the above mentioned convex relaxation methods for different thresholds and regularization parameters.[20]

Foreign exchange data: We consider monthly trends of foreign exchange rates[21] of 19 currencies[22] with respect to the US dollar from 10/1/1983 to 02/1/2012. We evaluate the BIC score for models estimated using our algorithm under different thresholds $\xi_{n,p}$ and different sizes of the local separator sets η, and compared it with the convex relaxation methods under different regularization parameters.

6.2 Performance Criteria

The BIC score has been extensively used to enable tradeoff between data fitting and model complexity (Schwarz, 1978). We use a modified version of the BIC score proposed for high-dimensional data sets (Foygel and Drton, 2010) as the performance criterion for model fitting. Given n samples $\mathbf{x}^n := [\mathbf{x}_1, \ldots, \mathbf{x}_n]$, and parameters θ,

$$\mathrm{BIC}(\mathbf{x}^n; \theta) := \sum_{k=1}^{n} \log f(\mathbf{x}_k; \theta) - 0.5|E| \log n - 2|E| \log p,$$

where $|E|$ is the number of edges in the Markov graph and $\boldsymbol{\theta}$ is the set of parameters characterizing the model. It has been observed elsewhere (Liu et al., 2009) that the BIC score tends to overselect the edges leading to dense graphs, and thus, we impose a hard threshold on the number of edges (both for our method and for convex relaxation methods), and select the model with the best BIC score. For the foreign exchange data, we limited the number of edges to 100, while for synthetic data, we limited it to 100 for cycle and Erdős-Rényi (ER) graphs and to 200 for the Watts-Strogatz model. We note that alternatively, the thresholds/regularization parameters can be selected via cross validation or other mechanisms, see Liu et al. (2009) for details.

19. The choice of parameters and graphs result in valid models in our experiments, that is, the potential matrix is positive definite.
20. For the convex relaxation methods in Ravikumar et al. (2011) and Meinshausen and Bühlmann (2006), the regularization parameter denotes the weight associated with the ℓ_1 term.
21. Data set available at `http://research.stlouisfed.org/fred2/categories/15/downloaddata`.
22. The European countries which switched to Euro are not considered in our analysis.

Graph	n	CMIT	ℓ_1 MLE	ℓ_1 Nbd
Cycle	10^2	1.0000	1.0000	1.0000
ER	10^2	1.0000	1.0000	1.0000
WS	10^2	1.0000	1.0000	1.0000
Cycle	10^3	0.95	0.9875	0.9000
ER	10^3	0.6825	1.1087	1.0000
WS	10^3	0.8580	0.9520	0.8063
Cycle	10^4	0.4125	0.3875	0.3625
ER	10^4	0.3273	0.3469	0.5435
WS	10^4	0.3252	0.3313	0.2688

Table 1: Normalized edit distance under CMIT, ℓ_1 penalized MLE and ℓ_1 penalized neighborhood selection on synthetic data from graphs listed above, where n denotes the number of samples.

6.3 Experimental Outcomes

Synthetic data: We compare the performance of our method CMIT with convex relaxation methods for synthetic data as described earlier. We evaluate the normalized edit distance (normalized with respect to the number of edges), since we know the ground truth for synthetic data and present the results in Table 1 for CMIT, ℓ_1 penalized MLE and ℓ_1 penalized neighborhood selection. methods. The results are also presented in figures 7a, 7b and 7c. We note that CMIT has better edit distance performance and BIC scores compared to ℓ_1 penalized MLE in most cases, and similar performance compared to the ℓ_1 penalized neighborhood selection.

Foreign exchange data: We evaluate the BIC scores under our algorithm CMIT with different values of η (the constraint on the size of subsets used for conditioning)[23] and threshold $\xi_{n,p}$. We present the results in Table 2, where for each value of η, we present the threshold $\xi_{n,p}$ which achieves the best BIC score under the sparsity constraint. We also present the regularization parameters for convex relaxation methods with the best BIC. The estimated graphs are shown in figures 4 and 5. We note that while CMIT distributes the edges fairly uniformly across the nodes, the ℓ_1 penalized MLE tends to cluster all the edges together between the "dominant" variables leading to a densely connected component and several isolated nodes. We observe from the reconstructed graphs that geography plays a crucial role in the foreign exchange trends. In Fig.4 recovered using the CMIT method, we note that among Asian countries India and S. Korea are high degree nodes and are connected to countries which are geographically close (e.g., Sri Lanka for India, and Australia, Thailand, Taiwan and China for S. Korea). On the other hand, the ℓ_1 method outputs a densely connected graph where such geographical relationships are missing. Thus, we see that in the experiments, the proposed CMIT method tends to enforce local sparsity in the graph, while the ℓ_1 method of Ravikumar et al. (2011) enforces global sparsity, and tends to cluster the edges together. On the other hand, the ℓ_1 penalized neighborhood selection (Meinshausen and Bühlmann, 2006) is better than the MLE in distributing the edges across all the nodes, but carries this out to a lesser extent than our method.

23. The BIC score for $\eta = 0$ is too low and we do not present it in our results. This implies that there is dependence between the variables.

Thres.(CMIT)	η	LL-train$\times 10^7$	LL-test$\times 10^7$	BIC-train$\times 10^7$	BIC-test$\times 10^7$	$\lvert E\rvert$
0.5	1	-2.9521	-7.4441	-2.9522	-7.4442	23
0.5	2	-3.2541	-8.5923	-3.2541	-8.5923	8
0.01	3	-2.9669	-7.3773	-2.9670	-7.3774	19
0.001	**4**	**-2.9653**	**-7.3674**	**-2.9654**	**-7.3675**	**25**
0.0005	5	-3.2901	-8.8396	-3.3068	-8.8397	24
0.0005	6	-3.2921	-8.8466	-3.2921	-8.8467	18
Thres.(ℓ_1 MLE)	–	LL-train$\times 10^7$	LL-test$\times 10^7$	BIC-train$\times 10^7$	BIC-test$\times 10^7$	$\lvert E\rvert$
6.5803	–	-2.5831	-6.3167	-2.5832	-6.3167	28
Thres.(ℓ_1 Nbd)	–	LL-train$\times 10^7$	LL-test$\times 10^7$	BIC-train$\times 10^7$	BIC-test$\times 10^7$	$\lvert E\rvert$
13.1606	–	-2.7971	-6.9630	-2.7972	-6.9631	26

Table 2: Experimental outcome for CMIT, ℓ_1 penalized MLE and ℓ_1 penalized neighborhood selection for different thresholds/regularization parameters and size of conditioning sets η for foreign exchange data. $|E|$ denotes the number of edges.

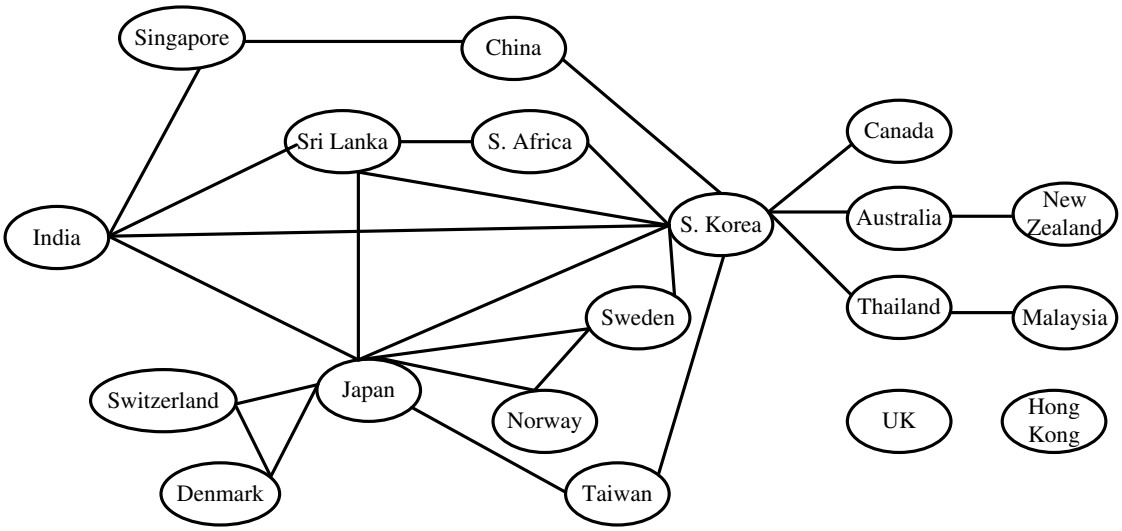


Figure 4: Graph estimate under CMIT algorithm for foreign exchange data set for $\eta = 4$, see Table 2.

7. Conclusion

In this paper, we adopted a novel and a unified paradigm for graphical model selection. We presented a simple local algorithm for structure estimation with low computational and sample complexities under a set of mild and transparent conditions. This algorithm succeeds on a wide range of graph ensembles such as the Erdős-Rényi ensemble, small-world networks etc. We also employed novel information-theoretic techniques for establishing necessary conditions for graphical model selection.

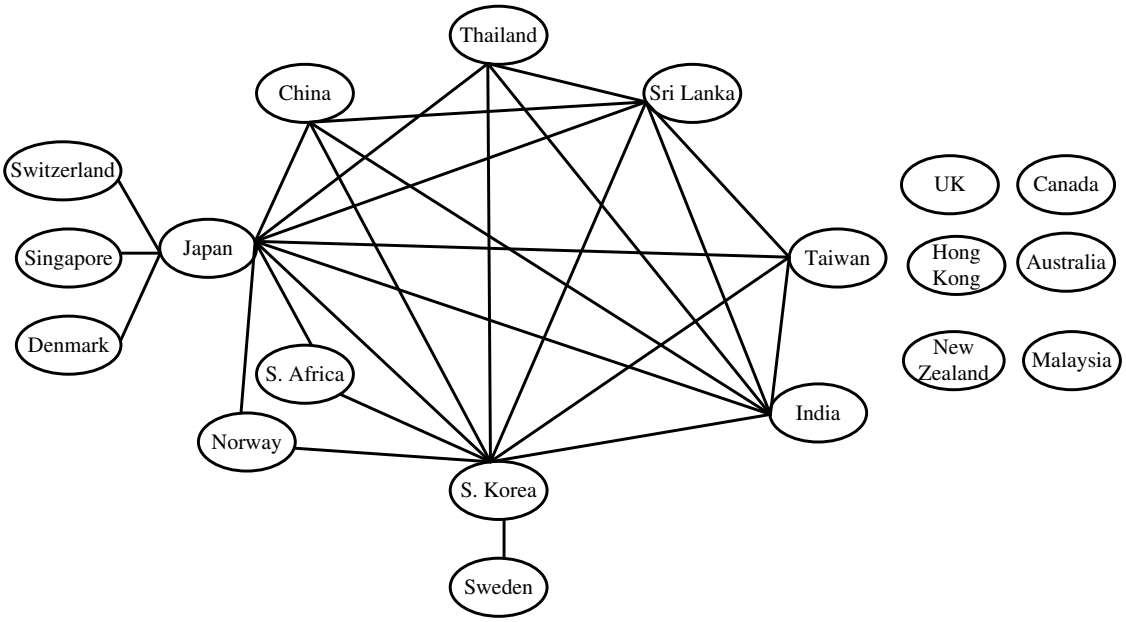


Figure 5: Graph estimate under ℓ_1 penalized MLE for foreign exchange data set. See Table 2.

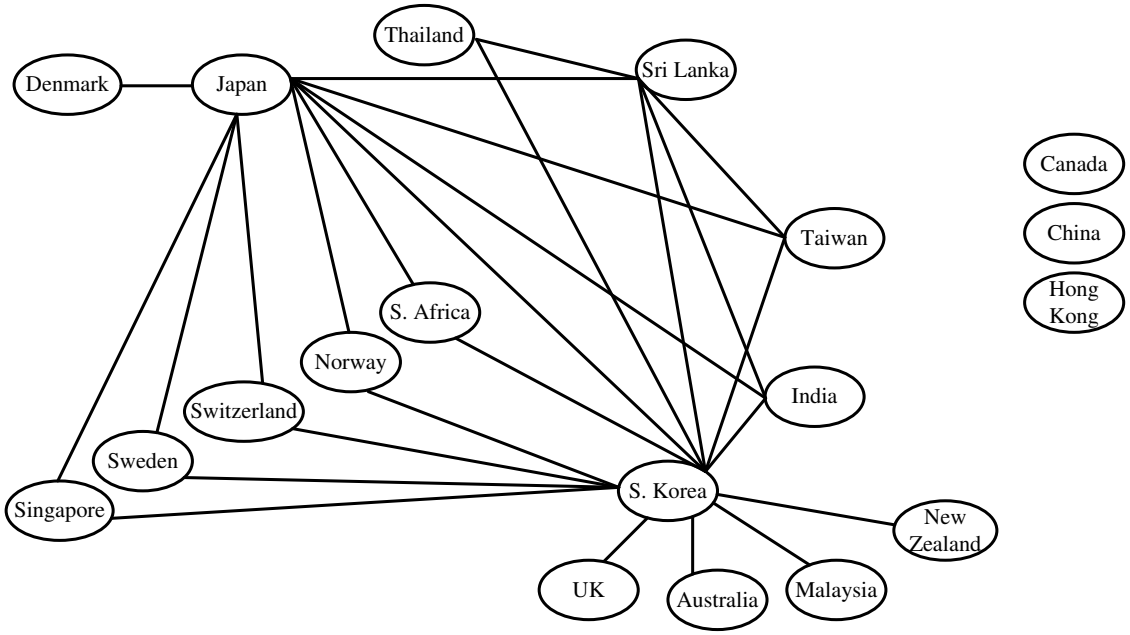
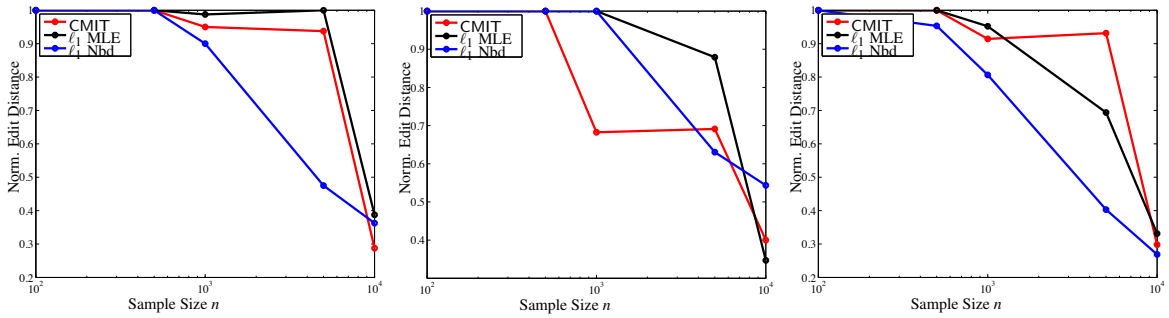


Figure 6: Graph estimate under ℓ_1 penalized neighborhood selection method for foreign exchange data set. See Table 2.



(a) Cycle (b) Erdös-Rényi (c) Watts-Strogatz

Figure 7: CMIT, ℓ_1 penalized MLE and ℓ_1 penalized neighborhood selection methods.

Acknowledgments

An abridged version of this paper appeared in the Proceedings of NIPS 2011. The first author is supported in part by the setup funds at UCI and the AFOSR Award FA9550-10-1-0310, the second author is supported by A*STAR, Singapore and the third author is supported in part by AFOSR under Grant FA9550-08-1-1080. The authors thank Venkat Chandrasekaran (UC Berkeley) for discussions on walk-summable models, Elchanan Mossel (UC Berkeley) for discussions on the necessary conditions for model selection and Divyanshu Vats (U. Minn.) for extensive comments. The authors thank the Associate Editor Martin Wainwright (Berkeley) and the anonymous reviewers for comments which significantly improved this manuscript.

Appendix A. Walk-summable Gaussian Graphical Models

We first provide an overview of the notion of walk-summability for Gaussian graphical models.

A.1 Background on Walk-Summability

We now recap the properties of walk-summable Gaussian graphical models, as given by (7). For details, see Malioutov et al. (2006). For simplicity, we first assume that the diagonal of the potential matrix $\mathbf{J}$ is normalized ($J(i,i) = 1$ for all $i \in V$). We remove this assumption and consider general unnormalized precision matrices in Section B.2. Consider splitting the matrix $\mathbf{J}$ into the identity matrix and the partial correlation matrix $\mathbf{R}$, defined in (3):

$$\mathbf{J} = \mathbf{I} - \mathbf{R}. \tag{19}$$

The covariance matrix $\mathbf{\Sigma}$ of the graphical model in (19) can be decomposed as

$$\mathbf{\Sigma} = \mathbf{J}^{-1} = (\mathbf{I} - \mathbf{R})^{-1} = \sum_{k=0}^{\infty} \mathbf{R}^k, \quad \|\mathbf{R}\| < 1, \tag{20}$$

using Neumann power series for the matrix inverse. Note that we require that $\|\mathbf{R}\| < 1$ for (20) to hold, which is implied by walk-summability in (7) (since $\|\mathbf{R}\| \leq \|\overline{\mathbf{R}}\|$).

We now relate the matrix power $\mathbf{R}^l$ to walks on graph G. A walk $\mathbf{w}$ of length $l \geq 0$ on graph G is a sequence of nodes $\mathbf{w} := (w_0, w_1, \ldots, w_l)$ traversed on the graph G, that is, $(w_k, w_{k+1}) \in G$. Let $|\mathbf{w}|$

denote the length of the walk. Given matrix $\mathbf{R}_G$ supported on graph G, let the weight of the walk be

$$\phi(\mathbf{w}) := \prod_{k=1}^{|\mathbf{w}|} R(w_{k-1}, w_k).$$

The elements of the matrix power $\mathbf{R}^l$ are given by

$$R^l(i,j) = \sum_{\mathbf{w}: i \xrightarrow{l} j} \phi(\mathbf{w}), \tag{21}$$

where $i \xrightarrow{l} j$ denotes the set of walks from i to j of length l. For this reason, we henceforth refer to $\mathbf{R}$ as the *walk matrix*.

Let $i \to j$ denote all the walks between i and j. Under the walk-summability condition in (7), we have convergence of $\sum_{\mathbf{w}: i \to j} \phi(\mathbf{w})$, irrespective of the order in which the walks are collected, and this is equal to the covariance $\Sigma(i,j)$.

In Section A.3, we relate walk-summability in (7) to the notion of correlation decay, where the effect of faraway nodes on covariances can be controlled and the local-separation property of the graphs under consideration can be exploited.

A.2 Sufficient Conditions for Walk-summability

We now provide sufficient conditions and suitable parameterization for walk-summability in (7) to hold. The adjacency matrix $\mathbf{A}_G$ of a graph G with maximum degree Δ_G satisfies

$$\lambda_{\max}(\mathbf{A}_G) \leq \Delta_G,$$

since it is dominated by a Δ-regular graph which has maximum eigenvalue of Δ_G. From Perron-Frobenius theorem, for adjacency matrix $\mathbf{A}_G$, we have $\lambda_{\max}(\mathbf{A}_G) = \|\mathbf{A}_G\|$, where $\|\mathbf{A}_G\|$ is the spectral radius of $\mathbf{A}_G$. Thus, for $\overline{\mathbf{R}}_G$ supported on graph G, we have

$$\alpha := \|\overline{\mathbf{R}}_G\| = O\left(J_{\max}\Delta\right),$$

where $J_{\max} := \max_{i,j} |R(i,j)|$. This implies that

$$J_{\max} = O\left(\frac{1}{\Delta}\right)$$

to have $\alpha < 1$, which is the requirement for walk-summability.

When the graph G is a Erdős-Rényi random graph, $G \sim \mathcal{G}_{\mathrm{ER}}(p, c/p)$, we can provide better bounds. When $G \sim \mathcal{G}_{\mathrm{ER}}(p, c/p)$, we have Krivelevich and Sudakov (2003), that

$$\lambda_{\max}(\mathbf{A}_G) = (1 + o(1)) \max(\sqrt{\Delta_G}, c),$$

where Δ_G is the maximum degree and $\mathbf{A}_G$ is the adjacency matrix. Thus, in this case, when $c = O(1)$, we require that

$$J_{\max} = O\left(\sqrt{\frac{1}{\Delta}}\right),$$

for walk-summability ($\alpha < 1$). Note that when $c = O(\mathrm{poly}(\log p))$, w.h.p. $\Delta_{G_p} = \Theta(\log p / \log\log p)$ (Bollobás, 1985, Ex. 3.6).

A.3 Implications of Walk-Summability

Recall that $\mathbf{\Sigma}_G$ denotes the covariance matrix for Gaussian graphical model on graph G and that $\mathbf{J}_G = \mathbf{\Sigma}_G^{-1}$ with $\mathbf{J}_G = \mathbf{I} - \mathbf{R}_G$ in (19). We now relate the walk-summability condition in (7) to correlation decay in the model. In other words, under walk-summability, we can show that the effect of faraway nodes on covariances decays with distance, as made precise in Lemma 10.

Let $B_\gamma(i)$ denote the set of nodes within γ hops from node i in graph G. Denote $H_{\gamma;ij} := G(B_\gamma(i) \cap B_\gamma(j))$ as the induced subgraph of G over the intersection of γ-hop neighborhoods at i and j and retaining the nodes in $V \setminus \{B_\gamma(i) \cap B_\gamma(j)\}$. Thus, $H_{\gamma;ij}$ has the same number of nodes as G. We first make the following simple observation: the (i,j) element in the γ^{th} power of walk matrix, $R_G^\gamma(i,j)$, is given by walks of length γ between i and j on graph G and thus, depends only on subgraph[24] $H_{\gamma;ij}$, see (21). This enables us to quantify the effect of nodes outside $B_\gamma(i) \cap B_\gamma(j)$ on the covariance $\Sigma_G(i,j)$.

Define a new walk matrix $\mathbf{R}_{H_{\gamma;ij}}$ such that

$$R_{H_{\gamma;ij}}(a,b) = \begin{cases} R_G(a,b), & a,b \in B_\gamma(i) \cap B_\gamma(j), \\ 0, & \text{o.w.} \end{cases}$$

In other words, $\mathbf{R}_{H_{\gamma;ij}}$ is formed by considering the Gaussian graphical model over graph $H_{\gamma;ij}$. Let $\mathbf{\Sigma}_{H_{\gamma;ij}}$ denote the corresponding covariance matrix.[25]

Lemma 10 (Covariance Bounds Under Walk-summability) *For any walk-summable Gaussian graphical model* ($\alpha := \|\overline{\mathbf{R}}_G\| < 1$)*, we have*[26]

$$\max_{i,j} |\Sigma_G(i,j) - \Sigma_{H_{\gamma;ij}}(i,j)| \leq \alpha^\gamma \frac{2\alpha}{1-\alpha} = O(\alpha^\gamma). \tag{22}$$

Thus, for walk-summable Gaussian graphical models, we have $\alpha := \|\overline{\mathbf{R}}_G\| < 1$, implying that the error in (22) in approximating the covariance by local neighborhood decays exponentially with distance. Parts of the proof below are inspired by Dumitriu and Pal (2009).

Proof: Using the power-series in (20), we can write the covariance matrix as

$$\mathbf{\Sigma}_G = \sum_{k=0}^{\gamma} \mathbf{R}_G^k + \mathbf{E}_G,$$

where the error matrix $\mathbf{E}_G$ has spectral radius

$$\|\mathbf{E}_G\| \leq \frac{\|\mathbf{R}_G\|^{\gamma+1}}{1 - \|\mathbf{R}_G\|},$$

from (20). Thus,[27] for any $i, j \in V$,

$$|\Sigma_G(i,j) - \sum_{k=0}^{\gamma} R_G^k(i,j)| \leq \frac{\|\mathbf{R}_G\|^{\gamma+1}}{1 - \|\mathbf{R}_G\|}. \tag{23}$$

24. Note that $R^\gamma(i,j) = 0$ if $B_\gamma(i) \cap B_\gamma(j) = \emptyset$.
25. When $B_\gamma(i) \cap B_\gamma(j) = \emptyset$ meaning that graph distance between i and j is more than γ, we obtain $\mathbf{\Sigma}_{H_{\gamma;ij}} = \mathbf{I}$.
26. The bound in (22) also holds if $H_{\gamma;ij}$ is replaced with any of its supergraphs.
27. For any matrix $\mathbf{A}$, we have $\max_{i,j} |A(i,j)| \leq \|\mathbf{A}\|$.

Similarly, we have

$$|\Sigma_{H_{\gamma;ij}}(i,j) - \sum_{k=0}^{\gamma} R^k_{H_{\gamma;ij}}(i,j)| \leq \frac{\|\mathbf{R}_{H_{\gamma;ij}}\|^{\gamma+1}}{1-\|\mathbf{R}_{H_{\gamma;ij}}\|} \stackrel{(a)}{\leq} \frac{\|\overline{\mathbf{R}}_G\|^{\gamma+1}}{1-\|\overline{\mathbf{R}}_G\|}, \tag{24}$$

where for inequality (a), we use the fact that

$$\|\mathbf{R}_{H_{\gamma;ij}}\| \leq \|\overline{\mathbf{R}}_{H_{\gamma;ij}}\| \leq \|\overline{\mathbf{R}}_G\|,$$

since $H_{\gamma;ij}$ is a subgraph[28] of G.

Combining (23) and (24), using the triangle inequality, we obtain (22). □

We also make some simple observations about conditional covariances in walk-summable models. Recall that $\overline{\mathbf{R}}_G$ denotes matrix with absolute values of $\mathbf{R}_G$, and $\mathbf{R}_G$ is the walk matrix over graph G. Also recall that the α-walk summability condition in (7), is $\|\overline{\mathbf{R}}_G\| \leq \alpha < 1$.

Proposition 11 (Conditional Covariances under Walk-Summability) *Given a walk-summable Gaussian graphical model, for any $i, j \in V$ and $S \subset V$ with $i, j \notin S$, we have*

$$\Sigma(i,j|S) = \sum_{\substack{\mathbf{w}: i \to j \\ \forall k \in \mathbf{w}, k \notin S}} \phi_G(\mathbf{w}). \tag{25}$$

Moreover, we have

$$\sup_{\substack{i \in V \\ S \subset V \setminus i}} \Sigma(i,i|S) \leq (1-\alpha)^{-1} = O(1). \tag{26}$$

Proof: We have, from Rue and Held (2005, Thm. 2.5),

$$\Sigma(i,j|S) = J^{-1}_{-S,-S;G}(i,j),$$

where $\mathbf{J}_{-S,-S;G}$ denotes the submatrix of potential matrix $\mathbf{J}_G$ by deleting nodes in S. Since submatrix of a walk-summable matrix is walk-summable, we have (25) by appealing to the walk-sum expression for conditional covariances.

For (26), let $\|\mathbf{A}\|_\infty$ denote the maximum absolute value of entries in matrix $\mathbf{A}$. Using monotonicity of spectral norm and the fact that $\|\mathbf{A}\|_\infty \leq \|\mathbf{A}\|$, we have

$$\sup_{\substack{i \in V \\ S \subset V, i \notin V}} \Sigma(i,i|S) \leq \|\mathbf{J}^{-1}_{-S,-S;G}\| = (1-\|\mathbf{R}_{-S,-S;G}\|)^{-1} \leq (1-\|\overline{\mathbf{R}}_{-S,-S;G}\|)^{-1} \leq (1-\|\overline{\mathbf{R}}_G\|)^{-1} = O(1).$$

□

Thus, the conditional covariance in (25) consists of walks in the original graph G, not passing through nodes in S.

28. When two matrices $\mathbf{A}$ and $\mathbf{B}$ are such that $|A(i,j)| \geq |B(i,j)|$ for all i, j, we have $\|\mathbf{A}\| \geq \|\mathbf{B}\|$.

Appendix B. Graphs with Local-Separation Property

We now provide bounds on conditional covariance for walk-summable matrices.

B.1 Conditional Covariance between Non-Neighbors: Normalized Case

We now provide bounds on the conditional covariance for Gaussian graphical models Markov on a graph $G \sim \mathcal{G}(p;\eta,\gamma)$ satisfying the local-separation property (η,γ), as per Definition 2.

Lemma 12 (Conditional Covariance Between Non-neighbors) *For a walk-summable Gaussian graphical model, the conditional covariance between non-neighbors i and j, conditioned on S_γ, the γ-local separator between i and j, satisfies*

$$\max_{j \notin \mathcal{N}(i)} \Sigma(i;j|S_\gamma) = O(\|\overline{\mathbf{R}}_G\|^\gamma).$$

Proof: In this proof, we abbreviate S_γ by S for notational convenience. The conditional covariance is given by the Schur complement, that is, for any subset A such that $A \cap S = \emptyset$,

$$\Sigma(A|S) = \Sigma(A,A) - \Sigma(A,S)\Sigma(S,S)^{-1}\Sigma(S,A). \tag{27}$$

We use the notation $\mathbf{\Sigma}_G(A,A)$ to denote the submatrix of the covariance matrix $\mathbf{\Sigma}_G$, when the underlying graph is G. As in Lemma 10, we may decompose $\mathbf{\Sigma}_G$ as follows:

$$\mathbf{\Sigma}_G = \mathbf{\Sigma}_{H_\gamma} + \mathbf{E}_\gamma,$$

where H_γ is the subgraph spanned by γ-hop neighborhood $B_\gamma(i)$, and $\mathbf{E}_\gamma$ is the error matrix. Let $\mathbf{F}_\gamma$ be the matrix such that

$$\mathbf{\Sigma}_G(S,S)^{-1} = \mathbf{\Sigma}_{H_\gamma}(S,S)^{-1} + \mathbf{F}_\gamma.$$

We have $\Sigma_{H_\gamma}(i,j|S) = 0$, where $\Sigma_{H_\gamma}(i,j|S)$ denotes the conditional covariance by considering the model given by the subgraph H_γ. This is due to the Markov property since i and j are separated by S in the subgraph H_γ.

Thus using (27), the conditional covariance on graph G can be bounded as

$$\Sigma_G(i,j|S) = O(\max(\|\mathbf{E}_\gamma\|, \|\mathbf{F}_\gamma\|)).$$

By Lemma 10, we have $\|\mathbf{E}_\gamma\| = O(\|\overline{\mathbf{R}}_G\|^\gamma)$. Using Woodbury matrix-inversion identity, we also have $\|\mathbf{F}_\gamma\| = O(\|\overline{\mathbf{R}}_G\|^\gamma)$. □

B.2 Extension to General Precision Matrices: Unnormalized Case

We now extend the above analysis to general precision matrices $\mathbf{J}$ where the diagonal elements are not assumed to be identity. Denote the precision matrix as

$$\mathbf{J} = \mathbf{D} - \mathbf{E},$$

where $\mathbf{D}$ is a diagonal matrix and $\mathbf{E}$ has zero diagonal elements. We thus have that

$$\mathbf{J}_{\text{norm}} := \mathbf{D}^{-0.5}\mathbf{J}\mathbf{D}^{-0.5} = \mathbf{I} - \mathbf{R},$$

where $\mathbf{R}$ is the partial correlation matrix. This also implies that

$$\mathbf{J} = \mathbf{D}^{0.5}\mathbf{J}_{\text{norm}}\mathbf{D}^{0.5}.$$

Thus, we have that

$$\mathbf{\Sigma} = \mathbf{D}^{-0.5}\mathbf{\Sigma}_{\text{norm}}\mathbf{D}^{-0.5}, \tag{28}$$

where $\mathbf{\Sigma}_{\text{norm}} := \mathbf{J}_{\text{norm}}^{-1}$ is the covariance matrix corresponding to the normalized model. When the model is walk-summable, that is, $\|\overline{\mathbf{R}}\| \leq \alpha < 1$, we have that $\mathbf{\Sigma}_{\text{norm}} = \sum_{k\geq 0}\mathbf{R}^k$.

We now use the results derived in the previous sections involving the normalized model (Lemma 10 and Lemma 12) to obtain bounds for general precision matrices.

Lemma 13 (Covariance Bounds for General Models) *For any walk-summable Gaussian graphical model* ($\alpha := \|\overline{\mathbf{R}}_G\| < 1$)*, we have the following results:*

1. Covariance Bounds: *The covariance entries upon limiting to a subgraph $H_{\gamma,ij}$ for any $i, j \in V$ satisfies*

$$\max_{i,j} |\Sigma_G(i,j) - \Sigma_{H_{\gamma,ij}}(i,j)| \leq \frac{\alpha^\gamma}{D_{\min}}\frac{2\alpha}{1-\alpha} = O\left(\frac{\alpha^\gamma}{D_{\min}}\right), \tag{29}$$

   where $D_{\min} := \min_i D(i,i) = \min_i J(i,i)$.

2. Conditional Covariance between Non-neighbors: *The conditional covariance between non-neighbors i and j, conditioned on S_γ, the γ-local separator between i and j, satisfies*

$$\max_{j\notin\mathcal{N}(i)} \Sigma(i;j|S_\gamma) = O\left(\frac{\alpha^\gamma}{D_{\min}}\right), \tag{30}$$

   where $D_{\min} := \min_i D(i,i) = \min_i J(i,i)$.

Proof: Using (28) and Lemma 10, we have (29). Similarly, it can be shown that for any $S \subset V \setminus \{i,j\}$, $i, j \in V$,

$$\Sigma(i,j|S) = \mathbf{D}^{-0.5}\Sigma_{\text{norm}}(i,j|S)\mathbf{D}^{-0.5},$$

where $\Sigma_{\text{norm}}(i,j|S)$ is the conditional covariance corresponding to the model with normalized precision matrix. From Lemma 12, we have (30). □

B.3 Conditional Covariance between Neighbors: General Case

We provide a lower bound on conditional covariance among the neighbors for the graphs under consideration. Recall that $J_{\min}$ denotes the minimum edge potentials. Let

$$K(i,j) := \|\mathbf{J}(V\setminus\{i,j\},\{i,j\})\|^2,$$

where $\mathbf{J}(V\setminus\{i,j\},\{i,j\})$ is a sub-matrix of the potential matrix $\mathbf{J}$.

Lemma 14 (Conditional Covariance Between Neighbors) *For an α-walk summable Gaussian graphical model satisfying*

$$D_{\min}(1-\alpha)\min_{(i,j)\in G_p}\frac{J(i,j)}{K(i,j)} > 1+\delta, \tag{31}$$

for some $\delta > 0$ (not depending on p), where $D_{\min} := \min_i J(i,i)$, we have

$$|\Sigma_G(i,j|S)| = \Omega(J_{\min}),$$

for any $(i,j) \in G$ such that $j \in \mathcal{N}(i)$ and any subset $S \subset V$ with $i,j \notin S$.

Proof: First note that for attractive models,

$$\begin{aligned}\Sigma_G(i,j|S) &\overset{(a)}{\geq} \Sigma_{G_1}(i,j|S) \\ &\overset{(b)}{=} \frac{-J(i,j)}{J(i,i)J(j,j)-J(i,j)^2} = \Omega(J_{\min}),\end{aligned}$$

where G_1 is the graph consisting only of edge (i,j). Inequality (a) arises from the fact that in attractive models, the weights of all the walks are positive, and thus, the weight of walks on G_1 form a lower bound for those on G (recall that the covariances are given by the sum-weight of walks on the graphs). Equality (b) is by direct matrix inversion of the model on G_1.

For general models, we need further analysis. Let $A = \{i,j\}$ and $B = V \setminus \{S \cup A\}$, for some $S \subset V \setminus A$. Let $\mathbf{\Sigma}(A,A)$ denote the covariance matrix on set A, and let $\widetilde{\mathbf{J}}(A,A) := \mathbf{\Sigma}(A,A)^{-1}$ denote the corresponding marginal potential matrix. We have for all $S \subset V \setminus A$

$$\widetilde{\mathbf{J}}(A,A) = \mathbf{J}(A,A) - \mathbf{J}(A,B)\mathbf{J}(B,B)^{-1}\mathbf{J}(B,A).$$

Recall that $\|\mathbf{A}\|_\infty$ denotes the maximum absolute value of entries in matrix $\mathbf{A}$.

$$\begin{aligned}\|\mathbf{J}(A,B)\mathbf{J}(B,B)^{-1}\mathbf{J}(B,A)\|_\infty &\overset{(a)}{\leq} \|\mathbf{J}(A,B)\mathbf{J}(B,B)^{-1}\mathbf{J}(B,A)\| \\ &\overset{(b)}{\leq} \|\mathbf{J}(A,B)\|^2 \|\mathbf{J}(B,B)^{-1}\| \\ &= \frac{\|\mathbf{J}(A,B)\|^2}{\lambda_{\min}(\mathbf{J}(B,B))}, \\ &\overset{(c)}{\leq} \frac{K(i,j)^2}{D_{\min}(1-\alpha)}\end{aligned}$$

where inequality (a) arises from the fact that the ℓ_∞ norm is bounded by the spectral norm, (b) arises from sub-multiplicative property of norms and (c) arises from walk-summability property. Inequality (b) is from the bound on edge potentials and α-walk summability of the model and since $K(i,j) \geq \|\mathbf{J}(A,B)\|$. Assuming (31), we have

$$|\widetilde{J}(i,j)| > J_{\min} - \frac{\|\mathbf{J}(A,B)\|^2}{D_{\min}(1-\alpha)} = \Omega(J_{\min}).$$

Since

$$\Sigma_G(i,j|S) = \frac{-\widetilde{J}(i,j)}{\widetilde{J}(i,i)\widetilde{J}(j,j)-\widetilde{J}(i,j)^2},$$

we have the result. □

B.4 Analysis of Loopy Belief Propagation

Proof of Proposition 9: From Lemma 10 in Section A.3, for any α-walk-summable Gaussian graphical model, we have, for all nodes $i \in V$ conditioned on the event $\Gamma(i;\gamma,G)$,

$$|\Sigma_G(i,i) - \Sigma_{\mathrm{LBP}}(i,i)| = O(\|\overline{\mathbf{R}}_G\|^{\gamma}).$$

This is because conditioned on $\Gamma(i;\gamma,G)$, it is shown that the series expansions based on walk-sums corresponding to the variances $\Sigma_{H_{\gamma;ij}}(i,i)$ and $\Sigma_{\mathrm{LBP}}(i,i)$ are identical up to length γ walks, and the effect of walks beyond length γ can be bounded as above. Moreover, for a sequence of α-walk-summable, we have $\Sigma(i,i) \le M$ for all $i \in V$, for some constant M and similarly $\Sigma_{\mathrm{LBP}}(i,j) \le M'$ for some constant M' since it is obtained by the set of self-avoiding walks in G. We thus have

$$\mathbb{E}\left[|\Sigma_G(i,i) - \Sigma_{\mathrm{LBP}}(i,i)|\right] \le \left[O(\|\overline{\mathbf{R}}_G\|^{\gamma}) + P[\Gamma^c(i;\gamma)]\right] = o(1),$$

where $\mathbb{E}$ is over the expectation of ensemble $\mathcal{G}(p)$. By Markov's inequality,[29] we have the result. $\square$

Appendix C. Sample-based Analysis

We now extend our analysis to the setting where we have access to samples instead of exact statistics.

C.1 Concentration of Empirical Quantities

For our sample complexity analysis, we recap the concentration result by Ravikumar et al. (2011, Lemma 1) for sub-Gaussian matrices and specialize it to Gaussian matrices.

Lemma 15 (Concentration of Empirical Covariances) *For any p-dimensional Gaussian random vector $\mathbf{X} = [X_1,\ldots,X_p]$, the empirical covariance obtained from n samples satisfies*

$$P\left[|\widehat{\Sigma}(i,j) - \Sigma(i,j)| > \varepsilon\right] \le 4\exp\left[-\frac{n\varepsilon^2}{3200M^2}\right], \tag{32}$$

for all $\varepsilon \in (0, 40M)$ and $M := \max_i \Sigma(i,i)$.

This translates to bounds for empirical conditional covariance.

Corollary 16 (Concentration of Empirical Conditional Covariance) *For a walk-summable p-dimensional Gaussian random vector $\mathbf{X} = [X_1,\ldots,X_p]$, we have*

$$P\left[\max_{\substack{i\neq j\\ S\subset V,|S|\le\eta}} |\widehat{\Sigma}(i,j|S) - \Sigma(i;j|S)| > \varepsilon\right] \le 4p^{\eta+2}\exp\left(-\frac{n\varepsilon^2}{K}\right), \tag{33}$$

where $K \in (0,\infty)$ is a constant which is bounded when $\|\mathbf{\Sigma}\|_\infty$ is bounded, for all $\varepsilon \in (0, 40M)$ with $M := \max_i \Sigma(i,i)$, and $n \ge \eta$.

29. By Markov's inequality, for a non-negative random variable X, we have $P[X > \delta] \le \mathbb{E}[X]/\delta$. By choosing $\delta = \omega(\mathbb{E}[X])$, we have the result.

Proof: For a given $i, j \in V$ and $S \subset V$ with $\eta \leq n$, using (27),

$$P\left[|\widehat{\Sigma}(i,j|S) - \Sigma(i;j|S)| > \varepsilon\right] \leq \mathbb{P}\left[\left(|\widehat{\Sigma}(i,j) - \Sigma(i;j)| > \varepsilon\right) \bigcup_{k \in S} \left(|\widehat{\Sigma}(i,k) - \Sigma(i;k)| > K'\varepsilon\right)\right],$$

where K' is a constant which is bounded when $\|\mathbf{\Sigma}\|_\infty$ is bounded. Using Lemma 15, we have the result. □

C.2 Proof of Theorem 4

We are now ready to prove Theorem 4. We analyze the error events for the conditional covariance threshold test CMIT. For any $(i,j) \notin G_p$, define the event

$$\mathcal{F}_1(i,j;\{\mathbf{x}^n\},G_p) := \left\{|\widehat{\Sigma}(i,j|S)| > \xi_{n,p}\right\},$$

where $\xi_{n,p}$ is the threshold in (10) and S is the γ-local separator between i and j (since the minimum in (1) is achieved by the γ-local separator). Similarly for any edge $(i,j) \in G_p$, define the event that

$$\mathcal{F}_2(i,j;\{\mathbf{x}^n\},G_p) := \left\{\exists S \subset V : |S| \leq \eta, |\widehat{\Sigma}(i,j|S)| < \xi_{n,p}\right\}.$$

The probability of error resulting from CMIT can thus be bounded by the two types of errors,

$$\mathbb{P}[\mathsf{CMIT}(\{\mathbf{x}^n\};\xi_{n,p}) \neq G_p] \leq \mathbb{P}\left[\bigcup_{(i,j)\in G_p} \mathcal{F}_2(i,j;\{\mathbf{x}^n\},G_p)\right] + \mathbb{P}\left[\bigcup_{(i,j)\notin G_p} \mathcal{F}_1(i,j;\{\mathbf{x}^n\},G_p)\right] \tag{34}$$

For the first term, applying union bound for both the terms and using the result (33) of Lemma 15,

$$\mathbb{P}\left[\bigcup_{(i,j)\in G_p} \mathcal{F}_2(i,j;\{\mathbf{x}^n\},G_p)\right] = O\left(p^{\eta+2}\exp\left[-\frac{n(C_{\min}(p)-\xi_{n,p})^2}{K^2}\right]\right) \tag{35}$$

where

$$C_{\min}(p) := \inf_{\substack{(i,j)\in G_p \\ S\subset V, i,j\notin S \\ |S|\leq\eta}} |\Sigma(i,j|S)| = \Omega\left(J_{\min}\right), \quad \forall\, p \in \mathbb{N},$$

from (37). Since $\xi_{n,p} = o(J_{\min})$, (35) is $o(1)$ when $n > L\log p/J_{\min}^2$, for sufficiently large L (depending on η and M). For the second term in (34),

$$\mathbb{P}\left[\bigcup_{(i,j)\notin G_p} \mathcal{F}_1(i,j;\{\mathbf{x}^n\},G_p)\right] = O\left(p^{\eta+2}\exp\left[-\frac{n(\xi_{n,p}-C_{\max}(p))^2}{K^2}\right]\right), \tag{36}$$

where

$$C_{\max}(p) := \max_{(i,j)\notin G_p} |\Sigma(i,j|S)| = O\left(\frac{\alpha^\gamma}{D_{\min}}\right).$$

For the choice of $\xi_{n,p}$ in (10), (36) is $o(1)$ and this completes the proof of Theorem 4. □

C.3 Conditional Mutual Information Thresholding Test

We now analyze the performance of conditional mutual information threshold test. We first note bounds on conditional mutual information.

Proposition 17 (Conditional Mutual Information) *Under the assumptions (A1)–(A5), we have that the conditional mutual information among non-neighbors, conditioned on the γ-local separation satisfies*

$$\max_{(i,j)\notin G} I(X_i;X_j|\mathbf{X}_{S_\gamma}) = O(\alpha^{2\gamma}),$$

and the conditional mutual information among the neighbors satisfy

$$\min_{\substack{(i,j)\in G \\ S\subset V\setminus\{i,j\}}} I(X_i;X_j|\mathbf{X}_S) = \Omega(J_{\min}^2). \tag{37}$$

Proof: The conditional mutual information for Gaussian variables is given by

$$I(X_i;X_j|\mathbf{X}_S) = -\frac{1}{2}\log\left[1-\rho^2(i,j|S)\right],$$

where $\rho(i,j|S)$ is the conditional correlation coefficient, given by

$$\rho(i,j|S) := \frac{\Sigma(i,j|S)}{\sqrt{\Sigma(i,i|S)\Sigma(j,j|S)}}.$$

From (26) in Proposition 11, we have $\Sigma(i,i|S) = O(1)$ and thus, the result holds. □

We now note the concentration bounds on empirical mutual information.

Lemma 18 (Concentration of Empirical Mutual Information) *For any p-dimensional Gaussian random vector $\mathbf{X} = [X_1,\ldots,X_p]$, the empirical mutual information obtained from n samples satisfies*

$$P(|\widehat{I}(X_i;X_j) - I(X_i;X_j)| > \varepsilon) \leq 24\exp\left(-\frac{nM\varepsilon^2}{204800L^2}\right), \tag{38}$$

for some constant L which is finite when $\rho_{\max} := \max_{i\neq j}|\rho(i,j)| < 1$, and all $\varepsilon < \rho_{\max}$, and for $M := \max_i \Sigma(i,i)$.

Proof: The result on empirical covariances can be found in Ravikumar et al. (2011, Lemma 1). The result in (38) will be shown through a sequence of transformations. First, we will bound

$P(|\widehat{\rho}(i,j)-\rho(i,j)|>\varepsilon)$. Consider,

$$
\begin{aligned}
&P(|\widehat{\rho}(i,j)-\rho(i,j)|>\varepsilon)\\
&=P\left(\left|\frac{\widehat{\Sigma}(i,j)}{(\widehat{\Sigma}(i,i)\widehat{\Sigma}(j,j))^{1/2}}-\frac{\Sigma(i,j)}{(\Sigma(i,i)\Sigma(j,j))^{1/2}}\right|>\varepsilon\right)\\
&=P\left(\left|\frac{\widehat{\Sigma}(i,j)}{\Sigma(i,j)}\left(\frac{\Sigma(i,i)}{\widehat{\Sigma}(i,i)}\frac{\Sigma(j,j)}{\widehat{\Sigma}(j,j)}\right)^{1/2}-1\right|>\frac{\varepsilon}{|\rho(i,j)|}\right)\\
&\overset{(a)}{\leq}P\left(\frac{\widehat{\Sigma}(i,j)}{\Sigma(i,j)}>\left(1+\frac{\varepsilon}{|\rho(i,j)|}\right)^{1/3}\right)+P\left(\frac{\widehat{\Sigma}(i,j)}{\Sigma(i,j)}<\left(1-\frac{\varepsilon}{|\rho(i,j)|}\right)^{1/3}\right)+\dots\\
&\quad+P\left(\frac{\Sigma(i,i)}{\widehat{\Sigma}(i,i)}>\left(1+\frac{\varepsilon}{|\rho(i,j)|}\right)^{2/3}\right)+P\left(\frac{\Sigma(i,i)}{\widehat{\Sigma}(i,i)}<\left(1-\frac{\varepsilon}{|\rho(i,j)|}\right)^{2/3}\right)+\dots\\
&\quad+P\left(\frac{\Sigma(j,j)}{\widehat{\Sigma}(j,j)}>\left(1+\frac{\varepsilon}{|\rho(i,j)|}\right)^{2/3}\right)+P\left(\frac{\Sigma(j,j)}{\widehat{\Sigma}(j,j)}<\left(1-\frac{\varepsilon}{|\rho(i,j)|}\right)^{2/3}\right)\\
&\overset{(b)}{\leq}P\left(\frac{\widehat{\Sigma}(i,j)}{\Sigma(i,j)}>1+\frac{\varepsilon}{8|\rho(i,j)|}\right)+P\left(\frac{\widehat{\Sigma}(i,j)}{\Sigma(i,j)}<1-\frac{\varepsilon}{8|\rho(i,j)|}\right)+\dots\\
&\quad+P\left(\frac{\Sigma(i,i)}{\widehat{\Sigma}(i,i)}>1+\frac{\varepsilon}{3|\rho(i,j)|}\right)+P\left(\frac{\widehat{\Sigma}(i,i)}{\Sigma(i,i)}<1-\frac{\varepsilon}{3|\rho(i,j)|}\right)+\dots\\
&\quad+P\left(\frac{\widehat{\Sigma}(j,j)}{\Sigma(j,j)}>1+\frac{\varepsilon}{3|\rho(i,j)|}\right)+P\left(\frac{\widehat{\Sigma}(j,j)}{\Sigma(j,j)}<1-\frac{\varepsilon}{3|\rho(i,j)|}\right)\\
&\overset{(c)}{\leq}24\exp\left(-\frac{nM\varepsilon^2}{204800|\rho(i,j)|^2}\right)\overset{(d)}{\leq}24\exp\left(-\frac{nM\varepsilon^2}{204800}\right)
\end{aligned}
$$

where in (a), we used the fact that $P(ABC>1+\delta)\leq P(A>(1+\delta)^{1/3}$ or $B>(1+\delta)^{1/3}$ or $C>(1+\delta)^{1/3})$ and the union bound, in (b) we used the fact that $(1+\delta)^3\leq 1+8\delta$ and $(1+\delta)^{-2/3}\leq 1-\delta/3$ for $\delta=\varepsilon/|\rho(i,j)|<1$. Finally, in (c), we used the result in (32) and in (d), we used the bounds on $\rho<1$.

Now, define the bijective function $I(|\rho|):=-1/2\log(1-\rho^2)$. Then we claim that there exists a constant $L\in(0,\infty)$, depending only on $\rho_{\max}<1$, such that

$$|I(x)-I(y)|\leq L|x-y|, \tag{39}$$

that is, the function $I:[0,\rho_{\max}]\to\mathbb{R}^+$ is $L=L(\rho_{\max})$-Lipschitz. This is because the slope of the function I is bounded in the interval $[0,\rho_{\max}]$. Thus, we have the inclusion

$$\{|\widehat{I}(X_i;X_j)-I(X_i;X_j)|>\varepsilon\}\subset\{|\widehat{\rho}(i,j)-\rho(i,j)|>\varepsilon/L\} \tag{40}$$

since if $|\widehat{I}(X_i;X_j)-I(X_i;X_j)|>\varepsilon$ it is true that $L|\widehat{\rho}(i,j)-\rho(i,j)|>\varepsilon$ from (39). We have by monotonicity of measure and (40) the desired result. □

We can now obtain the desired result on concentration of empirical conditional mutual information.

Lemma 19 (Concentration of Empirical Conditional Mutual Information) *For a walk-summable p-dimensional Gaussian random vector* $\mathbf{X} = [X_1, \ldots, X_p]$*, we have*

$$P\left[\max_{\substack{i\neq j \\ S\subset V\setminus\{i,j\},|S|\leq\eta}} |\widehat{I}(X_i;X_j|\mathbf{X}_S) - I(X_i;X_j|\mathbf{X}_S)| > \varepsilon\right] \leq 24p^{\eta+2}\exp\left(-\frac{nM\varepsilon^2}{204800L^2}\right),$$

for constants $M, L \in (0,\infty)$ *and all* $\varepsilon < \rho_{\max}$*, where* $\rho_{\max} := \max_{\substack{i\neq j \\ S\subset V\setminus\{i,j\},|S|\leq\eta}} |\rho(i,j|S)|$.

Proof: Since the model is walk-summable, we have that $\max_{i,S}\Sigma(i,i|S) = O(1)$ and thus, the constant M is bounded. Similarly, due to strict positive-definiteness we have $\rho_{\max} < 1$ even as $p \to \infty$, and thus, the constant L is also finite. The result then follows from union bound. □

The sample complexity for structural consistency of CMIT follows on lines of analysis for CMIT.

Appendix D. Necessary Conditions for Model Selection

We now provide proofs for necessary conditions for model selection.

D.1 Necessary Conditions for Exact Recovery

We provide the proof of Theorem 6 in this section. We collect four auxiliary lemmata whose proofs (together with the proof of Lemma 8) will be provided at the end of the section. For information-theoretic notation, the reader is referred to Cover and Thomas (2006).

Lemma 20 (Upper Bound on Differential Entropy of Mixture) *Let* $\alpha < 1$*. Suppose asymptotically almost surely each precision matrix* $\mathbf{J}_G = \mathbf{I} - \mathbf{R}_G$ *satisfies* (7)*, that is, that* $\|\overline{\mathbf{R}}_G\| \leq \alpha$ *for a.e.* $G \in \mathcal{G}(p)$*. Then, for the Gaussian model, we have*

$$h(\mathbf{X}^n) \leq \frac{pn}{2}\log_2\left(\frac{2\pi e}{1-\alpha}\right),$$

where recall that $\mathbf{X}^n|G \sim \prod_{i=1}^n f(\mathbf{x}_i|G)$*.*

For the sake of convenience, we define the random variable:

$$W = \begin{cases} 1 & G \in \mathcal{T}_\varepsilon^{(p)} \\ 0 & G \notin \mathcal{T}_\varepsilon^{(p)} \end{cases}.$$

The random variable W indicates whether $G \in \mathcal{T}_\varepsilon^{(p)}$.

Lemma 21 (Lower Bound on Conditional Differential Entropy) *Suppose that each precision matrix* $\mathbf{J}_G$ *has unit diagonal. Then,*

$$h(\mathbf{X}^n|G,W) \geq -\frac{pn}{2}\log_2(2\pi e).$$

Lemma 22 (Conditional Fano Inequality) *In the above notation, we have*

$$\frac{H(G|\mathbf{X}^n, G \in \mathcal{T}_\varepsilon^{(p)}) - 1}{\log_2(|\mathcal{T}_\varepsilon^{(p)}| - 1)} \leq P(\widehat{G}(\mathbf{X}^n) \neq G | G \in \mathcal{T}_\varepsilon^{(p)}).$$

Lemma 23 (Exponential Decay in Probability of Atypical Set) *Define the rate function* $K(c,\varepsilon) := \frac{c}{2}[(1+\varepsilon)\ln(1+\varepsilon) - \varepsilon]$. *The probability of the* ε*-atypical set decays as*

$$P((\mathcal{T}_\varepsilon^{(p)})^c) = P(G \notin \mathcal{T}_\varepsilon^{(p)}) \leq 2\exp(-pK(c,\varepsilon)) \tag{41}$$

for all $p \geq 1$.

Note the non-asymptotic nature of the bound in (41). The rate function $K(c,\varepsilon)$ satisfies $\lim_{\varepsilon \downarrow 0} K(c,\varepsilon)/\varepsilon^2 = c/4$. We prove Theorem 6 using these lemmata.

Proof: Consider the following sequence of lower bounds:

$$\begin{aligned}
\frac{pn}{2}\log_2\left(\frac{2\pi e}{1-\alpha}\right) &\overset{(a)}{\geq} h(\mathbf{X}^n) \\
&\overset{(b)}{\geq} h(\mathbf{X}^n|W) \\
&= I(\mathbf{X}^n; G|W) + h(\mathbf{X}^n|G,W) \\
&\overset{(c)}{\geq} I(\mathbf{X}^n; G|W) - \frac{pn}{2}\log_2(2\pi e) \\
&= H(G|W) - H(G|\mathbf{X}^n, W) - \frac{pn}{2}\log_2(2\pi e),
\end{aligned} \tag{42}$$

where (a) follows from Lemma 20, (b) is because conditioning does not increase differential entropy and (c) follows from Lemma 21. We will lower bound the first term in (42) and upper bound the second term in (42). Now consider the first term in (42):

$$\begin{aligned}
H(G|W) &= H(G|W=1)P(W=1) + H(G|W=0)P(W=0) \\
&\overset{(a)}{\geq} H(G|W=1)P(W=1) \\
&\overset{(b)}{\geq} H(G|G \in \mathcal{T}_\varepsilon^{(p)})(1-\varepsilon) \\
&\overset{(c)}{\geq} (1-\varepsilon)\binom{p}{2} H_b\left(\frac{c}{p}\right),
\end{aligned} \tag{43}$$

where (a) is because the entropy $H(G|W=0)$ and the probability $P(W=0)$ are both non-negative. Inequality (b) follows for all p sufficiently large from the definition of W as well as Lemma 8 part 1. Statement (c) comes from fact that

$$\begin{aligned}
H(G|G \in \mathcal{T}_\varepsilon^{(p)}) &= -\sum_{g \in \mathcal{T}_\varepsilon^{(p)}} P(g|g \in \mathcal{T}_\varepsilon^{(p)}) \log_2 P(g|g \in \mathcal{T}_\varepsilon^{(p)}) \\
&\geq -\sum_{g \in \mathcal{T}_\varepsilon^{(p)}} P(g|g \in \mathcal{T}_\varepsilon^{(p)}) \left[-\binom{p}{2} H_b\left(\frac{c}{p}\right)\right] = \binom{p}{2} H_b\left(\frac{c}{p}\right).
\end{aligned}$$

We are now done bounding the first term in the difference in (42).

Now we will bound the second term in (42). First we will derive a bound on $H(G|\mathbf{X}^n, W=1)$. Consider,

$$\begin{aligned}
P_e^{(p)} &:= P(\widehat{G}(\mathbf{X}^n) \neq G) \\
&\stackrel{(a)}{=} P(\widehat{G}(\mathbf{X}^n) \neq G | W=1)P(W=1) + P(\widehat{G}(\mathbf{X}^n) \neq G | W=0)P(W=0) \\
&\geq P(\widehat{G}(\mathbf{X}^n) \neq G | W=1)P(W=1) \\
&\stackrel{(b)}{\geq} P(\widehat{G}(\mathbf{X}^n) \neq G | G \in \mathcal{T}_\varepsilon^{(p)}) \left(\frac{1}{1+\varepsilon}\right) \\
&\stackrel{(c)}{\geq} \frac{H(G|\mathbf{X}^n, G \in \mathcal{T}_\varepsilon^{(p)}) - 1}{\log_2 |\mathcal{T}_\varepsilon^{(p)}|} \left(\frac{1}{1+\varepsilon}\right),
\end{aligned} \tag{44}$$

where (a) is by the law of total probability, (b) holds for all p sufficiently large by Lemma 8 part 1 and (c) is due to the conditional version of Fano's inequality (Lemma 22). Then, from (44), we have

$$\begin{aligned}
H(G|\mathbf{X}^n, W=1) &\leq P_e^{(p)}(1+\varepsilon)\log_2 |\mathcal{T}_\varepsilon^{(p)}| + 1 \\
&\leq P_e^{(p)}(1+\varepsilon)\binom{p}{2} H_{\mathrm{b}}\left(\frac{c}{p}\right) + 1.
\end{aligned} \tag{45}$$

Define the *rate function* $K(c,\varepsilon) := \frac{c}{2}[(1+\varepsilon)\ln(1+\varepsilon) - \varepsilon]$. Note that this function is positive whenever $c, \varepsilon > 0$. In fact it is monotonically increasing in both parameters. Now we use (45) to bound $H(G|\mathbf{X}^n, W)$:

$$\begin{aligned}
H(G|\mathbf{X}^n, W) &= H(G|\mathbf{X}^n, W=1)P(W=1) + H(G|\mathbf{X}^n, W=0)P(W=0) \\
&\stackrel{(a)}{\leq} H(G|\mathbf{X}^n, W=1) + H(G|\mathbf{X}^n, W=0)P(W=0) \\
&\stackrel{(b)}{\leq} H(G|\mathbf{X}^n, W=1) + H(G|\mathbf{X}^n, W=0)(2e^{-pK(c,\varepsilon)}) \\
&\stackrel{(c)}{\leq} H(G|\mathbf{X}^n, W=1) + p^2(2e^{-pK(c,\varepsilon)}) \\
&\stackrel{(d)}{\leq} P_e^{(p)}(1+\varepsilon)\binom{p}{2} H_{\mathrm{b}}\left(\frac{c}{p}\right) + 1 + 2p^2 e^{-pK(c,\varepsilon)},
\end{aligned} \tag{46}$$

where (a) is because we upper bounded $P(W=1)$ by unity, (b) follows by Lemma 23, (c) follows by upper bounding the conditional entropy by p^2 and (d) follows from (45).

Substituting (43) and (46) back into (42) yields

$$\begin{aligned}
\frac{pn}{2}\log_2\left[2\pi e\left(\frac{1}{1-\alpha}+1\right)\right] &\geq (1-\varepsilon)\binom{p}{2} H_{\mathrm{b}}\left(\frac{c}{p}\right) - P_e^{(p)}(1+\varepsilon)\binom{p}{2} H_{\mathrm{b}}\left(\frac{c}{p}\right) - 1 - 2p^2 e^{-pK(c,\varepsilon)} \\
&= \binom{p}{2} H_{\mathrm{b}}\left(\frac{c}{p}\right)\left[(1-\varepsilon) - P_e^{(p)}(1+\varepsilon)\right] - \Theta(p^2 e^{-pK(c,\varepsilon)}),
\end{aligned}$$

which implies that

$$n \geq \frac{2}{p\log_2\left[2\pi e\left(\frac{1}{1-\alpha}+1\right)\right]}\binom{p}{2} H_{\mathrm{b}}\left(\frac{c}{p}\right)\left[(1-\varepsilon) - P_e^{(p)}(1+\varepsilon)\right] - \Theta(pe^{-pK(c,\varepsilon)}).$$

Note that $\Theta(pe^{-pK(c,\varepsilon)}) \to 0$ as $p \to \infty$ since the rate function $K(c,\varepsilon)$ is positive. If we impose that $P_e^{(p)} \to 0$ as $p \to \infty$, then n has to satisfy (14) by the arbitrariness of $\varepsilon > 0$. This completes the proof of Theorem 6. □

D.2 Proof of Lemma 8

Proof: Part 1 follows directed from the law of large numbers. Part 2 follows from the fact that the Binomial pmf is maximized at its mean. Hence, for $G \in \mathcal{T}_\varepsilon^{(p)}$, we have

$$P(G) \leq \left(\frac{c}{p}\right)^{cp/2} \left(1-\frac{c}{p}\right)^{\binom{p}{2}-cp/2}.$$

We arrive at the upper bound after some rudimentary algebra. The lower bound can be proved by observing that for $G \in \mathcal{T}_\varepsilon^{(p)}$, we have

$$\begin{aligned} P(G) &\geq \left(\frac{c}{p}\right)^{cp(1+\varepsilon)/2} \left(1-\frac{c}{p}\right)^{\binom{p}{2}-cp(1+\varepsilon)/2} \\ &= \exp_2\left[\binom{p}{2}(\frac{c}{p}\log_2\frac{c}{p})(1+\varepsilon) + [1-c(1+\varepsilon)/p]\log_2(1-\frac{c}{p})\right] \\ &\geq \exp_2\left[\binom{p}{2}(\frac{c}{p}\log_2\frac{c}{p})(1+\varepsilon) + (1+\varepsilon)(1-\frac{c}{p})\log_2(1-\frac{c}{p})\right]. \end{aligned}$$

The result in Part 2 follows immediately by appealing to the symmetry of the binomial pmf about its mean. Part 3 follows by the following chain of inequalities:

$$\begin{aligned} 1 = \sum_{G\in\mathfrak{G}_n} P(G) &\geq \sum_{G\in\mathcal{T}_\varepsilon^{(p)}} P(G) \geq \sum_{G\in\mathcal{T}_\varepsilon^{(p)}} \exp_2\left[-\binom{p}{2} H_{\mathrm{b}}\left(\frac{c}{p}(1+\varepsilon)\right)\right] \\ &= |\mathcal{T}_\varepsilon^{(p)}| \exp_2\left[-\binom{p}{2} H_{\mathrm{b}}\left(\frac{c}{p}\right)(1+\varepsilon)\right]. \end{aligned}$$

This completes the proof of the upper bound on $|\mathcal{T}_\varepsilon^{(p)}|$. The lower bound follows by noting that for sufficiently large n, $P(\mathcal{T}_\varepsilon^{(p)}) \geq 1-\varepsilon$ (by Lemma 8 Part 1). Thus,

$$1-\varepsilon \leq \sum_{G\in\mathcal{T}_\varepsilon^{(p)}} P(G) \leq \sum_{G\in\mathcal{T}_\varepsilon^{(p)}} \exp_2\left[-\binom{p}{2} H_{\mathrm{b}}\left(\frac{c}{p}\right)\right] = |\mathcal{T}_\varepsilon^{(p)}| \exp_2\left[-\binom{p}{2} H_{\mathrm{b}}\left(\frac{c}{p}\right)\right].$$

This completes the proof. □

D.3 Proof of Lemma 20

Proof: Note that the distribution of $\mathbf{X}$ (with G marginalized out) is a Gaussian mixture model given by $\sum_{G\in\mathfrak{G}_p} P(G)\mathcal{N}(\mathbf{0}, \mathbf{J}_G^{-1})$. As such the covariance matrix of $\mathbf{X}$ is given by

$$\mathbf{\Sigma}_{\mathbf{X}} = \sum_{G\in\mathfrak{G}_p} P(G)\mathbf{J}_G^{-1}. \tag{47}$$

This is not immediately obvious but it is due to the zero-mean nature of each Gaussian probability density function $\mathcal{N}(\mathbf{0}, \mathbf{J}_G^{-1})$. Using (47), we have the following chain of inequalities:

$$
\begin{aligned}
h(\mathbf{X}^n) &\leq nh(\mathbf{X}) \\
&\overset{(a)}{\leq} \frac{n}{2} \log_2 \left((2\pi e)^p \det(\mathbf{\Sigma_X}) \right) \\
&= \frac{n}{2} \left[p \log_2(2\pi e) + \log_2 \det(\mathbf{\Sigma_X}) \right] \\
&\overset{(b)}{\leq} \frac{n}{2} \left[p \log_2(2\pi e) + p \log_2 \lambda_{\max}(\mathbf{\Sigma_X}) \right] \\
&= \frac{n}{2} \left[p \log_2(2\pi e) + p \log_2 \lambda_{\max} \left(\sum_{G \in \mathfrak{G}_p} P(G) \mathbf{J}_G^{-1} \right) \right] \\
&\overset{(c)}{\leq} \frac{n}{2} \left[p \log_2(2\pi e) + p \log_2 \left(\sum_{G \in \mathfrak{G}_p} P(G) \lambda_{\max} \left(\mathbf{J}_G^{-1} \right) \right) \right] \\
&= \frac{n}{2} \left[p \log_2(2\pi e) + p \log_2 \left(\sum_{G \in \mathfrak{G}_p} P(G) \frac{1}{\lambda_{\min}(\mathbf{J}_G)} \right) \right] \\
&\overset{(d)}{\leq} \frac{n}{2} \left[p \log_2(2\pi e) + p \log_2 \left(\sum_{G \in \mathfrak{G}_p} P(G) \frac{1}{1-\alpha} \right) \right] \\
&= \frac{pn}{2} \log_2 \left(\frac{2\pi e}{1-\alpha} \right),
\end{aligned}
$$

where (a) uses the maximum entropy principle (Cover and Thomas, 2006, Chapter 13), that is, that the Gaussian maximizes entropy subject to an average power constraint (b) uses the fact that the determinant of $\mathbf{\Sigma_X}$ is upper bounded by $\lambda_{\max}(\mathbf{\Sigma_X})^n$, (c) uses the convexity of $\lambda_{\max}(\cdot)$ (it equals to the operator norm $\|\cdot\|_2$ over the set of symmetric matrices, (d) uses the fact that $\alpha \geq \|\overline{\mathbf{R}}_G\|_2 \geq \|\mathbf{R}_G\|_2 = \|\mathbf{I} - \mathbf{J}_G\|_2 = \lambda_{\max}(\mathbf{I} - \mathbf{J}_G) = 1 - \lambda_{\min}(\mathbf{J}_G)$ a.a.s. This completes the proof. □

D.4 Proof of Lemma 21

Proof: Firstly, we lower bound $h(\mathbf{X}^n|G,W=1)$ as follows:

$$\begin{aligned}
h(\mathbf{X}^n|G) &= \sum_{g\in\mathfrak{G}_p} P(g)h(\mathbf{X}^n|G=g) \\
&\overset{(a)}{=} n\sum_{g\in\mathfrak{G}_p} P(g)h(\mathbf{X}|G=g) \\
&\overset{(b)}{=} \frac{n}{2}\sum_{g\in\mathfrak{G}_p} P(g)\log_2[(2\pi e)^p \det(\mathbf{J}_g^{-1})] \\
&= -\frac{n}{2}\sum_{g\in\mathfrak{G}_p} P(g)\log_2[(2\pi e)^p \det(\mathbf{J}_g)] \\
&\overset{(c)}{\geq} -\frac{n}{2}\sum_{g\in\mathfrak{G}_p} P(g)\log_2[(2\pi e)^p] \\
&\geq -\frac{pn}{2}\log_2(2\pi e),
\end{aligned}$$

where (a) is because the samples in $\mathbf{X}^n$ are conditionally independent given $G=g$, (b) is by the Gaussian assumption, (c) is by Hadamard's inequality

$$\det(\mathbf{J}_g) \leq \prod_{i=1}^{p}[\mathbf{J}_g]_{ii} = 1$$

and the assumption that each diagonal element of each precision matrix $\mathbf{J}_g = \mathbf{I} - \mathbf{R}_g$ is equal to 1 a.a.s. This proves the claim. □

D.5 Proof of Lemma 22

Proof: Define the "error" random variable

$$E = \begin{cases} 1 & \widehat{G}(\mathbf{X}^n) \neq G \\ 0 & \widehat{G}(\mathbf{X}^n) = G \end{cases}.$$

Now consider

$$\begin{aligned}
H(E,G|\mathbf{X}^n,W=1) &= H(E|\mathbf{X}^n,W=1) + H(G|E,\mathbf{X}^n,W=1) &(48)\\
&= H(G|\mathbf{X}^n,W=1) + H(E|G,\mathbf{X}^n,W=1). &(49)
\end{aligned}$$

The first term in (48) can be bounded above by 1 since the alphabet of the random variable E is of size 2. Since $H(G|E=0,\mathbf{X}^n,W=1)=0$, the second term in (48) can be bounded from above as

$$\begin{aligned}
H(G|E,\mathbf{X}^n,W=1) &= H(G|E=0,\mathbf{X}^n,W=1)P(E=0|W=1) \\
&\quad + H(G|E=1,\mathbf{X}^n,W=1)P(E=1|W=1) \\
&\leq P(\widehat{G}(\mathbf{X}^n)\neq G|G\in\mathcal{T}_\varepsilon^{(p)})\log_2(|\mathcal{T}_\varepsilon^{(p)}|-1).
\end{aligned}$$

The second term in (49) is 0. Hence, we have the desired conclusion. □

D.6 Proof of Lemma 23

Proof: The proof uses standard Chernoff bounding techniques but the scaling in p is somewhat different from the usual Chernoff (Cramér) upper bound. For simplicity, we will use $M := \binom{p}{2}$. Let $Y_i, i = 1,\ldots,M$ be independent Bernoulli random variables such that $P(Y_i = 1) = c/p$. Then the probability in question can be bounded as

$$\begin{aligned} P(G \notin \mathcal{T}_{\varepsilon}^{(p)}) &= P\left(\left|\frac{1}{cp}\sum_{i=1}^{M} Y_i - \frac{1}{2}\right| > \frac{\varepsilon}{2}\right) \\ &\overset{(a)}{\leq} 2P\left(\frac{1}{cp}\sum_{i=1}^{M} Y_i > \frac{1+\varepsilon}{2}\right) \\ &\overset{(b)}{\leq} 2\mathbb{E}\left[\exp\left(t\sum_{i=1}^{M} Y_i - pt\frac{c}{2}(1+\varepsilon)\right)\right] \qquad (50) \\ &= 2\exp\left(-pt\frac{c}{2}(1+\varepsilon)\right)\prod_{i=1}^{M}\mathbb{E}[\exp(tY_i)], \qquad (51) \end{aligned}$$

where (a) follows from the union bound, (b) follows from an application of Markov's inequality with $t \geq 0$ in (50). Now, the moment generating function of a Bernoulli random variable with probability of success q is $qe^t + (1-q)$. Using this fact, we can further upper bound (51) as follows:

$$\begin{aligned} P(G \notin \mathcal{T}_{\varepsilon}^{(p)}) &= 2\exp\left(-pt\frac{c}{2}(1+\varepsilon) + M\ln(\frac{c}{p}e^t + (1-\frac{c}{p})\right) \\ &\overset{(a)}{\leq} 2\exp\left(-pt\frac{c}{2}(1+\varepsilon) + \frac{p(p-1)}{2}\frac{c}{p}(e^t-1)\right) \\ &\leq 2\exp\left(-p\left[t\frac{c}{2}(1+\varepsilon) - \frac{c}{2}(e^t-1)\right]\right), \qquad (52) \end{aligned}$$

where in (a), we used the fact that $\ln(1+z) \leq z$. Now, we differentiate the exponent in square brackets with respect to $t \geq 0$ to find the tightest bound. We observe that the optimal parameter is $t^* = \ln(1+\varepsilon)$. Substituting this back into (52) completes the proof. □

D.7 Necessary Conditions for Recovery with Distortion

We now provide the proof for Corollary 7.

The proof of Corollary 7 follows from the following generalization of the conditional Fano's inequality presented in Lemma 22. This is a modified version of an analogous theorem in Kim et al. (2008).

Lemma 24 (Conditional Fano's Inequality (Generalization)) *In the above notation, we have*

$$\frac{H(G|\mathbf{X}^n, G \in \mathcal{T}_{\varepsilon}^{(p)}) - 1 - \log_2 L}{\log_2(|\mathcal{T}_{\varepsilon}^{(p)}| - 1)} \leq P(d(G, \widehat{G}(\mathbf{X}^n)) > D | G \in \mathcal{T}_{\varepsilon}^{(p)}) \qquad (53)$$

where $L = \binom{p}{2} H_b(\beta)$ *and* β *is defined in* (16).

We will only provide a proof sketch of Lemma 24 since it is similar to Lemma 22. *Proof:* The key to establishing (53) is to upper bound the cardinality of the set $\{G \in \mathfrak{G}_p : d(G,G') \leq D\}$, which is isomorphic to $\{E \in \mathfrak{E}_p : |E \triangle E'| \leq D\}$, where $\mathfrak{E}_p$ is the set of all edge sets (with p nodes). For this purpose, we order the node pairs in a labelled undirected graph lexicographically. Now, we map each edge set E into a length-$\binom{p}{2}$ bit-string $s(E) \in \{0,1\}^{\binom{p}{2}}$. The characters in the string $s(E)$ indicate whether or not an edge is present between two node pairs. Define $d_H(s,s')$ to be the Hamming distance between strings s and s'. Then, note that

$$|E \triangle E'| = d_H(s(E), s(E')) = d_H(s(E) \oplus s(E'), 0) \tag{54}$$

where $\oplus$ denotes addition in $\mathbb{F}_2$ and 0 denotes the all zeros string. The relation in (54) means that the cardinality of the set $\{E \in \mathfrak{E}_n : |E \triangle E'| \leq D\}$ is equal to the number of strings of Hamming weight less than or equal to D. With this realization, it is easy to see that

$$|\{s \in \{0,1\}^{\binom{p}{2}} : d_H(s,0) \leq D\}| = \sum_{k=1}^{D} \binom{\binom{p}{2}}{k} \leq 2^{\binom{p}{2} H_b(D/\binom{p}{2})} = 2^L.$$

By using the same steps as in the proof of Lemma 24 (or Fano's inequality for list decoding), we arrive at the desired conclusion. □

References

P. Abbeel, D. Koller, and A.Y. Ng. Learning factor graphs in polynomial time and sample complexity. *The Journal of Machine Learning Research*, 7:1743–1788, 2006.

A. Anandkumar, A. Hassidim, and J. Kelner. Topology discovery of sparse random graphs with few participants. *Accepted to J. of Random Structures and Algorithms*, Jan. 2012a.

A. Anandkumar, V. Y. F. Tan, F. Huang, and A. S. Willsky. High-dimensional structure learning of Ising models: Local separation criterion. *Accepted to Annals of Statistics*, Jan. 2012b.

P.J. Bickel and E. Levina. Covariance regularization by thresholding. *The Annals of Statistics*, 36 (6):2577–2604, 2008.

A. Bogdanov, E. Mossel, and S. Vadhan. The complexity of distinguishing Markov random fields. *Approximation, Randomization and Combinatorial Optimization. Algorithms and Techniques*, pages 331–342, 2008.

B. Bollobás. *Random graphs*. Academic Press, 1985.

G. Bresler, E. Mossel, and A. Sly. Reconstruction of Markov random fields from samples: Some observations and algorithms. In *Intl. Workshop APPROX Approximation, Randomization and Combinatorial Optimization*, pages 343–356. Springer, 2008.

J. Cheng, R. Greiner, J. Kelly, D. Bell, and W. Liu. Learning bayesian networks from data: an information-theory based approach. *Artificial Intelligence*, 137(1-2):43–90, 2002.

M.J. Choi, V.Y.F. Tan, A. Anandkumar, and A. Willsky. Learning latent tree graphical models. *J. of Machine Learning Research*, 12:1771–1812, May 2011.

C. Chow and C. Liu. Approximating discrete probability distributions with dependence trees. *IEEE Tran. on Information Theory*, 14(3):462–467, 1968.

F.R.K. Chung. *Spectral Graph Theory*. Amer Mathematical Society, 1997.

F.R.K. Chung and L. Lu. *Complex Graphs and Network*. Amer. Mathematical Society, 2006.

T. Cover and J. Thomas. *Elements of Information Theory*. John Wiley & Sons, Inc., 2006.

A. d'Aspremont, O. Banerjee, and L. El Ghaoui. First-order methods for sparse covariance selection. *SIAM. J. Matrix Anal. & Appl.*, 30(56), 2008.

S. Dommers, C. Giardinà, and R. van der Hofstad. Ising models on power-law random graphs. *Journal of Statistical Physics*, pages 1–23, 2010.

I. Dumitriu and S. Pal. Sparse regular random graphs: Spectral density and eigenvectors. *Arxiv preprint arXiv:0910.5306*, 2009.

R. Foygel and M. Drton. Extended bayesian information criteria for gaussian graphical models. In *Proc. of NIPS*, 2010.

A. Gamburd, S. Hoory, M. Shahshahani, A. Shalev, and B. Virag. On the girth of random cayley graphs. *Random Structures & Algorithms*, 35(1):100–117, 2009.

J.Z. Huang, N. Liu, M. Pourahmadi, and L. Liu. Covariance matrix selection and estimation via penalised normal likelihood. *Biometrika*, 93(1), 2006.

M. Kalisch and P. Bühlmann. Estimating high-dimensional directed acyclic graphs with the pc-algorithm. *J. of Machine Learning Research*, 8:613–636, 2007.

D. Karger and N. Srebro. Learning Markov networks: maximum bounded tree-width graphs. In *Proc. of ACM-SIAM Symposium on Discrete Algorithms*, pages 392–401, 2001.

Y.-H. Kim, A. Sutivong, and T. M. Cover. State amplification. *IEEE Transactions on Information Theory*, 54(5):1850 – 1859, May 2008.

M. Krivelevich and B. Sudakov. The largest eigenvalue of sparse random graphs. *Combinatorics, Probability and Computing*, 12(01):61–72, 2003.

C. Lam and J. Fan. Sparsistency and rates of convergence in large covariance matrix estimation. *Annals of Statistics*, 37(6B):4254, 2009.

S. L. Lauritzen. *Graphical Models*. Clarendon Press, 1996.

H. Liu, K. Roeder, and L. Wasserman. Stability approach to regularization selection (stars) for high dimensional graphical models. *Journal of Machine Learning Research (JMLR)*, 10:2295–2328, 2009.

H. Liu, M. Xu, H. Gu, A. Gupta, J. Lafferty, and L. Wasserman. Forest density estimation. *J. of Machine Learning Research*, 12:907–951, 2011.

L. Lovász, V. Neumann-Lara, and M. Plummer. Mengerian theorems for paths of bounded length. *Periodica Mathematica Hungarica*, 9(4):269–276, 1978.

D.M. Malioutov, J.K. Johnson, and A.S. Willsky. Walk-sums and belief propagation in Gaussian graphical models. *J. of Machine Learning Research*, 7:2031–2064, 2006.

B.D. McKay, N.C. Wormald, and B. Wysocka. Short cycles in random regular graphs. *The Electronic Journal of Combinatorics*, 11(R66):1, 2004.

N. Meinshausen and P. Bühlmann. High dimensional graphs and variable selection with the lasso. *Annals of Statistics*, 34(3):1436–1462, 2006.

P. Netrapalli, S. Banerjee, S. Sanghavi, and S. Shakkottai. Greedy learning of Markov network structure . In *Proc. of Allerton Conf. on Communication, Control and Computing*, Monticello, USA, Sept. 2010.

J. Pearl. *Probabilistic Reasoning in Intelligent Systems—Networks of Plausible Inference*. Morgan Kaufmann, 1988.

P. Ravikumar, M.J. 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.J. Rothman, P.J. Bickel, E. Levina, and J. Zhu. Sparse permutation invariant covariance estimation. *Electronic Journal of Statistics*, 2:494–515, 2008.

H. Rue and L. Held. *Gaussian Markov Random Fields: Theory and Applications*. Chapman and Hall, London, 2005.

N.P. Santhanam and M.J. Wainwright. Information-theoretic Limits of high-dimensional model selection. In *International Symposium on Information Theory*, Toronto, Canada, July 2008.

G. Schwarz. Estimating the dimension of a model. *Annals of Statistics*, 6(2):461–464, 1978.

P. Spirtes and C. Meek. Learning bayesian networks with discrete variables from data. In *Proc. of Intl. Conf. on Knowledge Discovery and Data Mining*, pages 294–299, 1995.

V.Y.F. Tan, A. Anandkumar, and A. Willsky. Learning Gaussian tree models: analysis of error exponents and extremal structures. *IEEE Tran. on Signal Processing*, 58(5):2701–2714, May 2010.

V.Y.F. Tan, A. Anandkumar, and A. Willsky. A large-deviation analysis for the maximum likelihood learning of tree structures. *IEEE Tran. on Information Theory*, 57(3):1714–1735, March 2011a.

V.Y.F. Tan, A. Anandkumar, and A. Willsky. Learning Markov forest models: Analysis of error rates. *J. of Machine Learning Research*, 12:1617–1653, May 2011b.

W. Wang, M.J. Wainwright, and K. Ramchandran. Information-theoretic bounds on model selection for Gaussian Markov random fields. In *IEEE International Symposium on Information Theory Proceedings (ISIT)*, Austin, Tx, June 2010.

D.J. Watts and S.H. Strogatz. Collective dynamics of small-worldnetworks. *Nature*, 393(6684): 440–442, 1998.

X. Xie and Z. Geng. A recursive method for structural learning of directed acyclic graphs. *J. of Machine Learning Research*, 9:459–483, 2008.

Journal of Machine Learning Research 13 (2012) 2339-2365 Submitted 11/12; Published 9/12

A Local Spectral Method for Graphs: With Applications to Improving Graph Partitions and Exploring Data Graphs Locally

Michael W. Mahoney MMAHONEY@CS.STANFORD.EDU
Department of Mathematics
Stanford University
Stanford, CA 94305

Lorenzo Orecchia ORECCHIA@EECS.BERKELEY.EDU
Computer Science Division
University of California
Berkeley, CA 94720

Nisheeth K. Vishnoi NISHEETH.VISHNOI@GMAIL.COM
Microsoft Research India
#9, Lavelle Road
Bangalore 560 025
India

Editor: Mikhail Belkin

Abstract

The second eigenvalue of the Laplacian matrix and its associated eigenvector are fundamental features of an undirected graph, and as such they have found widespread use in scientific computing, machine learning, and data analysis. In many applications, however, graphs that arise have several *local* regions of interest, and the second eigenvector will typically fail to provide information fine-tuned to each local region. In this paper, we introduce a locally-biased analogue of the second eigenvector, and we demonstrate its usefulness at highlighting local properties of data graphs in a semi-supervised manner. To do so, we first view the second eigenvector as the solution to a constrained optimization problem, and we incorporate the local information as an additional constraint; we then characterize the optimal solution to this new problem and show that it can be interpreted as a generalization of a Personalized PageRank vector; and finally, as a consequence, we show that the solution can be computed in nearly-linear time. In addition, we show that this locally-biased vector can be used to compute an approximation to the best partition *near* an input seed set in a manner analogous to the way in which the second eigenvector of the Laplacian can be used to obtain an approximation to the best partition in the entire input graph. Such a primitive is useful for identifying and refining clusters locally, as it allows us to focus on a local region of interest in a semi-supervised manner. Finally, we provide a detailed empirical evaluation of our method by showing how it can applied to finding locally-biased sparse cuts around an input vertex seed set in social and information networks.

Keywords: spectral graph partitioning, local spectral algorithms, Laplacian matrix, semi-supervised learning, personalized pagerank

1. Introduction

Spectral methods are popular in machine learning, data analysis, and applied mathematics due to their strong underlying theory and their good performance in a wide range of applications. In the study of undirected graphs, in particular, spectral techniques play an important role, as many fundamental structural properties of a graph depend directly on spectral quantities associated with matrices representing the graph. Two fundamental objects of study in this area are the second smallest eigenvalue of the graph Laplacian and its associated eigenvector. These quantities determine many features of the graph, including the behavior of random walks and the presence of sparse cuts. This relationship between the graph structure and an easily-computable quantity has been exploited in data clustering, community detection, image segmentation, parallel computing, and many other applications.

A potential drawback of using the second eigenvalue and its associated eigenvector is that they are inherently *global* quantities, and thus they may not be sensitive to very *local* information. For instance, a sparse cut in a graph may be poorly correlated with the second eigenvector (and even with all the eigenvectors of the Laplacian) and thus invisible to a method based only on eigenvector analysis. Similarly, based on domain knowledge one might have information about a specific target region in the graph, in which case one might be interested in finding clusters only near this prespecified local region, for example, in a semi-supervised manner; but this local region might be essentially invisible to a method that uses only global eigenvectors. For these and related reasons, standard global spectral techniques can have substantial difficulties in semi-supervised settings, where the goal is to learn more about a locally-biased target region of the graph.

In this paper, we provide a methodology to construct a locally-biased analogue of the second eigenvalue and its associated eigenvector, and we demonstrate both theoretically and empirically that this localized vector inherits many of the good properties of the global second eigenvector. Our approach is inspired by viewing the second eigenvector as the optimum of a constrained global quadratic optimization program. To model the localization step, we modify this program by adding a natural locality constraint. This locality constraint requires that any feasible solution have sufficient correlation with the target region, which we assume is given as input in the form of a set of nodes or a distribution over vertices. The resulting optimization problem, which we name LocalSpectral and which is displayed in Figure 1, is the main object of our work.

The main advantage of our formulation is that an optimal solution to LocalSpectral captures many of the same structural properties as the global eigenvector, except in a locally-biased setting. For example, as with the global optimization program, our locally-biased optimization program has an intuitive geometric interpretation. Similarly, as with the global eigenvector, an optimal solution to LocalSpectral is efficiently computable. To show this, we characterize the optimal solutions of LocalSpectral and show that such a solution can be constructed in nearly-linear time by solving a system of linear equations. In applications where the eigenvectors of the graph are pre-computed and only a small number of them are needed to describe the data, the optimal solution to our program can be obtained by performing a small number of inner product computations. Finally, the optimal solution to LocalSpectral can be used to derive bounds on the mixing time of random walks that start near the local target region as well as on the existence of sparse cuts near the locally-biased target region. In particular, it lower bounds the conductance of cuts as a function of how well-correlated they are with the seed vector. This will allow us to exploit the analogy between global eigenvectors

and our localized analogue to design an algorithm for discovering sparse cuts near an input seed set of vertices.

In order to illustrate the empirical behavior of our method, we will describe its performance on the problem of finding locally-biased sparse cuts in real data graphs. Subsequent to the dissemination of the initial technical report version of this paper, our methodology was applied to the problem of finding, given a small number of "ground truth" labels that correspond to known segments in an image, the segments in which those labels reside (Maji, Vishnoi, and Malik, 2011). This computer vision application will be discussed briefly. Then, we will describe in detail how our algorithm for discovering sparse cuts near an input seed set of vertices may be applied to the problem of exploring data graphs locally and to identifying locally-biased clusters and communities in a more difficult-to-visualize social network application. In addition to illustrating the performance of the method in a practical application related to the one that initially motivated this work (Leskovec, Lang, Dasgupta, and Mahoney, 2008, 2009; Leskovec, Lang, and Mahoney, 2010), this social graph application will illustrate how the various "knobs" of our method can be used in practice to explore the structure of data graphs in a locally-biased manner.

Our method uses ideas from spectral graph theory; for a detailed introduction to this topic, see Chung (1997). Recent theoretical work has focused on using spectral ideas to find good clusters nearby an input seed set of nodes (Spielman and Teng, 2004; Andersen, Chung, and Lang, 2006; Chung, 2007). These methods are based on running a number of local random walks around the seed set and using the resulting distributions to extract information about clusters in the graph. This line of work grew out of attempts to develop linear equation solvers that run in time nearly linear in the number of edges in the graph (Spielman and Teng, 2004), and work subsequent to the initial technical report version of this paper has provided implementations of these ideas (Koutis, Miller, and Peng, 2010). The connections with our work described in this article remain to be explored.

Recent empirical work has used Personalized PageRank, a particular variant of a local random walk, to characterize very finely the clustering and community structure in a wide range of very large social and information networks (Andersen and Lang, 2006; Leskovec, Lang, Dasgupta, and Mahoney, 2008, 2009; Leskovec, Lang, and Mahoney, 2010). In contrast with previous methods, our local spectral method is the first to be derived in a direct way from an explicit optimization problem inspired by the global spectral problem. Interestingly, our characterization also shows that optimal solutions to LocalSpectral are generalizations of Personalized PageRank, providing an additional insight to why local random walk methods work well in practice.

In the next section, we will describe relevant background and notation; and then, in Section 3, we will present our formulation of a locally-biased spectral optimization program, the solution of which will provide a locally-biased analogue of the second eigenvector of the graph Laplacian. Then, in Section 4 we will describe how our method may be applied to identifying and refining locally-biased partitions in a graph; and in Section 5 we will provide a detailed empirical evaluation of our algorithm. Finally, in Section 6, we will conclude with a discussion of our results in a broader context.

2. Background and Notation

Let $G = (V, E, w)$ be a connected undirected graph with $n = |V|$ vertices and $m = |E|$ edges, in which edge $\{i, j\}$ has weight w_{ij}. For a set of vertices $S \subseteq V$ in a graph, the *volume of* S is $\mathrm{vol}(S) \stackrel{\mathrm{def}}{=} \sum_{i \in S} d_i$, in which case the *volume of the graph* G is $\mathrm{vol}(G) \stackrel{\mathrm{def}}{=} \mathrm{vol}(V) = 2m$. In the following, $A_G \in \mathbb{R}^{V \times V}$

will denote the adjacency matrix of G, while $D_G \in \mathbb{R}^{V\times V}$ will denote the diagonal degree matrix of G, that is, $D_G(i,i) = d_i = \sum_{\{i,j\}\in E} w_{ij}$, the weighted degree of vertex i. The Laplacian of G is defined as $L_G \stackrel{\text{def}}{=} D_G - A_G$. (This is also called the combinatorial Laplacian, in which case the normalized Laplacian of G is $\mathcal{L}_G \stackrel{\text{def}}{=} D_G^{-1/2} L_G D_G^{-1/2}$.)

The Laplacian is the symmetric matrix having quadratic form $x^T L_G x = \frac{1}{2}\sum_{ij\in E} w_{ij}(x_i - x_j)^2$, for $x \in \mathbb{R}^V$. This implies that L_G is positive semidefinite and that the all-one vector $1 \in \mathbb{R}^V$ is the eigenvector corresponding to the smallest eigenvalue 0. For a symmetric matrix A, we will use $A \succeq 0$ to denote that it is positive semi-definite. Moreover, given two symmetric matrices A and B, the expression $A \succeq B$ will mean $A - B \succeq 0$. Further, for two $n \times n$ matrices A and B, we let $A \circ B$ denote $\mathrm{Tr}(A^T B)$. Finally, for a matrix A, let A^+ denote its (uniquely defined) Moore-Penrose pseudoinverse.

For two vectors $x, y \in \mathbb{R}^n$, and the degree matrix D_G for a graph G, we define the degree-weighted inner product as $x^T D_G y \stackrel{\text{def}}{=} \sum_{i=1}^n x_i y_i d_i$. Given a subset of vertices $S \subseteq V$, we denote by 1_S the indicator vector of S in $\mathbb{R}^V$ and by 1 the vector in $\mathbb{R}^V$ having all entries set equal to 1. We consider the following definition of the complete graph K_n on the vertex set V: $A_{K_n} \stackrel{\text{def}}{=} \frac{1}{\mathrm{vol}(G)} D_G 1 1^T D_G$. Note that this is not the standard complete graph, but a weighted version of it, where the weights depend on D_G. With this scaling we have $D_{K_n} = D_G$. Hence, the Laplacian of the complete graph defined in this manner becomes $L_{K_n} = D_G - \frac{1}{\mathrm{vol}(G)} D_G 1 1^T D_G$.

In this paper, the *conductance* $\phi(S)$ *of a cut* $(S, \bar{S})$ is $\phi(S) \stackrel{\text{def}}{=} \mathrm{vol}(G) \cdot \frac{|E(S,\bar{S})|}{\mathrm{vol}(S)\cdot\mathrm{vol}(\bar{S})}$. A sparse cut, also called a good-conductance partition, is one for which $\phi(S)$ is small. The *conductance of the graph* G is then $\phi(G) = \min_{S\subseteq V} \phi(S)$. Note that the conductance of a set S, or equivalently a cut $(S, \bar{S})$, is often defined as $\phi'(S) = |E(S,\bar{S})| / \min\{\mathrm{vol}(S), \mathrm{vol}(\bar{S})\}$. This notion is equivalent to that $\phi(S)$, in that the value $\phi(G)$ thereby obtained for the conductance of the graph G differs by no more than a factor of 2 times the constant $\mathrm{vol}(G)$, depending on which notion we use for the conductance of a set.

3. The LocalSpectral Optimization Program

In this section, we introduce the local spectral optimization program $\mathsf{LocalSpectral}(G, s, \kappa)$ as a strengthening of the usual global spectral program $\mathsf{Spectral}(G)$. To do so, we will augment $\mathsf{Spectral}(G)$ with a locality constraint of the form $(x^T D_G s)^2 \geq \kappa$, for a seed vector s and a correlation parameter κ. Both these programs are homogeneous quadratic programs, with optimization variable the vector $x \in \mathbb{R}^V$, and thus any solution vector x is essentially equivalent to $-x$ for the purpose of these optimizations. Hence, in the following we do not differentiate between x and $-x$, and we assume a suitable direction is chosen in each instance.

3.1 Motivation for the Program

Recall that the second eigenvalue $\lambda_2(G)$ of the Laplacian L_G can be viewed as the optimum of the standard optimization problem $\mathsf{Spectral}(G)$ described in Figure 1. In matrix terminology, the corresponding optimal solution v_2 is a generalized eigenvector of L_G with respect to D_G. For our purposes, however, it is best to consider the geometric meaning of this optimization formulation. To do so, suppose we are operating in a vector space $\mathbb{R}^V$, where the ith dimension is stretched by a factor of d_i, so that the natural identity operator is D_G and the inner product between two vectors

$$\begin{aligned} \min \quad & x^T L_G x \\ \text{s.t.} \quad & x^T D_G x = 1 \\ & (x^T D_G 1)^2 = 0 \\ & x \in \mathbb{R}^V \end{aligned} \qquad \begin{aligned} \min \quad & x^T L_G x \\ \text{s.t.} \quad & x^T D_G x = 1 \\ & (x^T D_G 1)^2 = 0 \\ & (x^T D_G s)^2 \geq \kappa \\ & x \in \mathbb{R}^V \end{aligned}$$

Figure 1: Global and local spectral optimization programs. Left: The usual spectral program $\mathsf{Spectral}(G)$. Right: Our new locally-biased spectral program $\mathsf{LocalSpectral}(G,s,\kappa)$. In both cases, the optimization variable is the vector $x \in \mathbb{R}^n$.

x and y is given by $\sum_{i \in V} d_i x_i y_i = x^T D_G y$. In this representation, $\mathsf{Spectral}(G)$ is seeking the vector $x \in \mathbb{R}^V$ that is orthogonal to the all-one vector, lies on the unit sphere, and minimizes the Laplacian quadratic form. Note that such an optimum v_2 may lie anywhere on the unit sphere.

Our goal here is to modify $\mathsf{Spectral}(G)$ to incorporate a bias towards a target region which we assume is given to us as an input vector s. We will assume (without loss of generality) that s is properly normalized and orthogonalized so that $s^T D_G s = 1$ and $s^T D_G 1 = 0$. While s can be a general unit vector orthogonal to 1, it may be helpful to think of s as the indicator vector of one or more vertices in V, corresponding to the target region of the graph. We obtain $\mathsf{LocalSpectral}(G,s,\kappa)$ from $\mathsf{Spectral}(G)$ by requiring that a feasible solution also have a sufficiently large correlation with the vector s. This is achieved by the addition of the constraint $(x^T D_G s)^2 \geq \kappa$, which ensures that the projection of x onto the direction s is at least $\sqrt{\kappa}$ in absolute value, where the parameter κ is also an input parameter ranging between 0 and 1. Thus, we would like the solution to be well-connected with or to lie near the *seed* vector s. In particular, as displayed pictorially in Figure 2, x must lie within the spherical cap centered at s that contains all vectors at an angle of at most $\arccos(\sqrt{\kappa})$ from s. Thus, higher values of κ demand a higher correlation with s and, hence, a stronger localization. Note that in the limit $\kappa = 0$, the spherical cap constituting the feasible region of the program is guaranteed to include v_2 and $\mathsf{LocalSpectral}(G,s,\kappa)$ is equivalent to $\mathsf{Spectral}(G)$. In the rest of this paper, we refer to s as the *seed vector* and to κ as the *correlation parameter* for a given $\mathsf{LocalSpectral}(G,s,\kappa)$ optimization problem. Moreover, we denote the objective value of the program $\mathsf{LocalSpectral}(G,s,\kappa)$ by the number $\lambda(G,s,\kappa)$.

3.2 Characterization of the Optimal Solutions of LocalSpectral

Our first theorem is a characterization of the optimal solutions of LocalSpectral. Although LocalSpectral is a non-convex program (as, of course, is Spectral), the following theorem states that solutions to it can be expressed as the solution to a system of linear equations which has a natural interpretation. The proof of this theorem (which may be found in Section 3.4) will involve a relaxation of the non-convex program LocalSpectral to a convex semidefinite program (SDP), that is, the variables in the optimization program will be distributions over vectors rather than the vectors themselves. For the statement of this theorem, recall that A^+ denotes the (uniquely defined) Moore-Penrose pseudoinverse of the matrix A.

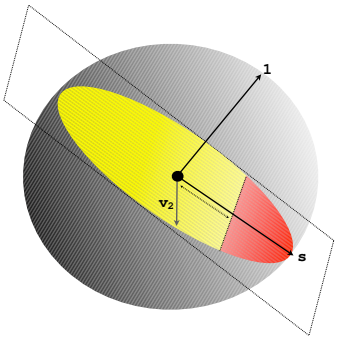


Figure 2: (Best seen in color.) Pictorial representation of the feasible regions of the optimization programs Spectral(G) and LocalSpectral(G,s,κ) that are defined in Figure 1. See the text for a discussion.

Theorem 1 (Solution Characterization) *Let $s \in \mathbb{R}^V$ be a seed vector such that $s^T D_G 1 = 0$, $s^T D_G s = 1$, and $s^T D_G v_2 \neq 0$, where v_2 is the second generalized eigenvector of L_G with respect to D_G. In addition, let $1 > \kappa \geq 0$ be a correlation parameter, and let $x^\star$ be an optimal solution to* LocalSpectral(G,s,κ). *Then, there exists some $\gamma \in (-\infty, \lambda_2(G))$ and a $c \in [0,\infty]$ such that*

$$x^\star = c(L_G - \gamma D_G)^+ D_G s. \tag{1}$$

There are several parameters (such as s, κ, γ, and c) in the statement of Theorem 1, and understanding their relationship is important: s and κ are the parameters of the program; c is a normalization factor that rescales the norm of the solution vector to be 1 (and that can be computed in linear time, given the solution vector); and γ is implicitly defined by κ, G, and s. The correct setting of γ ensures that $(s^T D_G x^\star)^2 = \kappa$, that is, that $x^\star$ is found exactly on the boundary of the feasible region. At this point, it is important to notice the behavior of $x^\star$ and γ as κ changes. As κ goes to 1, γ tends to $-\infty$ and $x^\star$ approaches s; conversely, as κ goes to 0, γ goes to $\lambda_2(G)$ and $x^\star$ tends towards v_2, the global eigenvector. We will discuss how to compute γ and $x^\star$, given a specific κ, in Section 3.3.

Finally, we should note that there is a close connection between the solution vector $x^\star$ and the popular PageRank procedure. Recall that PageRank refers to a method to determine a global rank or global notion of importance for a node in a graph such as the web that is based on the link structure of the graph (Brin and Page, 1998; Langville and Meyer, 2004; Berkhin, 2005). There have been several extensions to the basic PageRank concept, including Topic-Sensitive PageRank (Haveliwala, 2003) and Personalized PageRank (Jeh and Widom, 2003). In the same way that PageRank can be viewed as a way to express the quality of a web page over the entire web, Personalized PageRank expresses a link-based measure of page quality around user-selected pages. In particular, given a vector $s \in \mathbb{R}^V$ and a *teleportation* constant $\alpha > 0$, the Personalized PageRank vector can be written as $\mathrm{pr}_{\alpha,s} = \left(L_G + \frac{1-\alpha}{\alpha} D_G\right)^{-1} D_G s$ (Andersen, Chung, and Lang, 2006). By setting $\gamma = -\frac{1-\alpha}{\alpha}$, the optimal solution to LocalSpectral is proved to be a generalization of Personalized PageRank. In particular, this means that for high values of the correlation parameter κ, for which the corresponding γ in Theorem 1 is negative, the optimal solution to LocalSpectral takes the form of a Personalized PageRank vector. On the other hand, when $\gamma \geq 0$, the optimal solution to LocalSpectral provides a

smooth way of transitioning from the Personalized PageRank vector to the global second eigenvector v_2.

3.3 Computation of the Optimal Solutions of LocalSpectral

Here, we discuss how to compute efficiently an optimal solution for $\mathsf{LocalSpectral}(G,s,\kappa)$, for a fixed choice of the parameters G, s, and κ. The following theorem is our main result.

Theorem 2 (Solution Computation) *For any $\varepsilon > 0$, a solution to $\mathsf{LocalSpectral}(G,s,\kappa)$ of value at most $(1+\varepsilon)\cdot\lambda(G,s,\kappa)$ can be computed in time $\tilde{O}(m/\sqrt{\lambda_2(G)}\cdot\log(1/\varepsilon))$ using the Conjugate Gradient Method (Golub and Loan, 1996). Alternatively, such a solution can be computed in time $\tilde{O}(m\log(1/\varepsilon))$ using the Spielman-Teng linear-equation solver (Spielman and Teng, 2004).*

Proof By Theorem 1, we know that the optimal solution $x^\star$ must be a unit-scaled version of $y(\gamma) = (L_G - \gamma D_G)^+ D_G s$, for an appropriate choice of $\gamma \in (-\infty, \lambda_2(G))$. Notice that, given a fixed γ, the task of computing $y(\gamma)$ is equivalent to solving the system of linear equations $(L_G - \gamma D_G)y = D_G s$ for the unknown y. This operation can be performed, up to accuracy ε, in time $\tilde{O}(m/\sqrt{\lambda_2(G)}\cdot\log(1/\varepsilon))$ using the Conjugate Gradient Method, or in time $\tilde{O}(m\log(1/\varepsilon))$ using the Spielman-Teng linear-equation solver. To find the correct setting of γ, it suffices to perform a binary search over the possible values of γ in the interval $(-\mathrm{vol}(G), \lambda_2(G))$, until $(s^T D_G x)^2$ is sufficiently close to κ. ■

We should note that, depending on the application, other methods of computing a solution to $\mathsf{LocalSpectral}(G,s,\kappa)$ might be more appropriate. In particular, if an eigenvector decomposition of L_G has been pre-computed, as is the case in certain machine learning and data analysis applications, then this computation can be modified as follows. Given an eigenvector decomposition of L_G as $L_G = \sum_{i=2}^n \lambda_i D_G^{1/2} u_i u_i^T D_G^{1/2}$, then $y(\gamma)$ must take the form

$$y(\gamma) = (L_G - \gamma D_G)^+ D_G s = \sum_{i=2}^{n} \frac{1}{\lambda_i - \gamma} (s^T D_G^{1/2} u_i)^2,$$

for the same choice of c and γ, as in Theorem 1. Hence, given the eigenvector decomposition, each guess $y(\gamma)$ of the binary search can be computed by expanding the above series, which requires a linear number of inner product computations. While this may yield a worse running time than Theorem 2 in the worst case, in the case that the graph is well-approximated by a small number k of dominant eigenvectors, then the computation is reduced to only k straightforward inner product computations.

3.4 Proof of Theorem 1

We start with an outline of the proof. Although the program $\mathsf{LocalSpectral}(G,s,\kappa)$ is *not* convex, it can be relaxed to the convex semidefinite program $\mathsf{SDP}_p(G,s,\kappa)$ of Figure 3. Then, one can observe that strong duality holds for this SDP relaxation. Using strong duality and the related complementary slackness conditions, one can argue that the primal $\mathsf{SDP}_p(G,s,\kappa)$ has a rank one unique optimal solution under the conditions of the theorem. This implies that the optimal solution of $\mathsf{SDP}_p(G,s,\kappa)$ is the same as the optimal solution of $\mathsf{LocalSpectral}(G,s,\kappa)$. Moreover, combining this fact with the complementary slackness condition obtained from the dual $\mathsf{SDP}_d(G,s,\kappa)$ of Figure 3, one can derive that the optimal rank one solution is of the form promised by Theorem 1.

$$\begin{array}{rr} \text{minimize} & L_G \circ X \\ \text{s.t.} & L_{K_n} \circ X = 1 \\ & (D_G s)(D_G s)^T \circ X \geq \kappa \\ & X \succeq 0 \end{array} \qquad \begin{array}{rr} \text{maximize} & \alpha + \kappa\beta \\ \text{s.t.} & L_G \succeq \alpha L_{K_n} + \beta (D_G s)(D_G s)^T \\ & \beta \geq 0 \\ & \alpha \in \mathbb{R} \end{array}$$

Figure 3: Left: Primal SDP relaxation of $\mathsf{LocalSpectral}(G,s,\kappa)$: $\mathsf{SDP}_p(G,s,\kappa)$; for this primal, the optimization variable is $X \in \mathbb{R}^{V \times V}$ such that X is symmetric and positive semidefinite. Right: Dual SDP relaxation of $\mathsf{LocalSpectral}(G,s,\kappa)$: $\mathsf{SDP}_d(G,s,\kappa)$; for this dual, the optimization variables are $\alpha, \beta \in \mathbb{R}$. Recall that $L_{K_n} \stackrel{\text{def}}{=} D_G - \frac{1}{\text{vol}(G)} D_G 1 1^T D_G$.

Before proceeding with the details of the proof, we pause to make several points that should help to clarify our approach.

- First, since it may seem to some readers to be unnecessarily complex to relax LocalSpectral as an SDP, we emphasize that the motivation for relaxing it in this way is that we would like to prove Theorem 1. To prove this theorem, we must understand the form of the optimal solutions to the non-convex program LocalSpectral. Thus, in order to overcome the non-convexity, we relax LocalSpectral to $\mathsf{SDP}_p(G,s,\kappa)$ (of Figure 3) by "lifting" the rank-1 condition implicit in LocalSpectral. Then, strong duality applies; and it implies a set of sufficient optimality conditions. By combining these conditions, we will be able to establish that an optimal solution $X^\star$ to $\mathsf{SDP}_p(G,s,\kappa)$ has rank 1, that is, it has the form $X^\star = x^\star x^{\star T}$ for some vector $x^\star$; and thus it yields an optimal solution to LocalSpectral, that is, the vector $x^\star$.

- Second, in general, the value of a relaxation like $\mathsf{SDP}_p(G,s,\kappa)$ may be strictly less than that of the original program (LocalSpectral, in this case). Our characterization and proof will imply that the relaxation is tight, that is, that the optimum of $\mathsf{SDP}_p(G,s,\kappa)$ equals that of LocalSpectral. The reason is that one can find a rank-1 optimal solution to $\mathsf{SDP}_p(G,s,\kappa)$, which then yields an optimal solution of the same value for LocalSpectral. Note that this also implies that strong duality holds for the non-convex LocalSpectral, although this observation is not needed for our proof.

That is, although it may be possible to prove Theorem 1 in some other way that does not involve SDPs, we chose this proof since it is simple and intuitive and correct; and we note that Appendix B in the textbook of Boyd and Vandenberghe (2004) proves a similar statement by the same SDP-based approach.

Returning to the details of the proof, we will proceed to prove the theorem by establishing a sequence of claims. First, consider $\mathsf{SDP}_p(G,s,\kappa)$ and its dual $\mathsf{SDP}_d(G,s,\kappa)$ (as shown in Figure 3). The following claim uses the fact that, given $X = xx^T$ for $x \in \mathbb{R}^V$, and for any matrix $A \in \mathbb{R}^{V \times V}$, we have that $A \circ X = x^T A x$. In particular, $L_G \circ X = x^T L_G x$, for any graph G, and $(x^T D_G s)^2 = x^T D_G s s^T D_G x = D_G s s^T D_G \circ X$.

Claim 1 *The primal* $\mathsf{SDP}_p(G,s,\kappa)$ *is a relaxation of the vector program* $\mathsf{LocalSpectral}(G,s,\kappa)$.

Proof Consider a vector x that is a feasible solution to $\mathsf{LocalSpectral}(G,s,\kappa)$, and note that $X = xx^T$ is a feasible solution to $\mathsf{SDP}_p(G,s,\kappa)$. ■

Next, we establish the strong duality of $\mathsf{SDP}_p(G,s,\kappa)$. (Note that the feasibility conditions and complementary slackness conditions stated below may not suffice to establish the optimality, in the absence of this claim; hence, without this claim, we could not prove the subsequent claims, which are needed to prove the theorem.)

Claim 2 *Strong duality holds between* $\mathsf{SDP}_p(G,s,\kappa)$ *and* $\mathsf{SDP}_d(G,s,\kappa)$.

Proof Since $\mathsf{SDP}_p(G,s,\kappa)$ is convex, it suffices to verify that Slater's constraint qualification condition (Boyd and Vandenberghe, 2004) is true for this primal SDP. Consider $X = ss^T$. Then, $(D_G s)(D_G s)^T \circ ss^T = (s^T D_G s)^2 = 1 > \kappa$. ■

Next, we use this result to establish the following two claims. In particular, strong duality allows us to prove the following claim showing the KKT-conditions, that is, the feasibility conditions and complementary slackness conditions stated below, suffice to establish optimality.

Claim 3 *The following feasibility and complementary slackness conditions are sufficient for a primal-dual pair* $X^\star, \alpha^\star, \beta^\star$ *to be an optimal solution. The feasibility conditions are:*

$$\begin{aligned} L_{K_n} \circ X^\star &= 1, \\ (D_G s)(D_G s)^T \circ X^\star &\geq \kappa, \\ L_G - \alpha^\star L_{K_n} - \beta^\star (D_G s)(D_G s)^T &\succeq 0, \text{ and} \\ \beta^\star &\geq 0, \end{aligned} \tag{2}$$

and the complementary slackness conditions are:

$$\alpha^\star (L_{K_n} \circ X^\star - 1) = 0,$$

$$\beta^\star ((D_G s)(D_G s)^T \circ X^\star - \kappa) = 0, \text{ and} \tag{3}$$

$$X^\star \circ (L_G - \alpha^\star L_{K_n} - \beta^\star (D_G s)(D_G s)^T) = 0. \tag{4}$$

Proof This follows from the convexity of $\mathsf{SDP}_p(G,s,\kappa)$ and Slater's condition (Boyd and Vandenberghe, 2004). ■

Claim 4 *These feasibility and complementary slackness conditions, coupled with the assumptions of the theorem, imply that* $X^\star$ *must be rank* 1 *and* $\beta^\star > 0$.

Proof Plugging in v_2 in Equation (2), we obtain that $v_2^T L_G v_2 - \alpha^\star - \beta^\star (v_2^T D_G s)^2 \geq 0$. But $v_2^T L_G v_2 = \lambda_2(G)$ and $\beta^\star \geq 0$. Hence, $\lambda_2(G) \geq \alpha^\star$. Suppose $\alpha^\star = \lambda_2(G)$. As $s^T D_G v_2 \neq 0$, it must be the case that $\beta^\star = 0$. Hence, by Equation (4), we must have $X^\star \circ L(G) = \lambda_2(G)$, which implies that $X^\star = v_2 v_2^T$, that is, the optimum for LocalSpectral is the global eigenvector v_2. This corresponds to a choice of $\gamma = \lambda_2(G)$ and c tending to infinity.

Otherwise, we may assume that $\alpha^\star < \lambda_2(G)$. Hence, since G is connected and $\alpha^\star < \lambda_2(G)$, $L_G - \alpha^\star L_{K_n}$ has rank exactly $n-1$ and kernel parallel to the vector 1. From the complementary slackness condition (4) we can deduce that the image of $X^\star$ is in the kernel of $L_G - \alpha^\star L_{K_n} - \beta^\star (D_G s)(D_G s)^T$. If

$\beta^\star > 0$, we have that $\beta^\star(D_G s)(D_G s)^T$ is a rank one matrix and, since $s^T D_G 1 = 0$, it reduces the rank of $L_G - \alpha^\star L_{K_n}$ by one precisely. If $\beta^\star = 0$ then $X^\star$ must be 0 which is not possible if $\mathsf{SDP}_p(G,s,\kappa)$ is feasible. Hence, the rank of $L_G - \alpha^\star L_{K_n} - \beta^\star(D_G s)(D_G s)^T$ must be exactly $n-2$. As we may assume that 1 is in the kernel of $X^\star$, $X^\star$ must be of rank one. This proves the claim. ■

Now we complete the proof of the theorem. From the claim it follows that, $X^\star = x^\star x^{\star T}$ where $x^\star$ satisfies the equation $(L_G - \alpha^\star L_{K_n} - \beta^\star(D_G s)(D_G s)^T)x^\star = 0$. From the second complementary slackness condition, Equation (3), and the fact that $\beta^\star > 0$, we obtain that $(x^\star)^T D_G s = \pm\sqrt{\kappa}$. Thus, $x^\star = \pm\beta^\star\sqrt{\kappa}(L_G - \alpha^\star L_{K_n})^+ D_G s$, as required.

4. Application to Partitioning Graphs Locally

In this section, we describe the application of LocalSpectral to finding locally-biased partitions in a graph, that is, to finding sparse cuts around an input seed vertex set in the graph. For simplicity, in this part of the paper, we let the instance graph G be unweighted.

4.1 Background on Global Spectral Algorithms for Partitioning Graphs

We start with a brief review of global spectral graph partitioning. Recall that the basic global graph partitioning problem is: given as input a graph $G = (V,E)$, find a set of nodes $S \subseteq V$ to solve

$$\phi(G) = \min_{S \subseteq V} \phi(S).$$

Spectral methods approximate the solution to this intractable global problem by solving the relaxed problem $\mathsf{Spectral}(G)$ presented in Figure 1. To understand this optimization problem, recall that $x^T L_G x$ counts the number of edges crossing the cut and that $x^T D_G x = 1$ encodes a variance constraint; thus, the goal of $\mathsf{Spectral}(G)$ is to minimize the number of edges crossing the cut subject to a given variance. Recall that for $T \subseteq V$, we let $1_T \in \{0,1\}^V$ be a vector which is 1 for vertices in T and 0 otherwise. Then for a cut $(S,\bar{S})$, if we define the vector $v_S \stackrel{\text{def}}{=} \sqrt{\frac{\mathrm{vol}(S)\cdot\mathrm{vol}(\bar{S})}{\mathrm{vol}(G)}} \cdot \left(\frac{1_S}{\mathrm{vol}(S)} - \frac{1_{\bar{S}}}{\mathrm{vol}\bar{S}}\right)$, it can be checked that v_S satisfies the constraints of Spectral and has objective value $\phi(S)$. Thus, $\lambda_2(G) \leq \min_{S\subseteq V}\phi(S) = \phi(G)$.

Hence, $\mathsf{Spectral}(G)$ is a relaxation of the minimum conductance problem. Moreover, this program is a good relaxation in that a good cut can be recovered by considering a truncation, that is, a sweep cut, of the vector v_2 that is the optimal solution to $\mathsf{Spectral}(G)$. (That is, for example, consider each of the n cuts defined by the vector v_2, and return the cut with minimum conductance value.) This is captured by the following celebrated result often referred to as Cheeger's Inequality.

Theorem 3 (Cheeger's Inequality) *For a connected graph G, $\phi(G) \leq O(\sqrt{\lambda_2(G)})$.*

Although there are many proofs known for this theorem (see, e.g., Chung, 1997), a particularly interesting proof was found by Mihail (1989); this proof involves rounding any *test vector* (rather than just the optimal vector), and it achieves the same guarantee as Cheeger's Inequality.

Theorem 4 (Sweep Cut Rounding) *Let x be a vector such that $x^T D_G 1 = 0$. Then there is a t for which the set of vertices $S := \mathsf{SweepCut}_t(x) \stackrel{\text{def}}{=} \{i : x_i \geq t\}$ satisfies $\frac{x^T L_G x}{x^T D_G x} \geq \phi^2(S)/8$.*

It is the form of Cheeger's Inequality provided by Theorem 4 that we will use below.

4.2 Locally-Biased Spectral Graph Partitioning

Here, we will exploit the analogy between Spectral and LocalSpectral by applying the global approach just outlined to the following locally-biased graph partitioning problem: given as input a graph $G = (V,E)$, an input node u, and a positive integer k, find a set of nodes $T \subseteq V$ achieving

$$\phi(u,k) = \min_{T \subseteq V : u \in T, \mathrm{vol}(T) \leq k} \phi(T).$$

That is, the problem is to find the best conductance set of nodes of volume no greater than k that contains the input node v.

As a first step, we show that we can choose the seed set and correlation parameters s and κ such that LocalSpectral(G,s,κ) is a relaxation for this locally-biased graph partitioning problem.

Lemma 5 *For $u \in V$, LocalSpectral$(G, v_{\{u\}}, 1/k)$ is a relaxation of the problem of finding a minimum conductance cut T in G which contains the vertex u and is of volume at most k. In particular, $\lambda(G, v_{\{u\}}, 1/k) \leq \phi(u,k)$.*

Proof If we let $x = v_T$ in LocalSpectral$(G, v_{\{u\}}, 1/k)$, then $v_T^T L_G v_T = \phi(T)$, $v_T^T D_G 1 = 0$, and $v_T^T D_G v_T = 1$. Moreover, we have that $(v_T^T D_G v_{\{u\}})^2 = \frac{d_u(2m - \mathrm{vol}(T))}{\mathrm{vol}(T)(2m - d_u)} \geq 1/k$, which establishes the lemma. ■

Next, we can apply Theorem 4 to the optimal solution for LocalSpectral$(G, v_{\{u\}}, 1/k)$ and obtain a cut T whose conductance is quadratically close to the optimal value $\lambda(G, v_{\{u\}}, 1/k)$. By Lemma 5, this implies that $\phi(T) \leq O(\sqrt{\phi(u,k)})$. This argument proves the following theorem.

Theorem 6 (Finding a Cut) *Given an unweighted graph $G = (V,E)$, a vertex $u \in V$ and a positive integer k, we can find a cut in G of conductance at most $O(\sqrt{\phi(u,k)})$ by computing a sweep cut of the optimal vector for LocalSpectral$(G, v_{\{u\}}, 1/k)$. Moreover, this algorithm runs in nearly-linear time in the size of the graph.*

That is, this theorem states that we can perform a sweep cut over the vector that is the solution to LocalSpectral$(G, v_{\{u\}}, 1/k)$ in order to obtain a locally-biased partition; and that this partition comes with quality-of-approximation guarantees analogous to that provided for the global problem Spectral(G) by Cheeger's inequality.

Our final theorem shows that the optimal value of LocalSpectral also provides a lower bound on the conductance of *other cuts*, as a function of how well-correlated they are with the input seed vector. In particular, when the seed vector corresponds to a cut U, this result allows us to lower bound the conductance of an arbitrary cut T, in terms of the correlation between U and T. The proof of this theorem also uses in an essential manner the duality properties that were used in the proof of Theorem 1.

Theorem 7 (Cut Improvement) *Let G be a graph and $s \in \mathbb{R}^n$ be such that $s^T D_G 1 = 0$, where D_G is the degree matrix of G. In addition, let $\kappa \geq 0$ be a correlation parameter. Then, for all sets $T \subseteq V$ such that $\kappa' \stackrel{\mathrm{def}}{=} (s^T D_G v_T)^2$, we have that*

$$\phi(T) \geq \begin{cases} \lambda(G,s,\kappa) & \text{if } \kappa \leq \kappa' \\ \kappa'/\kappa \cdot \lambda(G,s,\kappa) & \text{if } \kappa' \leq \kappa. \end{cases}$$

Proof It follows from Theorem 1 that $\lambda(G,s,\kappa)$ is the same as the optimal value of $\mathsf{SDP}_p(G,s,\kappa)$ which, by strong duality, is the same as the optimal value of $\mathsf{SDP}_d(G,s,\kappa)$. Let $\alpha^\star,\beta^\star$ be the optimal dual values to $\mathsf{SDP}_d(G,s,\kappa)$. Then, from the dual feasibility constraint $L_G - \alpha^\star L_{K_n} - \beta^\star(D_G s)(D_G s)^T \succeq 0$, it follows that

$$s_T^T L_G s_T - \alpha^\star s_T^T L_{K_n} s_T - \beta^\star (s^T D_G s_T)^2 \geq 0.$$

Notice that since $s_T^T D_G 1 = 0$, it follows that $s_T^T L_{K_n} s_T = s_T^T D_G s_T = 1$. Further, since $s_T^T L_G s_T = \phi(T)$, we obtain, if $\kappa \leq \kappa'$, that

$$\phi(T) \geq \alpha^\star + \beta^\star (s^T D_G s_T)^2 \geq \alpha^\star + \beta^\star \kappa = \lambda(G,s,\kappa).$$

If on the other hand, $\kappa' \leq \kappa$, then

$$\phi(T) \geq \alpha^\star + \beta^\star (s^T D_G s_T)^2 \geq \alpha^\star + \beta^\star \kappa \geq {}^{\kappa'}\!/_{\kappa} \cdot (\alpha^\star + \beta^\star \kappa) = {}^{\kappa'}\!/_{\kappa} \cdot \lambda(G,s,\kappa).$$

Note that strong duality was used here. ■

Thus, although the relaxation guarantees of Lemma 5 only hold when the seed set is a single vertex, we can use Theorem 7 to consider the following problem: given a graph G and a cut $(T,\bar{T})$ in the graph, find a cut of minimum conductance in G which is well-correlated with T or certify that there is none. Although one can imagine many applications of this primitive, the main application that motivated this work was to explore clusters nearby or around a given *seed set* of nodes in data graphs. This will be illustrated in our empirical evaluation in Section 5.

4.3 Our Geometric Notion of Correlation Between Cuts

Here we pause to make explicit the geometric notion of correlation between cuts (or partitions, or sets of nodes) that is used by LocalSpectral, and that has already been used in various guises in previous sections. Given a cut $(T,\bar{T})$ in a graph $G=(V,E)$, a natural vector in $\mathbb{R}^V$ to associate with it is its characteristic vector, in which case the correlation between a cut $(T,\bar{T})$ and another cut $(U,\bar{U})$ can be captured by the inner product of the characteristic vectors of the two cuts. A somewhat more refined vector to associate with a cut is the vector obtained after removing from the characteristic vector its projection along the all-ones vector. In that case, again, a notion of correlation is related to the inner product of two such vectors for two cuts. More precisely, given a set of nodes $T \subseteq V$, or equivalently a cut $(T,\bar{T})$, one can define the unit vector s_T as

$$s_T(i) = \begin{cases} \sqrt{\mathrm{vol}(T)\mathrm{vol}(\bar{T})/2m} \cdot 1/\mathrm{vol}(T) & \text{if } i \in T \\ -\sqrt{\mathrm{vol}(T)\mathrm{vol}(\bar{T})/2m} \cdot 1/\mathrm{vol}(\bar{T}) & \text{if } i \in \bar{T}. \end{cases}$$

That is, $s_T \stackrel{\text{def}}{=} \sqrt{\frac{\mathrm{vol}(T)\mathrm{vol}(\bar{T})}{2m}} \left(\frac{1_T}{\mathrm{vol}(T)} - \frac{1_{\bar{T}}}{\mathrm{vol}(\bar{T})} \right)$, which is exactly the vector defined in Section 4.1. It is easy to check that this is well defined: one can replace s_T by $s_{\bar{T}}$ and the correlation remains the same with any other set. Moreover, several observations are immediate. First, defined this way, it immediately follows that $s_T^T D_G 1 = 0$ and that $s_T^T D_G s_T = 1$. Thus, $s_T \in \mathcal{S}_D$ for $T \subseteq V$, where we denote by $\mathcal{S}_D$ the set of vectors $\{x \in \mathbb{R}^V : x^T D_G 1 = 0\}$; and s_T can be seen as an appropriately normalized version of the vector consisting of the uniform distribution over T minus the uniform

distribution over $\bar{T}$.[1] Second, one can introduce the following measure of correlation between two sets of nodes, or equivalently between two cuts, say a cut $(T,\bar{T})$ and a cut $(U,\bar{U})$:

$$K(T,U) \stackrel{\text{def}}{=} (s_T D_G s_U)^2.$$

The proofs of the following simple facts regarding $K(T,U)$ are omitted: $K(T,U) \in [0,1]$; $K(T,U) = 1$ if and only if $T = U$ or $\bar{T} = U$; $K(T,U) = K(\bar{T},U)$; and $K(T,U) = K(T,\bar{U})$. Third, although we have described this notion of geometric correlation in terms of vectors of the form $s_T \in \mathcal{S}_D$ that represent partitions $(T,\bar{T})$, this correlation is clearly well-defined for other vectors $s \in \mathcal{S}_D$ for which there is not such a simple interpretation in terms of cuts. Indeed, in Section 3 we considered the case that s was an arbitrary vector in $\mathcal{S}_D$, while in the first part of Section 4.2 we considered the case that s was the seed set of a single node. In our empirical evaluation in Section 5, we will consider both of these cases as well as the case that s encodes the correlation with cuts consisting of multiple nodes.

5. Empirical Evaluation

In this section, we provide an empirical evaluation of LocalSpectral by illustrating its use at finding and evaluating locally-biased low-conductance cuts, that is, sparse cuts or good clusters, around an input seed set of nodes in a data graph. We start with a brief discussion of a very recent and pictorially-compelling application of our method to a computer vision problem; and then we discuss in detail how our method can be applied to identify clusters and communities in a more heterogeneous and more difficult-to-visualize social network application.

5.1 Semi-Supervised Image Segmentation

Subsequent to the initial dissemination of the technical report version of this paper, Maji, Vishnoi, and Malik (2011) applied our methodology to the problem of finding locally-biased cuts in a computer vision application. Recall that image segmentation is the problem of partitioning a digital image into segments corresponding to significant objects and areas in the image. A standard approach consists in converting the image data into a similarity graph over the the pixels and applying a graph partitioning algorithm to identify relevant segments. In particular, spectral methods have been popular in this area since the work of Shi and Malik (2000), which used the second eigenvector of the graph to approximate the so-called normalized cut (which, recall, is an objective measure for image segmentation that is practically equivalent to conductance). However, a difficulty in applying the normalized cut method is that in many cases global eigenvectors may fail to capture important local segments of the image. The reason for this is that they aggressively optimize a global objective function and thus they tend to combine multiple segments together; this is illustrated pictorially in the first row of Figure 4.

This difficulty can be overcome in a semi-supervised scenario by using our LocalSpectral method. Specifically, one often has a small number of "ground truth" labels that correspond to known segments, and one is interested in extracting and refining the segments in which those labels reside. In this case, if one considers an input seed corresponding to a small number of pixels within a target

1. Notice also that $s_T = -s_{\bar{T}}$. Thus, since we only consider quadratic functions of s_T, we can consider both s_T and $s_{\bar{T}}$ to be representative vectors for the cut $(T,\bar{T})$.

object, then LocalSpectral can recover the corresponding segment with high precision. This is illustrated in the second row of Figure 4. This computer vision application of our methodology was motivated by a preliminary version of this paper, and it was described in detail and evaluated against competing algorithms by Maji, Vishnoi, and Malik (2011). In particular, they show that LocalSpectral achieves a performance superior to that of other semi-supervised segmentation algorithms (Yu and Shi, 2002; Eriksson, Olsson, and Kahl, 2007); and they also show how LocalSpectral can be incorporated in an unsupervised segmentation pipeline by using as input seed distributions obtained by an object-detector algorithm (Bourdev, Maji, Brox, and Malik, 2010).

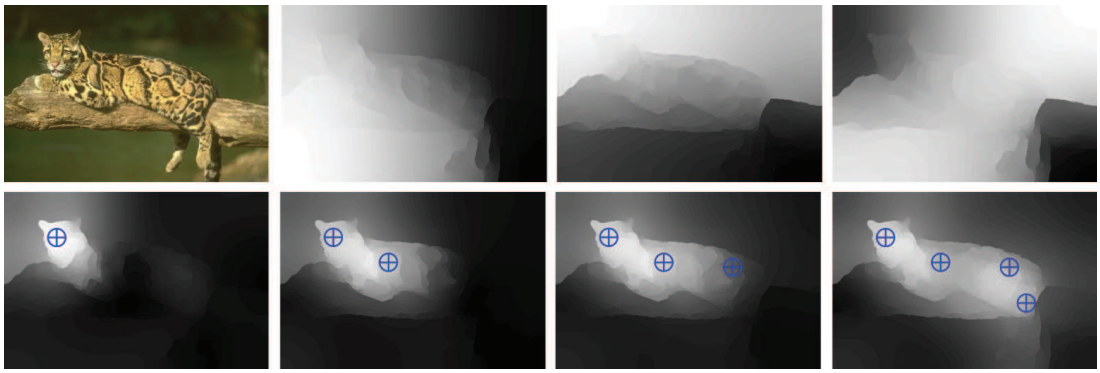


Figure 4: The first row shows the input image and the three smallest eigenvectors of the Laplacian of the corresponding similarity graph computed using the intervening contour cue (Maire, Arbelaez, Fowlkes, and Malik, 2008). Note that no sweep cut of these eigenvectors reveals the leopard. The second row shows the results of LocalSpectral with a setting of $\gamma = -10\lambda_2(G)$ with the seed pixels highlighted by crosshairs. Note how one can to recover the leopard by using a seed vector representing a set of only 4 pixels. In addition, note how the first seed pixel allows us to capture the head of the animal, while the other seeds help reveal other parts of its body.

5.2 Detecting Communities in Social Networks

Finding local clusters and meaningful locally-biased communities is also of interest in the analysis of large social and information networks. A standard approach to finding clusters and communities in many network analysis applications is to formalize the idea of a good community with an "edge counting" metric such as conductance or modularity and then to use a spectral relaxation to optimize it approximately (Newman, 2006b,a). For many very large social and information networks, however, there simply do not exist good large global clusters, but there do exist small meaningful local clusters that may be thought of as being nearby prespecified seed sets of nodes (Leskovec, Lang, Dasgupta, and Mahoney, 2008, 2009; Leskovec, Lang, and Mahoney, 2010). In these cases, a local version of the global spectral partitioning problem is of interest, as was shown by Leskovec, Lang, and Mahoney (2010). Typical networks are very large and, due to their expander-like properties, are not easily-visualizable (Leskovec, Lang, Dasgupta, and Mahoney, 2008, 2009). Thus, in order to illustrate the empirical behavior of our LocalSpectral methodology in a "real" network application

related to the one that motivated this work (Leskovec, Lang, Dasgupta, and Mahoney, 2008, 2009; Leskovec, Lang, and Mahoney, 2010), we examined a small "coauthorship network" of scientists. This network was previously used by Newman (2006b) to study community structure in small social and information networks.

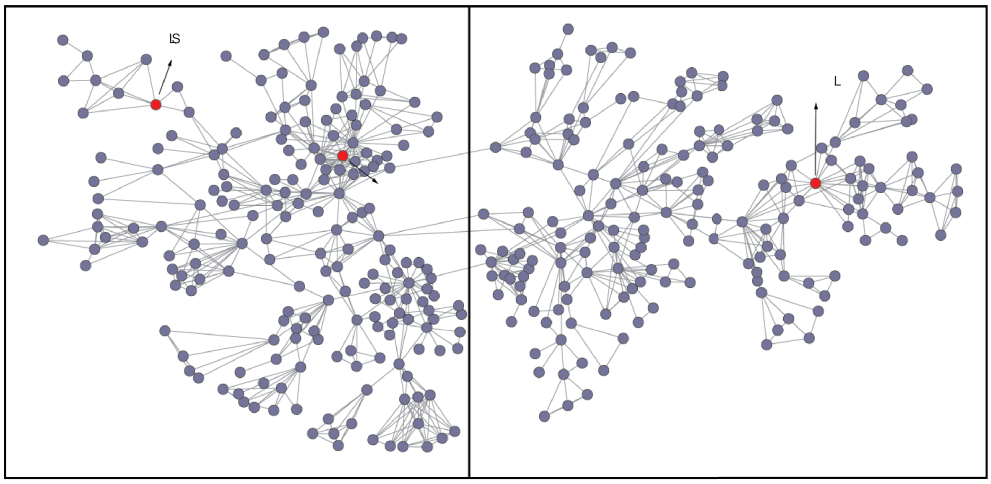


Figure 5: The coauthorship network of Newman (2006b). This layout was obtained in the Pajek (Batagelj and Mrvar, 2001) visualization software, using the Kamada-Kawai method (Kamada and Kawai, 1989) on each component of a partition provided by LocalCut and tiling the layouts at the end. Boxes show the two main global components of the network, which are displayed separately in subsequent figures.

The corresponding graph G is illustrated in Figure 5 and consists of 379 nodes and 914 edges, where each node represents an author and each unweighted edge represents a coauthorship relationship. The spectral gap $\lambda_2(G) = 0.0029$; and a sweep cut of the eigenvector corresponding to this second eigenvalue yields the globally-optimal spectral cut separating the graph into two well-balanced partitions, corresponding to the left half and the right half of the network, as shown in Figure 5. Our main empirical observations, described in detail in the remainder of this section, are the following.

- First, we show how varying the teleportation parameter allows us to detect low-conductance cuts of different volumes that are locally-biased around a prespecified seed vertex; and how this information, aggregated over multiple choices of teleportation, can improve our understanding of the network structure in the neighborhood of the seed.

- Second, we demonstrate the more general usefulness of our definition of a *generalized* Personalized PageRank vector (where the γ parameter in Equation (1) can be $\gamma \in (-\infty, \lambda_2(G))$ by

displaying specific instances in which that vector is more effective than the usual Personalized PageRank (where only positive teleportation probabilities are allowed and thus where γ must be negative). We do this by detecting a wider range of low-conductance cuts at a given volume and by interpolating smoothly between very locally-biased solutions to LocalSpectral and the global solution provided by the Spectral program.

- Third, we demonstrate how our method can find low-conductance cuts that are well-correlated to more general input seed vectors by demonstrating an application to the detection of sparse peripheral regions, for example, regions of the network that are well-correlated with low-degree nodes. This suggests that our method may find applications in leveraging feature data, which are often associated with the vertices of a data graph, to find interesting and meaningful cuts.

We emphasize that the goal of this empirical evaluation is to illustrate how our proposed methodology can be applied in real applications; and thus we work with a relatively easy-to-visualize example of a small social graph. This will allow us to illustrate how the "knobs" of our proposed method can be used in practice. In particular, the goal is not to illustrate that our method or heuristic variants of it or other spectral-based methods scale to much larger graphs—this latter fact is by now well-established (Andersen and Lang, 2006; Leskovec, Lang, Dasgupta, and Mahoney, 2008, 2009; Leskovec, Lang, and Mahoney, 2010).

5.2.1 ALGORITHM DESCRIPTION AND IMPLEMENTATION

We refer to our cut-finding algorithm, which will be used to guide our empirical study of finding and evaluating cuts around an input seed set of nodes and which is a straightforward extension of the algorithm referred to in Theorem 6, as LocalCut. In addition to the graph, the input parameters for LocalCut are a seed vector s (e.g., corresponding to a single vertex v), a teleportation parameter γ, and (optionally) a size factor c. Then, LocalCut performs the following steps.

- First, compute the vector $x^\star$ of Equation (1) with seed s and teleportation γ.
- Second, either perform a sweep of the vector $x^\star$, for example, consider each of the n cuts defined by the vector and return the the minimum conductance cut found along the sweep; or consider only sweep cuts along the vector $x^\star$ of volume at most $c \cdot k_\gamma$, where $k_\gamma = 1/\kappa_\gamma$, that contain the input vertex v, and return the minimum conductance cut among such cuts.

By Theorem 1, the vector computed in the first step of LocalCut, $x^\star$, is an optimal solution to LocalSpectral(G, s, κ_γ) for some choice of κ_γ. (Indeed, by fixing the above parameters, the κ parameter is fixed implicitly.) Then, by Theorem 6, when the vector $x^\star$ is rounded (to, for example, $\{-1, +1\}$) by performing the sweep cut, provably-good approximations are guaranteed. In addition, when the seed vector corresponds to a single vertex v, it follows from Lemma 5 that $x^\star$ yields a lower bound to the conductance of cuts that contain v and have less than a certain volume k_γ.

Although the full sweep-cut rounding does not give a specific guarantee on the volume of the output cut, empirically we have found that it is often possible to find small low-conductance cuts in the range dictated by k_γ. Thus, in our empirical evaluation, we also consider volume-constrained sweep cuts (which departs slightly from the theory but can be useful in practice). That is, we also introduce a new input parameter, a *size factor* $c > 0$, that regulates the maximum volume of the sweep cuts considered when s represents a single vertex. In this case, LocalCut does not consider

all n cuts defined by the vector $x^\star$, but instead it considers only sweep cuts of volume at most $c \cdot k_\gamma$ that contain the vertex v. (Note that it is a simple consequence of our optimization characterization that the optimal vector has sweep cuts of volume at most k_γ containing v.) This new input parameter turns out to be extremely useful in exploring cuts at different sizes, as it neglects sweep cuts of low conductance at large volume and allows us to pick out more local cuts around the seed vertex.

In our first two sets of experiments, summarized in Sections 5.2.2 and 5.2.3, we used single-vertex seed vectors, and we analyzed the effects of varying the parameters γ and c, as a function of the location of the seed vertex in the input graph. In the last set of experiments, presented in Section 5.2.4, we considered more general seed vectors, including both seed vectors that correspond to multiple nodes, that is, to cuts or partitions in the graph, as well as seed vectors that do not have an obvious interpretation in terms of input cuts. We implemented our code in a combination of MATLAB and C++, solving linear systems using the Stabilized Biconjugate Gradient Method (van der Vorst, 1992) provided in MATLAB 2006b. On this particular coauthorship network, and on a Dell PowerEdge 1950 machine with 2.33 GHz and 16GB of RAM, the algorithm ran in less than a few seconds.

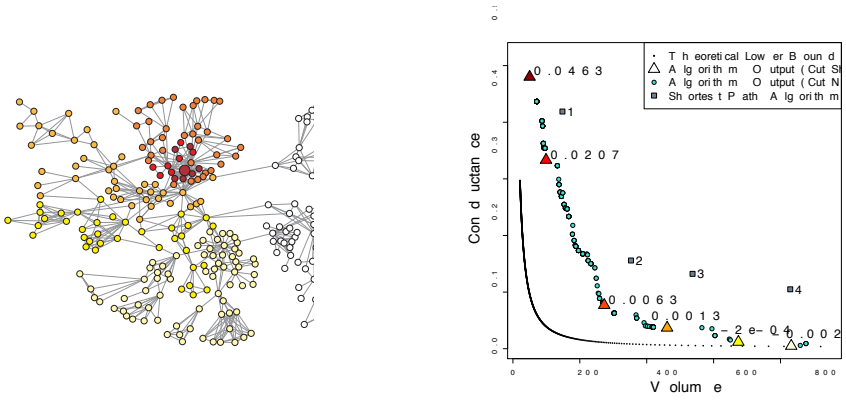
5.2.2 VARYING THE TELEPORTATION PARAMETER

Here, we evaluate the effect of varying the teleportation parameter $\gamma \in (-\infty, \lambda_2(G))$, where recall $\lambda_2(G) = 0.0029$. Since it is known that large social and information networks are quite heterogeneous and exhibit a very strong "nested core-periphery" structure (Leskovec, Lang, Dasgupta, and Mahoney, 2008, 2009; Leskovec, Lang, and Mahoney, 2010), we perform this evaluation by considering the behavior of LocalCut when applied to three types of seed nodes, examples of which are the highlighted vertices in Figure 5. These three nodes were chosen to represent three different types of nodes seen in larger networks: a *periphery-like node*, which belongs to a lower-degree and less expander-like part of the graph, and which tends to be surrounded by lower-conductance cuts of small volume; a *core-like node*, which belongs to a denser and higher-conductance or more expander-like part of the graph; and an *intermediate node*, which belongs to a regime between the core-like and the periphery-like regions.

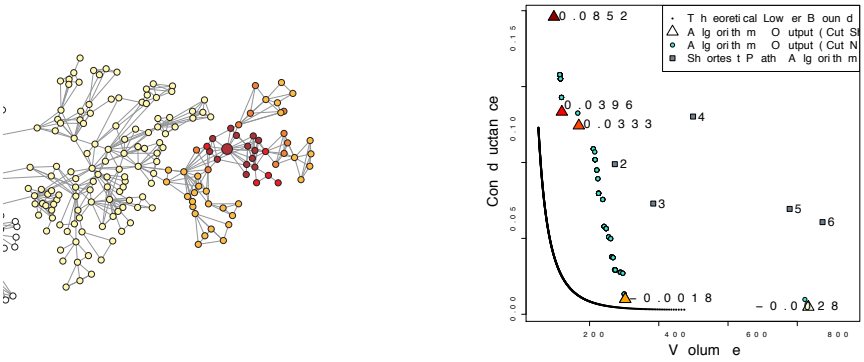
For each of the three representative seed nodes, we executed 1000 runs of LocalCut with $c = 2$ and γ varying by 0.001 increments. Figure 6 displays, for each of these three seeds, a plot of the conductance as a function of volume of the cuts found by each run of LocalCut. We refer to this type of plot as a *local profile plot* since it is a specialization of the *network community profile plot* (Leskovec, Lang, Dasgupta, and Mahoney, 2008, 2009; Leskovec, Lang, and Mahoney, 2010) to cuts around the specified seed vertex. In addition, Figure 6 also plots several other quantities of interest: first, the volume and conductance of the theoretical lower bound yielded by each run; second, the volume and conductance of the cuts defined by the shortest-path balls (in squares and numbered according to the length of the path) around each seed (which should and do provide a sanity-check upper bound); third, next to each of the plots, we present a color-coded/grayscale image of representative cuts detected by LocalCut; and fourth, for each of the cuts illustrated on the left, a color-coded/grayscale triangle and the numerical value of $-\gamma$ is shown on the right.

Several points about the behavior of the LocalCut algorithm as a function of the location of the input seed node and that are illustrated in Figure 6 are worth emphasizing.

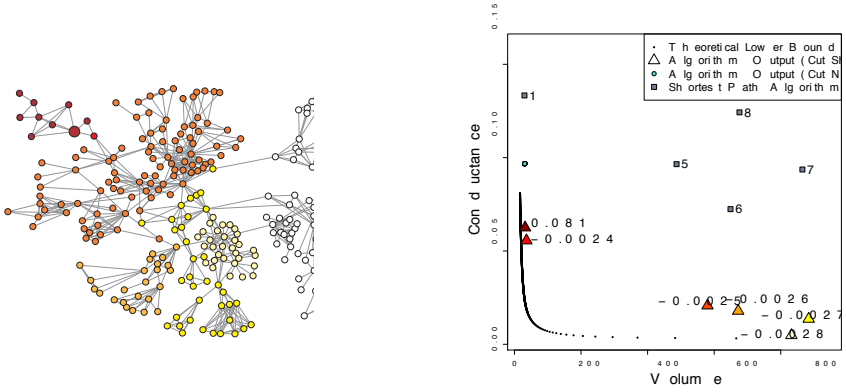
- First, for the core-like node, whose profile plot is shown in Figure 6(a), the volume of the output cuts grows relatively smoothly as γ is increased (i.e., as $-\gamma$ is decreased). For small



(a) Selected cuts and profile plot for the *core-like node*.



(b) Selected cuts and profiles plot for the *intermediate node*.



(c) Selected cuts and profile plot for the *periphery-like node*.

Figure 6: Selected cuts and local profile plots for varying γ. The cuts on the left are displayed by assigning to each vertex a color corresponding to the smallest selected cut in which the vertex was included. Smaller cuts are darker, larger cuts are lighter; and the seed vertex is shown slightly larger. Each profile plot on the right shows results from 1000 runs of LocalCut, with $c = 2$ and γ decreasing in 0.001 increments starting at 0.0028. For each color-coded/grayscale triangle, corresponding to a cut on the left, $-\gamma$ is also listed.

γ, for example, $\gamma = -0.0463$ or $\gamma = -0.0207$, the output cuts are forced to be small and hence display high conductance, as the region around the node is somewhat expander-like. By decreasing the teleportation, the conductance progressively decreases, as the rounding starts to hit nodes in peripheral regions, whose inclusion only improves conductance (since it increases the cut volume without adding many additional cut edges). In this case, this phenomena ends at $\gamma = -0.0013$, when a cut of conductance value close to that of the global optimum is found. (After that, larger and slightly better conductance cuts can still be found, but, as discussed below, they require $\gamma > 0$.)

- Second, a similar interpretation applies to the profile plot of the intermediate node, as shown in Figure 6(b). Here, however, the global component of the network containing the seed has smaller volume, around 300, and a very low conductance (again, requiring $\gamma > 0$). Thus, the profile plot *jumps* from this cut to the much larger eigenvector sweep cut, as will be discussed below.

- Third, a more extreme case is that of the periphery-like node, whose profile plot is displayed in Figure 6(c). In this case, an initial increase in γ does *not* yield larger cuts. This vertex is contained in a small-volume cut of low conductance, and thus diffusion-based methods get "stuck" on the small side of the cut. The only cuts of lower conductance in the network are those separating the global components, which can only be accessed when $\gamma > 0$. Hence, the teleportation must be greatly decreased before the algorithm starts outputting cuts at larger volumes. (As an aside, this behavior is also often seen with so-called "whiskers" in much larger social and information networks (Leskovec, Lang, Dasgupta, and Mahoney, 2008, 2009; Leskovec, Lang, and Mahoney, 2010).)

In addition, several general points that are illustrated in Figure 6 are worth emphasizing about the behavior of our algorithm.

- First, LocalCut found low-conductance cuts of different volumes around each seed vertex, outperforming the shortest-path algorithm (as it should) by a factor of roughly 4 in most cases. However, the results of LocalCut still lie away from the lower bound, which is also a factor of roughly 4 smaller at most volumes.

- Second, consider the range of the teleportation parameter necessary for the LocalCut algorithm to discover the well-balanced globally-optimal spectral partition. In all three cases, it was necessary to make γ positive (i.e., $-\gamma$ negative) to detect the well-balanced global spectral cut. Importantly, however, the quantitative details depend strongly on whether the seed is core-like, intermediate, or periphery-like. That is, by *formally* allowing "negative teleportation" probabilities, which correspond to $\gamma > 0$, the use of *generalized* Personalized PageRank vectors as an exploratory tool is much stronger than the usual Personalized PageRank (Andersen, Chung, and Lang, 2006; Andersen and Lang, 2006), in that it permits one to find a larger class of clusters, up to and including the global partition found by the solution to the global Spectral program. Relatedly, it provides a smooth interpolation between Personalized PageRank and the second eigenvector of the graph. Indeed, for $\gamma = 0.0028 \approx \lambda_2(G)$, LocalCut outputs the same cut as the eigenvector sweep cut for all three seeds.

- Third, recall that, given a teleportation parameter γ, the rounding step selects the cut of smallest conductance along the sweep cut of the solution vector. (Alternatively, if volume-

constrained sweeps are considered, then it selects the cut of smallest conductance among sweep cuts of volume at most $c \cdot k_\gamma$, where k_γ is the lower bound obtained from the optimization program.) In either case, increasing γ can lead LocalCut to pick out larger cuts, but it does not *guarantee* this will happen. In particular, due to the local topology of the graph, in many instances there may *not* be a way of slightly increasing the volume of a cut while slightly decreasing its conductance. In those cases, LocalCut may output the same small sweep cut for a range of teleportation parameters until a much larger, much lower-conductance cut is then found. The presence such horizontal and vertical *jumps* in the local profile plot conveys useful information about the structure of the network in the neighborhood of the seed at different size scales, illustrating that the practice follows the theory quite well.

5.2.3 VARYING THE OUTPUT-SIZE PARAMETER

Here, we evaluate the effect of varying the size factor c, for a fixed choice of teleportation parameter γ. (In the previous section, c was fixed at $c = 2$ and γ was varied.) We have observed that varying c, like varying γ, tends to have the effect of producing low-conductance cuts of different volumes around the seed vertex. Moreover, it is possible to obtain low-conductance large-volume cuts, even at lower values of the teleportation parameter, by increasing c to a sufficiently large value. This is illustrated in Figure 7, which shows the result of varying c with the core-like node as the seed and $-\gamma = 0.02$. Figure 6(a) illustrated that when $c = 2$, this setting only yielded a cut of volume close to 100 (see the red/darker triangle with $-\gamma = 0.0207$); but the yellow/lighter crosses in Figure 7 illustrate that by allowing larger values of c, better conductance cuts of larger volume can be obtained.

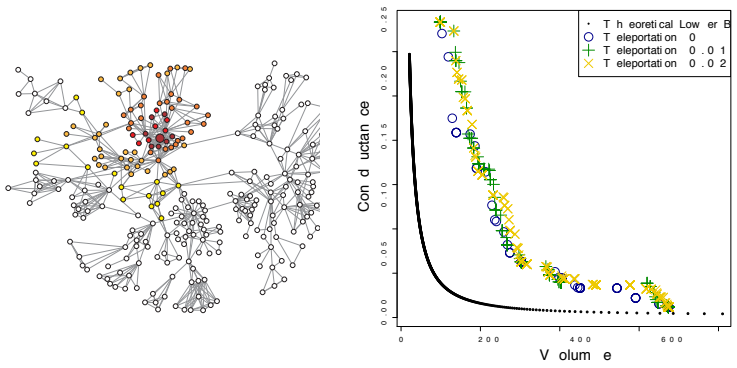


Figure 7: Selected cuts and local profile plots for varying c with the core-like node as the seed. The cuts are displayed by assigning to each vertex a color corresponding to the smallest selected cut in which the vertex was included. Smaller cuts are darker, larger are lighter. The seed vertex is shown larger. The profile plot shows results from 1000 runs of LocalCut, with varying c and $-\gamma \in \{0, 0.01, 0.02\}$.

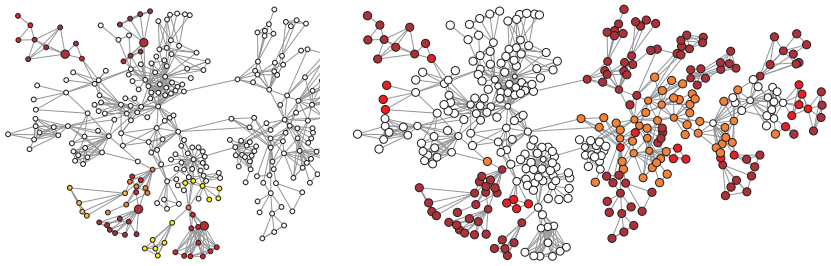
While many of these cuts tend to have conductance slightly worse than the best found by varying the teleportation parameter, the observation that cuts of a wide range of volumes can be obtained with a single value of γ leaves open the possibility that there exists a single choice of teleportation

parameter γ that produces good low-conductance cuts at all volumes simply by varying c. (This would allow us to only solve a single optimization problem and still find cuts of different volumes.) To address (and rule out) this possibility, we selected three choices of the teleportation parameter for each of the three seed nodes, and then we let c vary. The resulting output cuts for the core-like node as the seed are plotted in Figure 7. (The plots for the other seeds are similar and are not displayed.) Clearly, no single teleportation setting dominates the others: in particular, at volume 200 the lowest-conductance cut was produced with $-\gamma = 0.02$; at volume 400 it was produced with $-\gamma = 0.01$; and at volume 600 with it was produced with $\gamma = 0$. The highest choice of $\gamma = 0$ performed marginally better overall, recording lowest conductance cuts at both small and large volumes. That being said, the results of all three settings roughly track each other, and cuts of a wide range of volumes were able to be obtained by varying the size parameter c.

These and other empirical results suggest that the best results are achieved when we vary both the teleportation parameter and the size factor. In addition, the use of multiple teleportation choices have the side-effect advantage of yielding multiple lower bounds at different volumes.

5.2.4 MULTIPLE SEEDS AND CORRELATION

Here, we evaluate the behavior of LocalCut on more general seed vectors. We consider two examples—for the first example, there is an interpretation as a cut or partition consisting of multiple nodes; while the second example does not have any immediate interpretation in terms of cuts or partitions.



(a) Seed set of four seed nodes.

(b) A more general seed vector.

Figure 8: Multiple seeds and correlation. 8(a) shows selected cuts for varying γ with the seed vector corresponding to a subset of 4 vertices lying in the periphery-like region of the network. 8(b) shows selected cuts for varying γ with the seed vertex equal to a normalized version of the degree vector. In both cases, the cuts are displayed by assigning to each vertex a color corresponding to the smallest selected cut in which the vertex was included. Smaller cuts are darker, larger are lighter.

In our first example, we consider a seed vector representing a subset of four nodes, located in different peripheral branches of the left half of the global partition of the the network: see the four slightly larger (and darker) vertices in Figure 8(a). This is of interest since, depending on the size-scale at which one is interested, such sets of nodes can be thought of as either "nearby" or "far apart." For example, when viewing the entire graph of 379 nodes, these four nodes are all close, in

that they are all on the left side of the optimal global spectral partition; but when considering smaller clusters such as well-connected sets of 10 or 15 nodes, these four nodes are much farther apart. In Figure 8(a), we display a selection of the cuts found by varying the teleportation, with $c = 2$. The smaller cuts tend to contain the branches in which each seed node is found, while larger cuts start to incorporate nearby branches. Not shown in the color-coding/grayscale is that the optimal global spectral partition is eventually recovered. Identifying peripheral areas that are well-separated from the rest of the graph is a useful primitive in studying the structure of social networks (Leskovec, Lang, Dasgupta, and Mahoney, 2008, 2009; Leskovec, Lang, and Mahoney, 2010); and thus, this shows how LocalCut may be used in this context, when some periphery-like seed nodes of the graph are known.

In our second example, we consider a seed vector that represents a feature vector on the vertices but that does not have an interpretation in terms of cuts. In particular, we consider a seed vector that is a normalized version of the degree distribution vector. Since nodes that are periphery-like tend to have lower degree than those that are core-like (Leskovec, Lang, Dasgupta, and Mahoney, 2008, 2009; Leskovec, Lang, and Mahoney, 2010), this choice of seed vector biases LocalCut towards cuts that are well-correlated with periphery-like and low-degree vertices. A selection of the cuts found on this seed vector when varying the teleportation with $c = 2$ is displayed in Figure 8(b). These cuts partition the network naturally into three well-separated regions: a sparser periphery-like region in darker colors, a lighter-colored intermediate region, and a white dense core-like region, where higher-degree vertices tend to lie. Clearly, this approach could be applied more generally to find low-conductance cuts that are well-correlated with a known feature of the node vector.

6. Discussion

In this final section, we provide a brief discussion of our results in a broader context.

6.1 Relationship to Local Graph Partitioning

Recent theoretical work has focused on using spectral ideas to find good clusters nearby an input seed set of nodes (Spielman and Teng, 2004; Andersen, Chung, and Lang, 2006; Chung, 2007). In particular, local graph partitioning—roughly, the problem of finding a low-conductance cut in a graph in time depending only on the volume of the output cut—was introduced by Spielman and Teng (2004). They used random walk based methods; and they used this as a subroutine to give a nearly linear-time algorithm for outputting balanced cuts that match the Cheeger Inequality up to polylog factors. In our language, a local graph partitioning algorithm would start a random walk at a seed node, truncating the walk after a suitably chosen number of steps, and outputting the nodes visited by the walk. This result was improved by Andersen, Chung, and Lang (2006) by performing a truncated Personalized PageRank computation. These and subsequent papers building on them were motivated by local graph partitioning (Chung, 2007), but they do not address the problem of discovering cuts near general seed vectors, as do we, or of generalizing the second eigenvector of the Laplacian. Moreover, these approaches are more operationally-defined, while ours is axiomatic and optimization-based. For more details on these issues as well as the relationship between these issues and implementing regularization implicitly via approximate computation, see Mahoney and Orecchia (2011) and Mahoney (2012).

6.2 Relationship to Empirical Work on Community Structure

Recent empirical work has used Personalized PageRank, a particular variant of a local random walk, to characterize very finely the clustering and community structure in a wide range of very large social and information networks (Andersen and Lang, 2006; Leskovec, Lang, Dasgupta, and Mahoney, 2008, 2009; Leskovec, Lang, and Mahoney, 2010). In particular, Andersen and Lang used local spectral methods to identify communities in certain informatics graphs using an input set of nodes as a seed set (Andersen and Lang, 2006). Subsequently, Leskovec, Lang, Dasgupta, and Mahoney used related methods to characterize the small-scale and large-scale clustering and community structure in a wide range of large social and information networks (Leskovec, Lang, Dasgupta, and Mahoney, 2008, 2009; Leskovec, Lang, and Mahoney, 2010). Our optimization program and empirical results suggest that this line of work can be extended to ask in a theoretically principled manner much more refined questions about graph structure near prespecified seed vectors.

6.3 Relationship to Cut-improvement Algorithms

Many recently-popular algorithms for finding minimum-conductance cuts, such as those in Khandekar, Rao, and Vazirani (2006) and in Orecchia, Schulman, Vazirani, and Vishnoi (2008), use as a crucial building block a primitive that takes as input a cut $(T,\bar{T})$ and attempts to find a lower-conductance cut that is *well correlated* with $(T,\bar{T})$. This primitive is referred to as a *cut-improvement algorithm* (Lang and Rao, 2004; Andersen and Lang, 2008), as its original purpose was limited to post-processing cuts output by other algorithms. Recently, cut-improvement algorithms have also been used to find low conductance cuts in specific regions of large graphs (Leskovec, Lang, and Mahoney, 2010). Given a notion of correlation between cuts, cut-improvement algorithms typically produce approximation guarantees of the following form: for any cut $(C,\bar{C})$ that is ε-correlated with the input cut, the cut output by the algorithm has conductance upper-bounded by a function of the conductance of $(C,\bar{C})$ and ε. This line of work has typically used flow-based techniques. For example, Gallo, Grigoriadis, and Tarjan (1989) were the first to show that one can find a subset of an input set $T \subseteq V$ with minimum conductance in polynomial time. Similarly, Lang and Rao (2004) implement a closely related algorithm and demonstrate its effectiveness at refining cuts output by other methods. Finally, Andersen and Lang (2008) give a more general algorithm that uses a small number of single-commodity maximum-flows to find low-conductance cuts not only inside the input subset T, but among all cuts which are well-correlated with $(T,\bar{T})$. Viewed from this perspective, our work may be seen as a spectral analogue of these flow-based techniques, since Theorem 7 provides lower bounds on the conductance of other cuts as a function of how well-correlated they are with the seed vector.

6.4 Alternate Interpretation of Our Main Optimization Program

There are a few interesting ways to view our local optimization problem of Figure 1 which would like to point out here. Recall that LocalSpectral may be interpreted as augmenting the standard spectral optimization program with a constraint that the output cut be well-correlated with the input seed set. To understand this program from the perspective of the dual, recall that the dual of

LocalSpectral is given by the following.

$$\begin{aligned} \text{maximize} \quad & \alpha + \beta\kappa \\ \text{s.t.} \quad & L_G \succeq \alpha L_{K_n} + \beta\Omega_T \\ & \beta \geq 0, \end{aligned}$$

where $\Omega_T = D_G s_T s_T^T D_G$. Alternatively, by subtracting the second constraint of LocalSpectral from the first constraint, it follows that

$$x^T \left(L_{K_n} - L_{K_n} s_T s_T^T L_{K_n} \right) x \leq 1 - \kappa.$$

It can be shown that

$$L_{K_n} - L_{K_n} s_T s_T^T L_{K_n} = \frac{L_{K_T}}{\mathrm{vol}(\bar{T})} + \frac{L_{K_{\bar{T}}}}{\mathrm{vol}(T)},$$

where L_{K_T} is the D_G-weighted complete graph on the vertex set T. Thus, LocalSpectral is clearly equivalent to

$$\begin{aligned} \text{minimize} \quad & x^T L_G x \\ \text{s.t.} \quad & x^T L_{K_n} x = 1 \\ & x^T \left(\frac{L_{K_T}}{\mathrm{vol}(\bar{T})} + \frac{L_{K_{\bar{T}}}}{\mathrm{vol}(T)} \right) x \leq 1 - \kappa. \end{aligned}$$

The dual of this program is given by the following.

$$\begin{aligned} \text{maximize} \quad & \alpha - \beta(1-\kappa) \\ \text{s.t.} \quad & L_G \succeq \alpha L_{K_n} - \beta \left(\frac{L_{K_T}}{\mathrm{vol}(\bar{T})} + \frac{L_{K_{\bar{T}}}}{\mathrm{vol}(T)} \right) \\ & \beta \geq 0. \end{aligned}$$

From the perspective of this dual, this can be viewed as "embedding" a combination of a complete graph K_n and a weighted combination of complete graphs on the sets T and $\bar{T}$, that is, K_T and $K_{\bar{T}}$. Depending on the value of β, the latter terms clearly discourage cuts that substantially cut into T or $\bar{T}$, thus encouraging partitions that are well-correlated with the input cut $(T, \bar{T})$.

6.5 Bounding the Size of the Output Cut

Readers familiar with the spectral method may recall that given a graph with a small balanced cut, it is not possible, in general, to guarantee that the sweep cut procedure of Theorem 4 applied to the optimal of Spectral outputs a balanced cut. One may have to iterate several times before one gets a balanced cut. Our setting, building up on the spectral method, also suffers from this; we cannot hope, in general, to bound the size of the output cut (which is a sweep cut) in terms of the correlation parameter κ. This was the reason for considering volume-constrained sweep cuts in our empirical evaluation.

References

R. Andersen and K. Lang. Communities from seed sets. In *WWW '06: Proceedings of the 15th International Conference on World Wide Web*, pages 223–232, 2006.

R. Andersen and K. Lang. An algorithm for improving graph partitions. In *SODA '08: Proceedings of the 19th ACM-SIAM Symposium on Discrete algorithms*, pages 651–660, 2008.

R. Andersen, F.R.K. Chung, and K. Lang. Local graph partitioning using PageRank vectors. In *FOCS '06: Proceedings of the 47th Annual IEEE Symposium on Foundations of Computer Science*, pages 475–486, 2006.

V. Batagelj and A. Mrvar. Pajek—analysis and visualization of large networks. In *Proceedings of Graph Drawing*, pages 477–478, 2001.

P. Berkhin. A survey on PageRank computing. *Internet Mathematics*, 2(1):73–120, 2005.

L. Bourdev, S. Maji, T. Brox, and J. Malik. Detecting people using mutually consistent poselet activations. In *Proceedings of the 11th European Conference on Computer Vision*, 2010.

S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, Cambridge, UK, 2004.

S. Brin and L. Page. The anatomy of a large-scale hypertextual web search engine. In *Proceedings of the 7th International Conference on World Wide Web*, pages 107–117, 1998.

F.R.K. Chung. *Spectral Graph Theory*, volume 92 of *CBMS Regional Conference Series in Mathematics*. American Mathematical Society, 1997.

F.R.K Chung. Four proofs of Cheeger inequality and graph partition algorithms. In *Proceedings of ICCM*, 2007.

A. P. Eriksson, C. Olsson, and F. Kahl. Normalized cuts revisited: A reformulation for segmentation with linear grouping constraints. In *Proceedings of the 11th International Conference on Computer Vision*, pages 1–8, 2007.

G. Gallo, M.D. Grigoriadis, and R.E. Tarjan. A fast parametric maximum flow algorithm and applications. *SIAM Journal on Computing*, 18(1):30–55, 1989.

G.H. Golub and C.F. Van Loan. *Matrix Computations*. Johns Hopkins University Press, Baltimore, 1996.

T.H. Haveliwala. Topic-sensitive PageRank: A context-sensitive ranking algorithm for web search. *IEEE Transactions on Knowledge and Data Engineering*, 15(4):784–796, 2003.

G. Jeh and J. Widom. Scaling personalized web search. In *WWW '03: Proceedings of the 12th International Conference on World Wide Web*, pages 271–279, 2003.

T. Kamada and S. Kawai. An algorithm for drawing general undirected graphs. *Information Processing Letters*, 31(1):7–15, 1989.

R. Khandekar, S. Rao, and U. Vazirani. Graph partitioning using single commodity flows. In *STOC '06: Proceedings of the 38th annual ACM Symposium on Theory of Computing*, pages 385–390, 2006.

I. Koutis, G. L. Miller, and R. Peng. Approaching optimality for solving SDD linear systems. In *Proceedings of the 51st Annual IEEE Symposium on Foundations of Computer Science*, pages 235–244, 2010.

K. Lang and S. Rao. A flow-based method for improving the expansion or conductance of graph cuts. In *IPCO '04: Proceedings of the 10th International IPCO Conference on Integer Programming and Combinatorial Optimization*, pages 325–337, 2004.

A. N. Langville and C. D. Meyer. Deeper inside PageRank. *Internet Mathematics*, 1(3):335–380, 2004.

J. Leskovec, K.J. Lang, A. Dasgupta, and M.W. Mahoney. Statistical properties of community structure in large social and information networks. In *WWW '08: Proceedings of the 17th International Conference on World Wide Web*, pages 695–704, 2008.

J. Leskovec, K.J. Lang, A. Dasgupta, and M.W. Mahoney. Community structure in large networks: Natural cluster sizes and the absence of large well-defined clusters. *Internet Mathematics*, 6(1): 29–123, 2009. Also available at: arXiv:0810.1355.

J. Leskovec, K.J. Lang, and M.W. Mahoney. Empirical comparison of algorithms for network community detection. In *WWW '10: Proceedings of the 19th International Conference on World Wide Web*, pages 631–640, 2010.

M. W. Mahoney. Approximate computation and implicit regularization for very large-scale data analysis. In *Proceedings of the 31st ACM Symposium on Principles of Database Systems*, pages 143–154, 2012.

M. W. Mahoney and L. Orecchia. Implementing regularization implicitly via approximate eigenvector computation. In *Proceedings of the 28th International Conference on Machine Learning*, pages 121–128, 2011.

M. Maire, P. Arbelaez, C. Fowlkes, and J. Malik. Using contours to detect and localize junctions in natural images. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8, 2008.

S. Maji, N. K. Vishnoi, and J. Malik. Biased normalized cuts. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2057–2064, 2011.

M. Mihail. Conductance and convergence of Markov chains—a combinatorial treatment of expanders. In *Proceedings of the 30th Annual IEEE Symposium on Foundations of Computer Science*, pages 526–531, 1989.

M.E.J. Newman. Modularity and community structure in networks. *Proceedings of the National Academy of Sciences of the United States of America*, 103(23):8577–8582, 2006a.

M.E.J. Newman. Finding community structure in networks using the eigenvectors of matrices. *Physical Review E*, 74:036104, 2006b.

L. Orecchia, L. Schulman, U.V. Vazirani, and N.K. Vishnoi. On partitioning graphs via single commodity flows. In *Proceedings of the 40th Annual ACM Symposium on Theory of Computing*, pages 461–470, 2008.

J. Shi and J. Malik. Normalized cuts and image segmentation. *IEEE Transcations of Pattern Analysis and Machine Intelligence*, 22(8):888–905, 2000.

D.A. Spielman and S.-H. Teng. Nearly-linear time algorithms for graph partitioning, graph sparsification, and solving linear systems. In *STOC '04: Proceedings of the 36th annual ACM Symposium on Theory of Computing*, pages 81–90, 2004.

H. A. van der Vorst. Bi-CGSTAB: a fast and smoothly converging variant of Bi-CG for the solution of nonsymmetric linear systems. *SIAM J. Sci. and Stat. Comput.*, 13(2):631–644, 1992.

S. X. Yu and J. Shi. Grouping with bias. In *Annual Advances in Neural Information Processing Systems 14: Proceedings of the 2001 Conference*, pages 1327–1334, 2002.

Journal of Machine Learning Research 13 (2012) 2367-2407 Submitted 9/10; Revised 12/11; Published 8/12

Multi-Target Regression with Rule Ensembles

Timo Aho[*] TIMO.AHO@IKI.FI
Department of Software Systems
Tampere University of Technology
P. O. Box 553
FI-33101 Tampere, Finland

Bernard Ženko[†] BERNARD.ZENKO@IJS.SI
Sašo Džeroski[‡] SASO.DZEROSKI@IJS.SI
Department of Knowledge Technologies
Jožef Stefan Institute
Jamova cesta 39
SI-1000 Ljubljana, Slovenia

Tapio Elomaa TAPIO.ELOMAA@TUT.FI
Department of Software Systems
Tampere University of Technology
P. O. Box 553
FI-33101 Tampere, Finland

Editor: Carla Brodley

Abstract

Methods for learning decision rules are being successfully applied to many problem domains, in particular when understanding and interpretation of the learned model is necessary. In many real life problems, we would like to predict multiple related (nominal or numeric) target attributes simultaneously. While several methods for learning rules that predict multiple targets at once exist, they are all based on the covering algorithm, which does not work well for regression problems. A better solution for regression is the rule ensemble approach that transcribes an ensemble of decision trees into a large collection of rules. An optimization procedure is then used to select the best (and much smaller) subset of these rules and to determine their respective weights.

We introduce the FIRE algorithm for solving multi-target regression problems, which employs the rule ensembles approach. We improve the accuracy of the algorithm by adding simple linear functions to the ensemble. We also extensively evaluate the algorithm with and without linear functions. The results show that the accuracy of multi-target regression rule ensembles is high. They are more accurate than, for instance, multi-target regression trees, but not quite as accurate as multi-target random forests. The rule ensembles are significantly more concise than random forests, and it is also possible to create compact rule sets that are smaller than a single regression tree but still comparable in accuracy.

Keywords: multi-target prediction, rule learning, rule ensembles, regression

∗. Also in Microtask, Tampere, Finland.

†. Also in the Centre of Excellence for Integrated Approaches in Chemistry and Biology of Proteins, Ljubljana, Slovenia.

‡. Also in the Centre of Excellence for Integrated Approaches in Chemistry and Biology of Proteins, Ljubljana, Slovenia and the Jožef Stefan International Postgraduate School, Ljubljana, Slovenia.

1. Introduction

In the most common machine learning setting, one predicts the value of a single target attribute, categorical or numeric. A natural generalization of this setting is to predict multiple target attributes simultaneously. The task comes in two slightly different flavors. In *multi-target prediction* (Blockeel et al., 1998), all target attributes are (equally) important and predicted simultaneously with a single model. *Multi-task learning* (Caruana, 1997), on the other hand, originally focused on a single target attribute and used the rest for assistance only. Nowadays, however, multi-task models typically predict each target attribute individually but with at least partially distinct models.

A typical example coming from the environmental sciences is the task of predicting species distribution or community structure (Demšar et al., 2006), where we are interested in predicting the abundances of a set of different species living in the same environment. These species represent the target attributes, which might, but need not be related. Examples from other areas, ranging from natural language processing to bioinformatics and medicine are also plentiful (Jeong and Lee, 2009; Liu et al., 2010; Bickel et al., 2008).

With multiple targets, a typical solution is to create a collection of single-target models. Nevertheless, especially if we are interested in the interpretability of the model, the collection of single-target models is more complex and harder to interpret than a single model that jointly predicts all target attributes (Blockeel, 1998; Suzuki et al., 2001; Ženko and Džeroski, 2008). Furthermore, learning several tasks together may increase the predictive performance for the individual tasks due to inductive transfer, where the knowledge from one task is transfered to the other tasks (Piccart et al., 2008; Kocev et al., 2007; Suzuki et al., 2001). An additional benefit of the multi-target models is that they are less likely to overfit the data than the corresponding collections of single-target models (Blockeel, 1998; Caruana, 1997).

Rule sets, together with decision trees, are one of the most expressive and human readable model representations. They are frequently used when an interpretable model is desired. The majority of rule learning methods are based on the sequential covering algorithm (Michalski, 1969), originally designed for learning ordered rule lists for binary classification domains. This is also the case with the existing methods for learning multi-target rules (Ženko and Džeroski, 2008; Ženko, 2007). Unfortunately, on both single-target and multi-target regression problems, the accuracy of rule sets that are learned by the sequential covering approach is considerably lower than that of other regression methods, like for example, regression trees (for an empirical comparison see Ženko, 2007).

An alternative rule learning method that performs well also on (single-target) regression problems is the approach of *rule ensembles* (Friedman and Popescu, 2005, 2008; Dembczyński et al., 2008). It creates a collection of rules and uses an optimization procedure with the purpose of finding a small (and therefore interpretable) subset of rules. Optionally, rules can be combined with simple linear functions of descriptive attributes.

In this paper, we introduce FIRE, an algorithm for multi-target regression based on the rule ensembles approach. On one hand, we compare our approach with multi-target random forests (Kocev et al., 2007), which yield good accuracy at the expense of (very) large models. On the other hand, we compare it with multi-target regression trees (Blockeel et al., 1998) and multi-target model trees (Appice and Džeroski, 2007), both of which give rise to small models of (only) moderate accuracy. Our approach provides a solution that lies in between these two extremes: It produces accurate multi-target regression models that are significantly more concise than random forests. In

addition, the algorithm enables us to adjust the trade-off between the interpretability and accuracy of the learned models as desired.

An early version of the FIRE algorithm has been presented in a conference paper (Aho et al., 2009). However, the algorithm presented in this paper adds the ability to combine rules with simple linear functions, which increases its accuracy. In addition, the paper improves several details of the optimization procedure used. The new version stops the optimization if continuing seems unlikely to be fruitful. Moreover, instead of choosing the step size for the optimization algorithm in an ad-hoc manner, the choice is now made in a more sound manner as detailed in Appendix A.

Finally, the present paper includes a significantly extended empirical evaluation: We consider a larger collection of data sets and methods with which our algorithm is compared, including the latest rule ensemble methods (Dembczyński et al., 2008) and multi-target model trees (Appice and Džeroski, 2007), which combine tree models and linear functions. In addition, we compare our approach with a recent multi-task learning algorithm (Jalali et al., 2011).

The remainder of this article is organized as follows. Section 2 presents related work on multi-target prediction, rule learning, and rule ensembles. The FIRE algorithm for learning multi-target regression rule ensembles is introduced in Section 3. Section 4 describes the experimental evaluation setting and Section 5 reports the results of the empirical evaluation. The last section concludes and gives some directions for further research.

2. Related Work

In the *multi-target prediction* task, we are given a set of training examples E of the form $(\boldsymbol{x}, \boldsymbol{y})$, where $\boldsymbol{x} = (x_1, x_2, \ldots, x_K)$ is a vector of K descriptive attributes and $\boldsymbol{y} = (y_1, y_2, \ldots, y_T)$ is a vector of T target attributes. Our task is to learn a model that, given a new unlabeled instance $\boldsymbol{x}$, can predict the values of all target attributes $\boldsymbol{y}$ simultaneously. Several standard learning methods such as neural networks, decision trees, model trees, classification rules and random forests have been extended towards multi-target prediction (Caruana, 1997; Blockeel et al., 1998; Appice and Džeroski, 2007; Suzuki et al., 2001; Ženko and Džeroski, 2008; Kocev et al., 2007).

An approach related to multi-target learning is *multi-task learning* (Caruana, 1997; Argyriou et al., 2008; Chapelle et al., 2010; Jalali et al., 2011; Rakotomamonjy et al., 2011; Parameswaran and Weinberger, 2011). In multi-task learning, the aim is to solve multiple single-target learning tasks $(\boldsymbol{x}, y)_{t=1}^{T}$ with different training sets E_t (and in general with different descriptive attributes) at the same time. Multi-task learning should be able to benefit from relationships between tasks, just like multi-target prediction. The result of multi-task training is a distinct trained model $f^t(\boldsymbol{x}^t)$ for each of the T tasks.

While it is true that multi-target and multi-task learning have some common background, there are also some clear differences between them. The most obvious one is the number of trained models: a separate model for each of the tasks versus a single model trained for the entire problem. Multi-target learning aims to predict the target features and explicitly describe their relationship with the descriptive features. Moreover, it implicitly describes the relationships among the target features. The multi-task model, on the other hand, does not specifically aim to describe the relationships between the target features.

Multi-target learning implicitly captures the dependencies among the targets and represents them in the single model generated. By going through this model, we can determine the effect of the descriptive features on all the targets, and analyze the relationships, either linear or nonlinear,

between targets (or groups of targets). In case the targets are related, we can obtain information about these relationships.

To place our algorithm into a broader context, we include an up-to-date multi-task linear regression algorithm as a reference in our experiments. Most of the multi-task algorithms are originally designed for classification purposes (Chapelle et al., 2010; Rakotomamonjy et al., 2011; Parameswaran and Weinberger, 2011), but the method by Jalali et al. (2011) is readily suitable for our regression tasks. The authors try to find a compromise between selecting important weights for separate tasks and for all tasks together. That is, they are searching for both shared features and features important for each task separately. The authors do this by using both separate elementwise L1 and block L1/Lq regularization and alternate between the two during optimization. Here L1/Lq is matrix regularization, with $q > 1$ in the latter case. Because of mixing up the two "clean" regularization terms, Jalali et al. (2011) call their method "dirty", therefore we refer to their algorithm as DIRTY.

Since our method learns regression rules, it is closely related to rule learning (Flach and Lavrač, 2003). A method for learning multi-target rules has been recently developed (Ženko, 2007; Ženko and Džeroski, 2008). It employs the standard covering approach (Michalski, 1969) and can learn ordered or unordered rule sets for classification and regression domains. Its accuracy on classification domains is comparable to other classification methods, such as (multi-target) decision trees. However, on regression domains, the approach performs significantly worse than the alternatives (Ženko, 2007).

An alternative approach to rule learning is called rule ensembles (Friedman and Popescu, 2005, 2008; Dembczyński et al., 2008). Strictly speaking, any set of (unordered) rules can be called a rule ensemble, as for example, in Indurkhya and Weiss (2001). In this paper, however, a rule ensemble is understood to be a set of unordered rules whose predictions are combined through weighted voting, which is the approach introduced by the RULEFIT (Friedman and Popescu, 2005, 2008) and REGENDER methods (Dembczyński et al., 2008).

The RULEFIT algorithm starts by generating a set of decision trees in much the same way as ensembles are generated in methods like bagging (Breiman, 1996) and random forests (Breiman, 2001). Because such large ensembles are hard or even impossible to interpret, all the trees are transcribed into a collection of rules, and an optimization procedure is used to select a small subset of the rules and to determine their weights. As a result, we get a relatively small set of weighted rules combined in a linear fashion. In addition to rules, we can also use descriptive attributes in the linear combination if we add them to the initial set of rules, and likewise determine their weights in the optimization step. The final prediction for a given example is obtained by a weighted voting of all linear terms and those rules that apply (cover the example). The resulting model can thus be written as:

$$\hat{y} = f(\boldsymbol{x}) = w_0 + \sum_{i=1}^{M} w_i r_i(\boldsymbol{x}) + \underbrace{\sum_{j=1}^{K} w_{(M+j)} x_j}_{\text{optional}}, \tag{1}$$

where w_0 is the baseline prediction, the first sum is the correction value obtained from the M rules, and the second sum is the correction value obtained from the (optional) K linear terms. The rules r_i are functions, which have a value of 1 for all examples that they cover, and 0 otherwise. During the learning phase, all the weights w_i are optimized by a gradient directed optimization algorithm. The linear terms part of the model is global, that is, it covers the entire example space. Note that

this is different from model trees (Quinlan, 1992; Karalič, 1992; Wang and Witten, 1997), where we may also have local linear models in tree leaves, where each such model only applies to the specific examples covered by the leaf.

3. Learning Rule Based Ensembles for Multi-Target Regression

Our algorithm for learning rule based ensembles for multi-target regression problems (which we call FIRE: Fitted rule ensembles) is greatly influenced by the RULEFIT method. The top level of the FIRE algorithm is outlined in Algorithm 1. It starts by generating a set of diverse regression trees. To add more diversity among the base models, the trees are converted to rules. Because linear dependencies are known to be difficult to approximate with rules, we optionally add linear terms (simple linear functions) of all numeric descriptive attributes to the collection.

FIRE then optimizes the weights of rules and linear terms with a gradient directed optimization algorithm. This optimization procedure depends on a gradient threshold parameter τ; we repeat the optimization for different values of τ in order to find a set of weights with the smallest validation error. In the end, we remove all the rules and linear terms whose weights are zero.

Algorithm 1 The FIRE algorithm for learning rule ensembles for multi-target regression.

Input: training examples E
Output: rules and linear terms P with their weights W

$D \leftarrow \text{GenerateSetOfTrees}(E)$
$R \leftarrow \text{ConvertTreesToRules}(D)$
$P \leftarrow R \cup \text{LinearTerms}(E)$ {Optional}
$ERR_{\text{min}} \leftarrow \infty$
for $\tau = 1.0$ **to** 0.0 **with** *step* **do**
  $(W_\tau, ERR_\tau) \leftarrow \text{OptimizeWeights}(P, E, \tau)$
  if $ERR_\tau < ERR_{\text{min}}$ **then**
    $(W_{\text{opt}}, ERR_{\text{min}}) \leftarrow (W_\tau, ERR_\tau)$
  else if $ERR_\tau > \mathit{threshold} \cdot ERR_{\text{min}}$ **then**
    break
  end if
end for
$(P, W) \leftarrow \text{RemoveZeroWeightedTerms}(P, W_{\text{opt}})$
return (P, W)

The resulting rule ensemble is a vector function $\boldsymbol{f}$; given an unlabeled example $\boldsymbol{x}$ it predicts a vector $\hat{\boldsymbol{y}}$ consisting of the values of all target attributes:

$$\hat{\boldsymbol{y}} = \boldsymbol{f}(\boldsymbol{x}) = w_0 \boldsymbol{avg} + \sum_{i=1}^{M} w_i \boldsymbol{r}_i(\boldsymbol{x}) + \underbrace{\sum_{t=1}^{T} \sum_{j=1}^{K} w_{(t,j)} \boldsymbol{x}_{(t,j)}}_{\text{optional}} . \tag{2}$$

Note that this vector function is an extension of the scalar function (Equation 1) to the case of multi-target regression. The first term in Equation 2 includes a constant vector $\boldsymbol{avg}$, whose components are the average values for each of the targets. The first sum is the contribution of the M rules: each

rule $\boldsymbol{r}_i$ is a vector function that gives a constant prediction (for each of the targets), if it covers the example $\boldsymbol{x}$, or returns a zero vector otherwise. The double sum is the contribution of optional linear terms. There is a term for each combination of a target and a numeric descriptive attribute, thus the total number of linear terms is the number of numeric descriptive attributes K times the number of target attributes T. A linear term $\boldsymbol{x}_{(t,j)}$ is a vector that corresponds to the influence of the j-th numerical descriptive attribute x_j on the t-th target attribute; its t-th component is equal to x_j, while all other components are zero:

$$\boldsymbol{x}_{(t,j)} = (0,\ldots,\underset{t-1}{0},\underset{t}{x_j},\underset{t+1}{0},\ldots,0).$$

The values of all weights w are determined during the optimization phase, and our goal is to have as many weights equal to zero as possible.

Example 1 *Let the problem domain have eight descriptive attributes $\boldsymbol{x} = (x_1,\ldots,x_8)$ and three target attributes $\boldsymbol{y} = (y_1,y_2,y_3)$. A hypothetic rule ensemble that predicts all the target values of this domain simultaneously could be:*

$$\begin{aligned}\hat{\boldsymbol{y}} = \boldsymbol{f}(\boldsymbol{x}) = {} & 0.95\,(16.2, 6.0, 21.1) \\ & + 0.34\,[\,\mathsf{IF}\ (x_8 > 3.8)\&(x_6 > 7.2)\ \mathsf{THEN}\ (15.9, 36.2, 14.4)] \\ & + 0.21\,[\,\mathsf{IF}\ (x_3 \leq 12.1)\ \mathsf{THEN}\ (6.3, 50.0, -14.3)] \\ & + 0.80(x_2, 0, 0) + 0.11(0, 0, x_2) + 0.17(0, x_5, 0) + 0.22(0, 0, x_5) \\ = {} & (15.4 + 0.80x_2,\ 5.7 + 0.17x_5,\ 20.0 + 0.11x_2 + 0.22x_5) \\ & + [\,\mathsf{IF}\ (x_8 > 3.8)\&(x_6 > 7.2)\ \mathsf{THEN}\ (5.4, 12.3, 4.9)] \\ & + [\,\mathsf{IF}\ (x_3 \leq 12.1)\ \mathsf{THEN}\ (1.3, 10.5, -3.0)]\,.\end{aligned}$$

It comprises a constant vector, two rules and four linear terms (of attributes x_2 and x_5), but can also be simplified to a sum of a vector of linear equations and two rules.

So far, we have only briefly mentioned two important aspects of our algorithm, the generation of the initial collection of trees, rules and linear terms, and the weight optimization procedure. We describe each of them in detail in the next two subsections.

3.1 Generation of Base Models

The basic decision tree learning method used within the GenerateSetOfTrees procedure of FIRE (Algorithm 1) is the predictive clustering tree learning method (Blockeel et al., 1998) that can learn multi-target regression trees. As a starting point, we use the implementation of this paradigm within the system CLUS (Blockeel and Struyf, 2002), which can learn multi-target regression trees (Struyf and Džeroski, 2006). A set of diverse trees is generated with the multi-target implementation of the random forest ensemble method (Kocev et al., 2007), modified to stop tree building when a given tree depth limit is reached.

It is well known that variability of constituent base models is essential for good accuracy of ensembles (Dietterich, 2000). In order to increase the tree (and, thus, rule) variability, we limit the depth of a particular tree in a randomized fashion as suggested by Friedman and Popescu (2008). The maximum depth of a tree generated in the m-th iteration (denoted as d_m) is computed as follows.

Let the number of terminal nodes t_m of a tree m be a random variable $t_m = 2 + \lfloor \gamma \rfloor$, where γ is drawn from an exponential distribution

$$\Pr(\gamma) = \frac{\exp\left[-\gamma/\left(\bar{L}-2\right)\right]}{\bar{L}-2},$$

and the parameter $\bar{L}$ is the average number of terminal nodes in all trees. The depth limit for a tree m can now be computed as $d_m = \lceil \log_2(t_m) \rceil$, assuming that the root has a depth of 0. The average number of terminal nodes $\bar{L}$ of all trees is specified as a parameter to the algorithm. It should be emphasized that the parameter $\bar{L}$ only affects the *average* of all tree depth limits d_m and thus trees with larger depths can still be generated.[1]

All regression trees generated with the above procedure are transcribed into rules with the ConvertTreesToRules procedure. Each leaf of each tree is converted to a rule. The weights of these rules are later computed with gradient directed optimization, as described in Section 3.3.

However, before optimizing the rule weights, it is necessary to normalize the predictions of the rules. In order to equalize the importance of different rules and different targets we proceed in three separate steps: First, we simply zero-center all the targets. Second, we scale each rule with a factor that corresponds to the magnitude of the values predicted by the rule. This should equalize the effect of the rules on the optimization process. Third, we normalize the differing scales of target spaces away. This last step is in effect only during the optimization. The first and last steps of the process are trivial and are also repeated in most other optimization processes. The normalization process may seem overly complicated, but is necessary. In Appendix B, we describe in detail why it can not be omitted or simplified to a single scaling step. Let us now describe the three normalization steps in more detail.

In the first step, we zero-center all the rule target predictions by subtracting the average $\boldsymbol{avg}$ from each of the original rule predictions $\boldsymbol{r}''$: $\boldsymbol{r}' = \boldsymbol{r}'' - \boldsymbol{avg}$. The average $\boldsymbol{avg}$ contains the average values of the target attributes on the learning set.

In the second, more complex, step, we scale the predicted values r'_t of each target attribute t by dividing them with a factor χ:

$$r_t = \frac{r'_t}{\chi}. \tag{3}$$

We choose χ so that it is related to both the largest predicted value of the rule $\boldsymbol{r'}$ and to the standard deviation σ_t of a target attribute t. In detail, the normalization factor χ in Equation 3 is of the form

$$\chi = \frac{r'_m}{2\sigma_m}.$$

Here the target index $m \in \{1, \ldots, T\}$ of the maximum target value r'_m is defined by:

$$m = \arg\max_t \left| \frac{r'_t}{2\sigma_t} \right|.$$

In this way, we make all the rules have an equal maximal target prediction value.

1. In our preliminary experiments, the results varied only slightly with different constant depth limits. In addition, the optimal limit depended on the data set. Thus, a randomized limit seems like a natural choice. The algorithm performance was not sensitive to the distribution shape or values. Moreover, we did not use tree pruning methods, because they could reduce the diversity of the base models.

Finally, the last step is normalization, which is in effect only during the optimization. In this step, we equalize the scaling differences between different target attribute spaces. Our intention is to use this normalization only temporarily, during optimization. Otherwise, the resulting model would not be applicable to real world data anymore. As usual, we do this simply by dividing the target attribute prediction values r_t by twice their standard deviations $2\sigma_t$:

$$r_t^* = \frac{r_t}{2\sigma_t} = \frac{r_t'}{2\sigma_t\chi} = \frac{r_t'' - avg_t}{2\sigma_t\chi}.$$

We again refer to Appendix B for a detailed justification of the normalization process.

3.2 Optional Linear Terms

From Equation 2 we recall that, in addition to rules, we can also add linear terms to the rule ensemble. As already mentioned, a single linear term is defined as

$$\boldsymbol{x}''_{(t,j)} = (0, \ldots, \underset{t-1}{0}, \underset{t}{x''_j}, \underset{t+1}{0}, \ldots, 0).$$

Linear terms are normalized in a similar way as rules. We again shift the linear terms by the average $\overline{x''_j}$ of the j-th descriptive attribute $\boldsymbol{x}'_{(t,j)} = (0, \ldots, x''_j - \overline{x''_j}, \ldots, 0)$. However, we continue by normalizing the terms to the target attribute scale

$$\boldsymbol{x}_{(t,j)} = \boldsymbol{x}'_{(t,j)} \frac{\sigma_t}{\sigma_j}.$$

Linear terms normalized like this appear in the final rule ensemble model.

Analogously to the third stage of rule normalization we also scale the target dimension space out temporarily:

$$\boldsymbol{x}^*_{(t,j)} = \frac{\boldsymbol{x}_{(t,j)}}{2\sigma_t} = \frac{\boldsymbol{x}'_{(t,j)}}{2\sigma_j}.$$

This is, again, only intended to equalize the terms referring to different target attributes during the optimization procedure. See Appendix B for details.

3.3 Gradient Descent Weight Optimization

The weights from Equation 2 are determined within the OptimizeWeights procedure presented in Algorithm 2. The optimization problem that we address is typically formulated as:

$$\arg\min_{\boldsymbol{w}} \sum_{(\boldsymbol{x},\boldsymbol{y})\in E} L\left(w_0\,\boldsymbol{avg} + \sum_{i=1}^{M} w_i \boldsymbol{r}_i(\boldsymbol{x}) + \sum_{t=1}^{T}\sum_{j=1}^{K} w_{(t,j)}\boldsymbol{x}_{(t,j)}, \boldsymbol{y}\right) + \lambda \sum_{i=1}^{M} |w_i|^{\alpha}, \tag{4}$$

where L is the loss function and the last term is the regularization part. The purpose of the regularization part is to make as many weights equal to zero as possible, which means that the resulting rule ensemble will be as small as possible.

The regularization part $\sum_{i=1}^{M} |w_i|^{\alpha}$ in Equation 4 forces the weights to be smaller and adds stability to the optimization procedure. Popular values for α include $\alpha = 2$ (L2 or ridge regularization, see Vapnik, 1995) and $\alpha = 1$ (L1 or lasso regularization, see Tibshirani, 1996). The best suited

Algorithm 2 The OptimizeWeights procedure for gradient directed optimization.

Input: base models P, training examples E and gradient threshold parameter τ
Output: weights W and an error estimate ERR
Constant: the gradient step size β

$W_0 = \{0, 0, \ldots, 0\}$
$(E_t, E_v) \leftarrow \text{SplitSet}(E)$ {Training and validation}
for $i = 0$ to Maximum number of iterations **do**
  if i is multiple of 100 **then**
    $ERR_i \leftarrow \text{Error}(E_v, P, W_i)$
    if $ERR_i > threshold \cdot \min_{j<i} ERR_j$ **then**
      break
    else if $ERR_i < \min_{j<i} ERR_j$ **then**
      StoreWeights(W_i, ERR_i)
    end if
  end if
  $G \leftarrow \text{ComputeGradients}(E_t, P, W_i)$
  if Limit of allowed nonzero weights is reached **then**
    $G \leftarrow \{g_k \in G | w_k \in W_i : w_k \neq 0\}$
  end if
  $G_{\max} \leftarrow \{g_j \in G |\ |g_j| \geq \tau \max_k |g_k|\}$
  $W_{i+1} \leftarrow W_i - \beta G_{\max}$
end for
$(W, ERR) \leftarrow \text{WeightsWithSmallestError}(E_v, P)$
return (W, ERR)

value depends on the data set and the user's needs. We are interested in lasso type solutions because, as explained later, they result in models having some desired properties. Unfortunately, lasso optimization is considered to be computationally complex and thus the methods that use it have a tendency to be quite slow (Yuan et al., 2010). In our case, this is especially problematic, since the multi-target setting increases the size of the optimization problem (the number of variables) significantly. While lasso-like regularization in multi-task optimization has been under extensive research recently (Argyriou et al., 2008; Jalali et al., 2011; Rakotomamonjy et al., 2011), its computational complexity remains an issue of concern. For computational complexity reasons, we decided to use a simple but efficient optimization procedure, which performs implicit adaptive regularization. We aim for the same goals as explicit regularization, that is, minimizing the error and achieving a sparse solution, but do not include an explicit regularization term in the optimized function. We follow the approach of Friedman and Popescu (2004), where sparsity is achieved by allowing only a small number of weights (all originally set to zero) to change. Below we describe our optimization approach in detail.

Friedman and Popescu (2004) show, that for the gradient directed optimization, an effect very similar to the effect of explicit regularization can also be achieved in a different and more efficient way, without solving the optimization problem directly. They propose a gradient directed optimiza-

tion method with squared loss

$$L_{\text{sqrd}}(f_t(\boldsymbol{x}), y_t) = \frac{1}{2}(f_t(\boldsymbol{x}) - y_t)^2,$$

where $f_t(\boldsymbol{x})$ is the predicted value and y_t is the true value. The square loss function is often used for solving such optimization problems, but is only applicable to single-target problems.

If we want to use a similar gradient directed optimization algorithm for multi-target problems, we have to define a suitable loss function that is convex. A typical solution is to take the above squared loss function for each of the T target attributes and aggregate the per-target losses by taking their average:

$$L(\boldsymbol{f}(\boldsymbol{x}), \boldsymbol{y}) = \frac{1}{T}\sum_{t=1}^{T} L_{\text{sqrd}}(f_t(\boldsymbol{x}), y_t). \tag{5}$$

Such an aggregated loss function is convex and allows for efficient computation of the gradients.

Another possibility is, for example, the maximum value of the single-target loss functions. However, our preliminary experiments showed that this results in larger and less accurate models. In addition, this loss function would result in a significantly slower algorithm because, in addition to the gradients, we would also have to compute the loss function values for each target explicitly.

Instead of adding a regularization term to the optimization problem, we explicitly control the number of weights that are changed during every optimization iteration in the following way. Let M be the number of weights we are optimizing with a gradient method. Instead of allowing changes to all the weights simultaneously, we only allow changes to the weights w_j whose gradients g_j have a value above a threshold

$$|g_j| \geq \tau \cdot \max_k |g_k|.$$

If $\tau = 0$, we are changing all the weights during every iteration, resulting in a behavior similar to ridge regularization ($\alpha = 2$). On the other hand, if $\tau = 1$, only one gradient during every iteration is modified and the behavior is similar to lasso regularization ($\alpha = 1$). In our case, lasso regularization seems to be the best choice, because it has been shown to lead to many weights being set to zero (Tibshirani, 1996), which means simpler and more interpretable models with fewer rules.

In practice it is hard to know in advance which value of τ will result in the most accurate model. Both theoretical and experimental results suggest that different data sets are best suited by different regularizations (Lounici et al., 2009; Rakotomamonjy et al., 2011). Thus, the most suitable value of τ depends on the properties of the learning data. We overcome this problem by trying a set of different values of τ (Algorithm 1, line 5) and estimating their accuracies on a separate internal validation set E_v (Algorithm 2, line 19), which is the same for all τ values. In the end, the model with the smallest validation error is selected.

Our aim is to efficiently learn a rule ensemble model that is both small and accurate. Therefore, we start with a τ value that creates a small model, $\tau = 1$, and then iteratively decrease the value of τ until it reaches zero. We stop the loop if the validation error stops decreasing, since it is unlikely that trying smaller values would result in a more accurate model.

It is possible that we are stopping in only a local optimum and the result can be a suboptimal model. However, in practice, not evaluating the lower τ values does not seem to lower the accuracy significantly. Also, this procedure is very effective, because most of the optimization time, namely $|E|(M+TK)^2/2$, is spent on computing the covariance matrix of weights. Here $|E|$ is the learning set size and $M+TK$ the number of optimized weights. Nevertheless, we do not have to compute

the covariances for zero weighted predictive terms. Thus, most of the resources are usually used for optimization with lower values of τ. For example, in practice, the case $\tau = 0$ seems to use at least half of the computing time alone.

The complete optimization procedure OptimizeWeights is presented in Algorithm 2. It starts by initializing all the weights to zero and splitting the entire learning set E into a learning set E_t and an internal validation set E_v. Within the loop, we iteratively compute the gradients g_k for each of the weights (line 12) and then change the selected weights w_j in the most promising direction $-G_{\max}$ for a predefined step size β (line 17). The step size is an automatically computed constant, which is based on the theoretically optimal step. See Appendix A for a more detailed description of the step size computation.

In addition to this basic idea, there are some additional details. First, on every 100-th iteration we stop the optimization if we are overfitting (lines 6–7), that is, if the validation error starts to increase. Second, we can define a maximum number of nonzero weights in advance (lines 13–15), which makes a suitable parameter for setting the accuracy-for-simplicity trade-off. An extensive experimental evaluation of the algorithm's performance is presented in the next two sections.

4. Experimental Setup

In the experimental evaluation, we investigate three different issues. First, we evaluate our algorithm FIRE on single-target regression domains in order to show that our algorithm is also applicable to standard regression problems and that its performance on such problems is comparable to the performance of other tree and rule based regression methods. We compare FIRE with regression trees (Breiman et al., 1984), random forests (Breiman, 2001), model trees (Quinlan, 1992; Karalič, 1992), the L1/Lq regularized multi-task regression algorithm DIRTY (Jalali et al., 2011), the rule ensemble methods RULEFIT (Friedman and Popescu, 2005, 2008) and REGENDER (Dembczyński et al., 2008). In the comparison, we focus on the accuracy and size of the learned models. The model size is used to indicate the interpretability of the model.

Second, we evaluate FIRE on multi-target domains, that is, on the problems for which it was designed in the first place. We compare it with three other multi-target variants of popular tree based methods: regression trees (Blockeel et al., 1998), random forests (Kocev et al., 2007), and model trees (Appice and Džeroski, 2007). In addition, we use the L1/Lq regularized multi-task regression algorithm DIRTY (Jalali et al., 2011) for a reference. This is the main part of the evaluation since it was designed to show how effective FIRE is when compared with other state-of-the-art multi-target prediction methods. Again, we focus on the accuracy and size of the learned models.

As described in Section 3.3, FIRE has a parameter that can be used to limit the total number of nonzero weights, that is, the number of rules and linear terms. We use this parameter in the third part of the evaluation to investigate how the size of the rule ensemble influences its accuracy. The preliminary experiments showed that, in some cases, we can significantly reduce the model size with only a marginal decrease in accuracy. This is the reason that in both above mentioned evaluations we also include results for FIRE models with an arbitrary limit of 30 rules and terms (denoted as "Max 30" in the results). Another optional setting of the FIRE (and RULEFIT) algorithm(s) is whether to include linear terms in the model or not. We present both cases in all evaluation scenarios; models with linear terms are denoted as "+ linear". In the remainder of this section, we present the algorithms and their parameter settings, evaluation methodology and data sets used in the empirical evaluation.

4.1 Algorithms and Parameters

When generating the initial set of trees (Algorithm 1, line 1) we use 100 random trees with an average depth of 3. The optimization procedure is run with the gradient threshold parameter τ values ranging from 1 to 0 in 0.1 decrements (Algorithm 1, line 5). In the OptimizeWeights procedure, the initial set E is split into $2/3$ for training E_t and $1/3$ for validation E_v (Algorithm 2, line 2). The maximum number of optimization iterations is 10,000. The threshold for detecting the error increase (Algorithm 1, line 9 and Algorithm 2, line 6) is 1.1, the step size β (Algorithm 2, line 17) is computed automatically based on the optimal step size as described in Appendix A. Our algorithm, as well as the regression trees (Blockeel et al., 1998) and random forests (Kocev et al., 2007) used in our experiments are implemented in the CLUS predictive clustering system (Blockeel and Struyf, 2002).[2] All the parameters for regression trees and random forests are set to their default values; regression trees use reduced error pruning, random forests consist of 100 trees.

For experiments with model trees, we use the multi-target implementation MTSMOTI by Appice and Džeroski (2007) with the recommended settings: First we set the stopping criterion so that a non-leaf node covers at least a tenth of the training set. Also for continuous attributes all the distinct values are treated as candidate split thresholds for node tests. However, a value is not a candidate threshold if it causes either child to cover less than 15 examples. Finally, the maximum number of variables included in the regression model of a pruned leaf is set to 10. We report the pruned trees. The implementation of MTSMOTI does not handle missing values, so we pre-process the data by replacing them with averages or the most common labels.

Experiments with the rule ensemble methods RULEFIT and REGENDER were performed with the original implementations by the authors and the settings that lead to the best performance in the original papers (Friedman and Popescu, 2008, 2004; Dembczyński et al., 2008). For RULEFIT, we use the Huber distance with a trimming factor of 0.9. The linear variable conditioning trimming factor is set to 0.025, the average number of tree terminal nodes to 6, the maximum number of rules to 5,000, the incentive factor for using fewer variables in tree based rules to the default value 3.0, the model memory parameter value to 0.01, and the sampling fraction for the trees to $|E|/5$. The regularization parameter τ is chosen with full *internal* cross-validation. Hence, the used parameter τ value should be the best one, but may be slow to find. The maximum step size for the gradient descent algorithm is set to the default of 0.01. The convergence factor is set to 1.1. For a detailed analysis of the differences in the settings of RULEFIT and FIRE see Appendix C. The number of rules for REGENDER is set to 200, the shrinkage amount to 0.5, the data is resampled without replacement and missing values are not replaced, which is the default setting. Moreover, the minimization technique is set to gradient descent with squared error loss.

For the multi-task learning algorithm DIRTY (Jalali et al., 2011) we slightly modified the original R code by the authors. Recall that the multi-task problem domain consists of multiple single-target learning tasks in the separate training sets E_t: $\{(\boldsymbol{x},y) \in E_t | t = 1 \ldots T\}$. In our multi-target data sets, however, the descriptive parts $\boldsymbol{x}$ of example sets are all the same: there is a single example $(\boldsymbol{x'}, (y'_t)_{t=1}^T) = (\boldsymbol{x}, \boldsymbol{y'}) \in E$ in the multi-target setting corresponding to a collection of instances $\{(\boldsymbol{x'}, y'_t) \in E_t | t = 1 \ldots T\}$ in the multi-task setting. In practice, we modified the code of DIRTY so that it uses the same descriptive features $\boldsymbol{x}$ for all the tasks.

Otherwise we used the parameter values and methodology suggested by Jalali et al. (2011, Appendix H) and by Ali Jalali in personal communication: For the optimization stopping crite-

2. Available at `http://clus.sourceforge.net` under the GNU General Public License.

rion we set $\varepsilon = 10^{-10}$. As in RULEFIT, we used internal 10-fold cross-validation for selecting the best values for the regularization weights λ_b and λ_b. As suggested by Jalali et al. (2011, Appendix H) we used the resulting matrices B and S of previous λ combination as the initial weight matrices for the next combination. The tried λ values were of the form $c\sqrt{\log(TK)/|E|}$, where K is the number of descriptive attributes. For λ_b, the constant c has seven values $c_b \in \{0.001, 0.005, 0.01, 0.05, 0.1, 0.5, 1\}$ and for λ_s similarly $c_s = c_b\sqrt{T}$ for each value of c_b. Thus, we have 49 λ value pairs to try out. Finally, the descriptive features were normalized by scaling with the maximum feature norm of the whole set: $\max_{j=1\ldots K}\sqrt{\sum_{(\boldsymbol{x},\boldsymbol{y})\in E} x_j^2}$, where $\boldsymbol{x} = (x_1, x_2, \ldots, x_K)$. Thus, each descriptive feature is shrank to the interval $[-1, 1]$.

4.2 Data Sets and Evaluation Methodology

The data sets used in the experiments, together with their properties are presented in Table 1. Twenty-four single-target regression data sets are taken from the following data repositories: UCI (Asuncion and Newman, 2011), Weka (2011), StatLib (2011), Delve (2011), and Torgo (2011). Publicly available multi-target data sets, however, are scarce. In addition to a single public data set from the UCI repository, we have collected ten previously analyzed data sets with the following references: DS1 (Kampichler et al., 2000), DS2 (Karalič and Bratko, 1997), DS3 (Džeroski et al., 2006), DS4 (Stojanova, 2009), DS5 (Džeroski et al., 2002), DS6 (Demšar et al., 2006), DS7 (Demšar et al., 2005), DS8 (Džeroski et al., 2005), DS9 (Gjorgjioski et al., 2008), and DS10 (Džeroski et al., 2000).

The accuracy of the learned regression models is estimated for each target attribute by computing the relative root mean squared error (RRMSE). For a single-target model $f(\boldsymbol{x})$ and an example set E, RRMSE is computed as

$$\mathrm{RRMSE}(f, E) = \sqrt{\frac{\sum_{(\boldsymbol{x},y)\in E}\left(f(\boldsymbol{x}) - y\right)^2}{\sum_{(\boldsymbol{x},y)\in E}\left(\bar{y} - y\right)^2}},$$

where $\bar{y}$ is the mean value of target attribute y over data set E. The size of tree based models (regression trees, random forests, and MTSMOTI) is measured as the number of leaves in all the trees. The size of rule ensemble models (FIRE, RULEFIT, and REGENDER) is measured as the number of rules or the sum of the number of rules and linear terms, if linear terms are used.[3] For DIRTY, we present the number of nonzero weights (weight matrix support) as was done by Jalali et al. (2011). All the above measures are estimated using 10-fold cross-validation, where the folds for each data set are the same for all the algorithms.

To test whether any of the observed differences in accuracy and size between the algorithms are significant, we followed the methodology suggested by Demšar (2006). First, we use the Friedman test to check if there are any statistically significant differences between the compared algorithms. If the answer is positive, we additionally use the Nemenyi post-hoc test to find out what these differences are, and we present them by average ranks diagrams. These diagrams show all the compared algorithms in the order of their average ranks; the best are on the right and the worst are on the left side of the diagram. The algorithms that differ by less than a critical distance for a

3. These measurements were chosen because they relate quite naturally to the number of linear terms. For example, with the total number of tests or conditions in trees and rules this would have been problematic. However, the results were similar with both choices.

DATA SET	# EXS	% MISS VALS	# NOM ATTS	# NUM ATTS	# TAR ATTS	# ALL ATTS	SOURCE
ABALONE	4,177	0.0	1	7	1	9	UCI
AILERONS	1,533	0.0	0	40	1	41	TORGO
AUTO-MPG	398	0.2	3	4	1	8	UCI
AUTO-PRICE	159	0.0	0	15	1	16	WEKA
BREAST CANCER	286	0.4	8	1	1	10	WEKA
CENSUS	22,784	0.0	0	8	1	9	DELVE
CLOUD	108	0.0	2	4	1	7	UCI
CPU ACTIVITY	8,192	0.0	0	12	1	13	DELVE
COMPUTER HW	209	0.0	1	6	1	8	UCI
DELTA-AILERONS	7,129	0.0	0	5	1	6	TORGO
DIABETES	43	0.0	0	2	1	3	TORGO
ECHOCARDIOGRAM	130	8.3	3	6	1	10	WEKA
HOUSING	506	0.0	1	12	1	14	UCI
HOUSING CA	20,640	0.0	0	8	1	9	STATLIB
KINEMATICS	8,192	0.0	0	8	1	9	DELVE
META-DATA	528	4.6	2	19	1	22	UCI
PBC	418	16.5	8	10	1	19	UCI
POLE TELECOMM	15,000	0.0	0	48	1	49	TORGO
PYRIMIDINES	74	0.0	0	27	1	28	TORGO
QUAKE	2,178	0.0	0	3	1	4	UCI
SENSORY	576	0.0	11	0	1	12	UCI
SERVO	167	0.0	4	0	1	5	UCI
STRIKE	625	0.0	1	5	1	7	UCI
VETERAN	137	0.0	4	3	1	8	UCI
COLLEMBOLAN	393	20.4	8	40	3	51	DS1
EDM	154	0.0	0	16	2	18	DS2
FOREST KRAS	60,607	0.0	0	160	11	171	DS3
FOREST SLIVNICA	6,219	0.0	0	149	2	151	DS4
META LEARNING	42	27.9	0	56	10	66	DS5
MICROARTHROPODS	1,944	0.1	0	142	3	145	DS6
SIGMEA REAL	817	0.0	0	4	2	6	DS7
SIGMEA SIMULATED	10,368	0.0	2	8	3	13	DS8
SOLAR FLARE	323	0.0	10	0	3	13	UCI
VEGETATION	29,679	0.0	0	64	11	75	DS9
WATER QUALITY	1,060	0.0	0	16	14	30	DS10

Table 1: Data sets used in the experimental evaluation and their properties. Please see the text for source references.

p-value of 0.05 are connected with a horizontal bar and are not significantly different. We perform such significance testing for both the RRMSE and model size metrics.

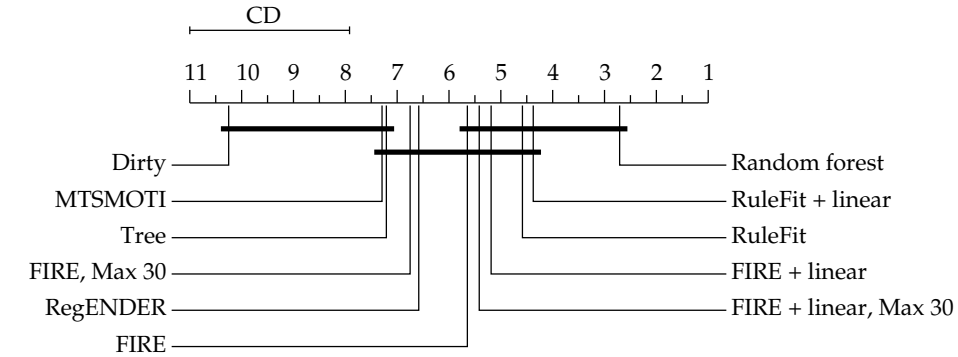
However, when testing the differences in RRMSE for multi-target data, we have two possibilities. On one hand, we can treat each of the target attributes as an independent measurement. The argument against this option is that target attributes within one data set are probably not independent and as a result our test will show more significant differences than there actually are. On the other

hand, we can compute the average over all targets within each data set and consider such averages as independent measurements. The argument against the second option is that when computing averages across all target attributes within a data set, we are actually "summing apples and oranges," and the resulting average is probably not a valid quantity. In the absence of a better solution, we present the tests of RRMSE differences for both options. The results of the experimental evaluation are presented in the next section.

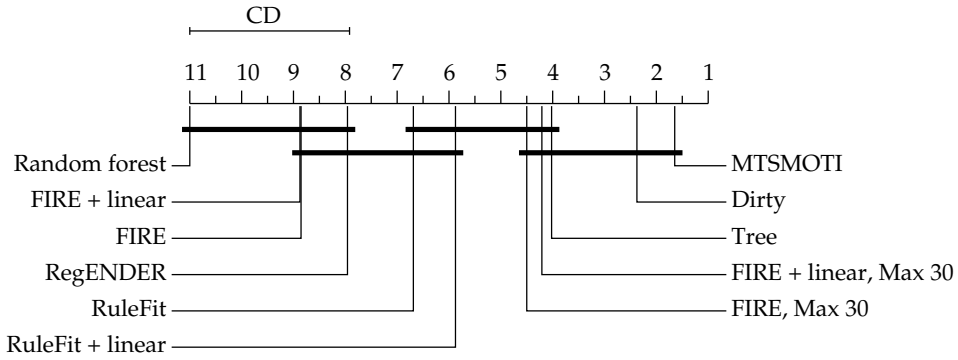
5. Results

As already mentioned, we perform three groups of experiments. We evaluate FIRE first briefly on single-target and then extensively on multi-target regression data sets. The latter is the most important part, since it shows whether our generalization of rule ensembles towards multi-target regression is successful. Next, we investigate the influence of the size of a rule ensemble on its accuracy. Finally, we present the running times for all experiments.

5.1 Single-Target Regression



(a) RRMSE



(b) model size

Figure 1: Average ranks diagrams on *single-target* data for *RRMSE* (a) and *model size* (b). Better algorithms are on the right-hand side, the ones that do not differ significantly by the Nemenyi test (p-value $= 0.05$) are connected with a horizontal bar. CD is the critical distance.

On single-target data sets, we compare regression trees, random forests, linear regression (DIRTY), model trees (MTSMOTI), and rule ensembles: four versions of FIRE, two versions of RULEFIT, and REGENDER. For FIRE and RULEFIT, we include the experiments with and without linear terms. Moreover, for FIRE we include experiments without any limitation on the model size, and experiments with the maximum number of rules and linear terms set to 30.

The Friedman test shows that the RRMSEs are statistically different with a p-value $= 6.2 \cdot 10^{-14}$ and model sizes with a p-value $< 2.2 \cdot 10^{-16}$. The average ranks for all algorithms together with the results of the Nemenyi test are given in Figure 1, separately for RRMSE and model size. The better algorithms are the ones with higher ranks (with 1 being the highest rank) and are placed on the right-hand side of the diagram. Algorithms whose ranks differ by less than a critical distance (CD) are not statistically significantly different in performance with a p-value $= 0.05$.

From the RRMSE diagram (Figure 1a), we can see that random forests are the most accurate method, followed by all the rule ensemble methods, MTSMOTI, regression trees and finally DIRTY, which seems to perform poorly on single-target data. However, there are not many statistically significant differences: Only random forests and DIRTY seem to be significantly different from part of the main group. Random forests are better and DIRTY clearly worse. The statistical similarity is somewhat surprising, considering that we have as many as 24 data sets. An intuitive reason for this is, naturally, the homogeneous background of many of the algorithms.

Our unlimited FIRE versions and limited FIRE with linear terms are in the middle class. However, they are slightly outperformed by RULEFIT, which is very closely related to FIRE. Not surprisingly, further experiments presented in Appendix C show that these differences are due to the RULEFIT features not implemented in FIRE. We also notice that linear terms tend to increase the accuracy of both versions of FIRE and RULEFIT. In particular, adding linear terms still yields accurate models even when we limit the number of FIRE rules. In this case, linear terms seem to be surprisingly effective, as they bring the limited FIRE of 30 terms to the same accuracy level with much larger unlimited FIRE set of rules.

The diagram for model size (Figure 1b) shows that MTSMOTI and DIRTY create the smallest models, while random forests are at the other extreme. Both unlimited FIRE versions seem to generate models that are larger than the reference rule ensembles (RULEFIT and REGENDER), but the differences are not significant. While the unlimited version of FIRE generates smaller models than random forests, the difference in size is below the significance threshold. Also the sizes of limited FIRE versions and individual regression trees seem to be very similar according to this statistical test.

The detailed results are shown in Table 4 in Appendix D. Note the large size of random forest models in comparison with all other algorithms. On average, the model size is 120,865 terminal nodes. The accuracy of random forests clearly comes with the price of increased model size. The detailed results also shed some light to the performance of limited FIRE with linear terms against unlimited FIRE without them. In a pairwise comparison, the algorithm tie in the wins over data sets (12 wins each) and the unlimited version has a slightly lower average accuracy. Thus, the result can not be explained only with the nature of Nemenyi test lacking pairwise comparison. The two algorithms really, surprisingly, seem to perform equally well while the unlimited FIRE creates much larger models.

In sum, the results of this first part of the evaluation show that our rule ensemble method FIRE performs well on single-target regression problems.

5.2 Multi-Target Regression

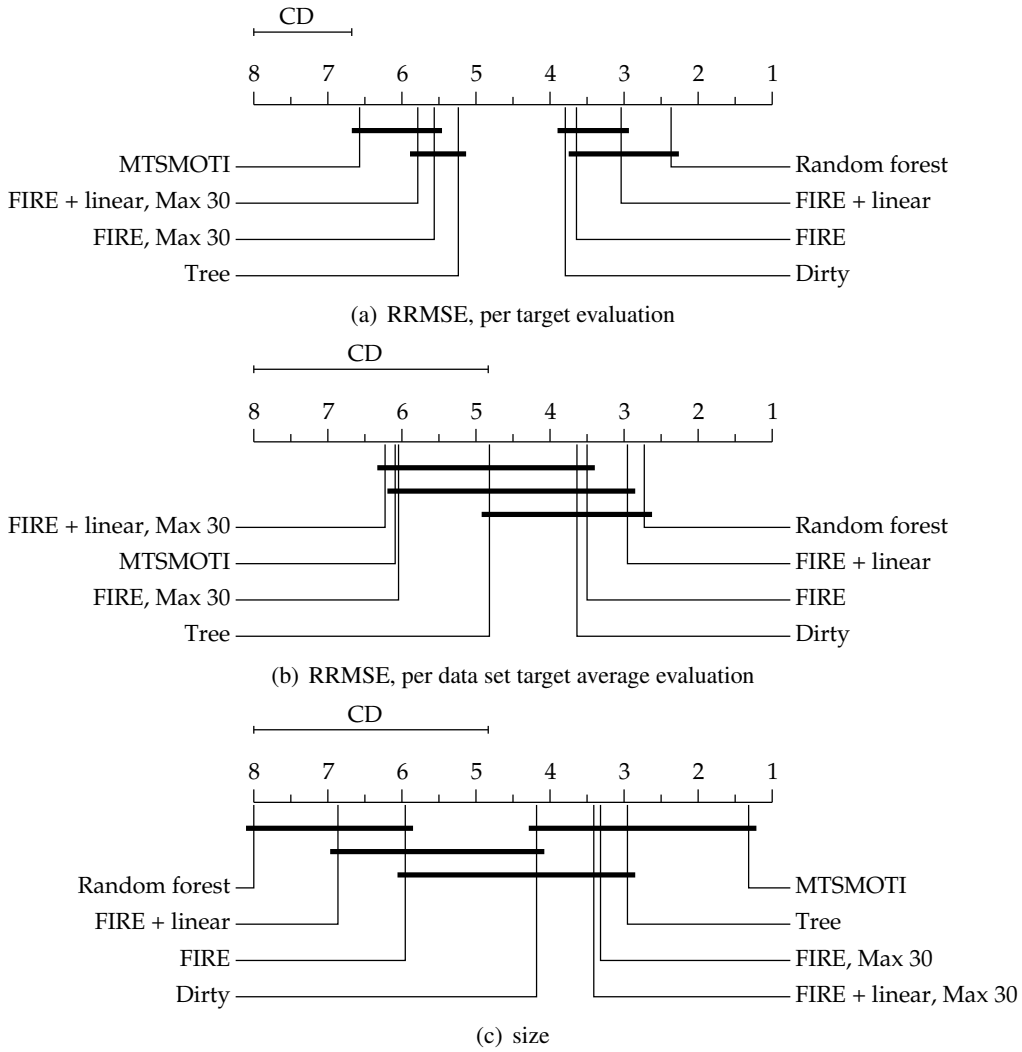


Figure 2: Average ranks diagrams on *multi-target* data for *RRMSE evaluated on separate targets* (a), *on target averages within data sets* (b) and *model size* (c). Better algorithms are on the right-hand side, the ones that do not differ significantly by the Nemenyi test (p-value $=$ 0.05) are connected with a horizontal bar. CD is the critical distance.

In the multi-target experiments, we use all the algorithms from the previous section that work on multi-target data sets. These are regression trees, random forests, MTSMOTI, DIRTY, and the same four versions of FIRE.

For multi-target data, the Friedman test shows that the RRMSE values of algorithms are significantly different with a p-value $< 2.2 \cdot 10^{-16}$, if we treat each target separately, and with a p-value $= 2.1 \cdot 10^{-4}$, if we compare target averages over each data set. The model sizes are different with a p-value $= 1.2 \cdot 10^{-11}$. The average ranks and results of the Nemenyi test are given in

Figure 2 for RRMSE evaluated on separate targets (a), RRMSE evaluated on target averages within data sets (b), and model size (c).

Looking at Figure 2(a), the general picture of algorithm ranking is similar to the one for single-target prediction: random forests and both unlimited FIRE versions are more accurate than DIRTY, regression trees, the limited versions of FIRE, and MTSMOTI. Due to a smaller critical distance, the four more accurate algorithms are significantly better than the rest. Evaluation over target averages within data sets (b) shows a similar picture, but because the sample size is smaller (11 data sets vs. 63 targets), the critical distance is larger and there are very few significant differences. The diagram for size (c) is very similar to the single-target case: DIRTY, the limited FIRE, regression trees, and MTSMOTI are significantly smaller than random forests.

In general, in the ranking of the algorithms according to their accuracy, the largest change from the single-target to multi-target setting is that DIRTY now performs practically as well as the unlimited FIRE without linear terms. Clearly, the moderately complex regularization of DIRTY does not pay off for single-target prediction, but does pay off for multi-target prediction. Moreover, as can be seen in the detailed results in Table 5 in Appendix D, the accuracy of DIRTY is sometimes remarkably high in comparison to the accuracies of the other algorithms. However, the performance is quite unstable and the algorithm performs poorly on several data sets. Apart from DIRTY, the results follow the single-target case in a straightforward manner: larger and more complex models are more accurate. In this sense, DIRTY is more or less an outlier—the model itself is simple but the creation process is complicated. From this point of view, its behavior being unstable but sometimes resulting in a very accurate model is intuitively understandable.

In addition, the linear terms do not seem to be as useful for the limited version of FIRE as for the single-target prediction. Moreover, the limited FIRE version performs, rather surprisingly, relatively much worse on multi-target data. Specifically, we notice this in comparison with regression trees, which were one of the least accurate models for single-target data, but they are in the middle class now. This is especially clear in per data set average evaluation and occurs in the pairwise comparison with the limited FIRE (in per target evaluation 7 out of 11 wins and per data set average 34–36 out of 63 wins for regression trees). The detailed results in Appendix D show that regression trees tend to win only when the size of the tree produced is much larger than the FIRE limit of 30. This suggests that the limit is too strict for good accuracy in general. Additional examination shows that we need a size limit of 60–80 for FIRE to overtake regression trees in terms of accuracy. The limit is, thus, much higher than the one needed in single-target case.

Surprising results are also achieved with MTSMOTI, which seems to underachieve, given the complexity of the model and the process of its induction. When we compare the results with the ones introduced in the original article (Appice and Džeroski, 2007), we notice some differences. Especially the common reference algorithm, multi-target regression tree, has radically improved results in our experiments. After studying the issue, it seems that the implementation details of the multi-target regression trees in CLUS have been modified causing the change in performance. Thus, the reason for the lower relative performance of MTSMOTI is partly due to the change of the performance of the most important reference algorithm. In our experiments, MTSMOTI model sizes are similar to those reported in the original paper. The slight changes must be due to the differences in data preprocessing and parameter values.

There are some additional interesting points in the detailed results. We note that while the difference in size between random forests and the unlimited FIRE versions is not statistically significant, the average size of a random forest (276,075 terminal nodes) is more than 300 times larger than the

average sizes of FIRE models. At the same time, the difference in average accuracy is small. When both size and accuracy are taken into account, the unlimited version of FIRE with linear terms seems to perform very well on multi-target regression problems.

We also notice that the average proportion of linear terms in the unlimited model is 24%, but drops to only 5% if we limit the model size to 30. This, and more generally the effect of linear terms on FIRE, is analyzed more in detail in Section 5.5. Nevertheless, it is also interesting that the proportion of linear terms highly depends on the data set. This suggests that for some data sets, using only rules is not enough for high accuracy. For example, FOREST SLIVNICA seem to be quite linear in nature, because MTSMOTI, and DIRTY achieve very good accuracy with moderately small models. On these data sets, the unlimited FIRE version with linear terms also seems to be better than the one without. Both MTSMOTI and DIRTY, however, seem to perform much worse on some other data sets (such as EDM).

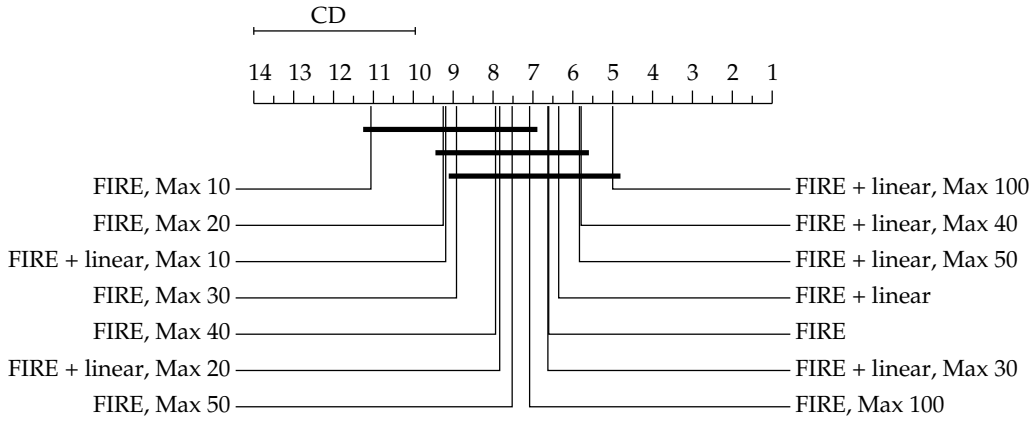
5.3 Model Size Limitation

Experiments presented in the previous two subsections considered two size-related options for the FIRE algorithm, one with the maximum model size set to 30, and one without any model size restrictions. In this subsection, we present experiments with different values of the maximum model size parameter, which show how this model size limit influences the accuracy of models. We use the values of 10, 20, 30, 40, 50, 100, and ∞.

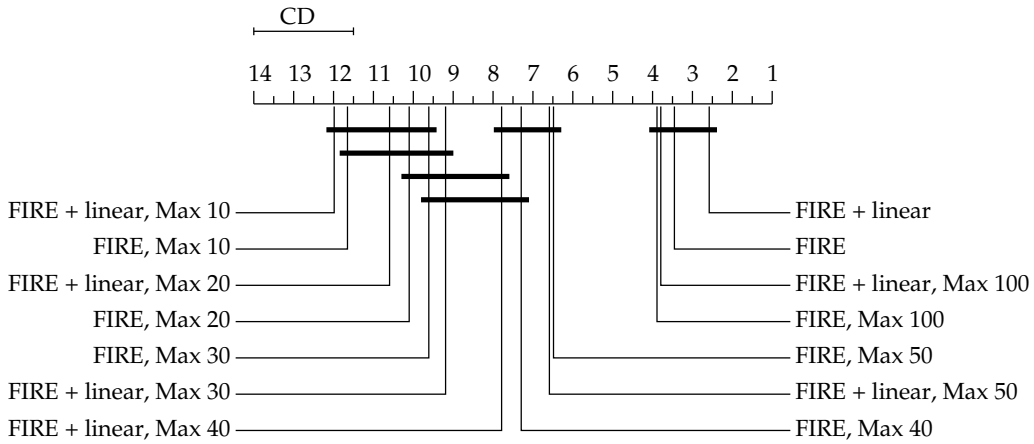
The average ranks diagrams comparing different maximum model size parameters are presented in Figure 3. Diagram (a) shows the results on single-target data. While all the differences in RRMSE are not significant, it is clear that increasing the model size improves the accuracy. Diagrams (b) and (c) show the RRMSE on multi-target data for per-target evaluation and for per-data set target average evaluation, respectively. Because of a larger sample, there are more significant differences in (b) than in (c). However, what is common to both diagrams is the trend that smaller models are also less accurate. The size limitation parameter can therefore be used as an accuracy-for-simplicity (and interpretability) trade-off setting.

Another interesting conclusion that can be drawn from these diagrams is that while a model size of 30 seems enough to get models that are not significantly less accurate than the unlimited models for single-target domains, this is not the case for multi-target domains. This clearly depends on the domain, the number of target attributes, and relations between them. The task of modeling a multi-target domain is clearly harder than the task of modeling a single-target domain, and therefore the corresponding models have to be more complex.

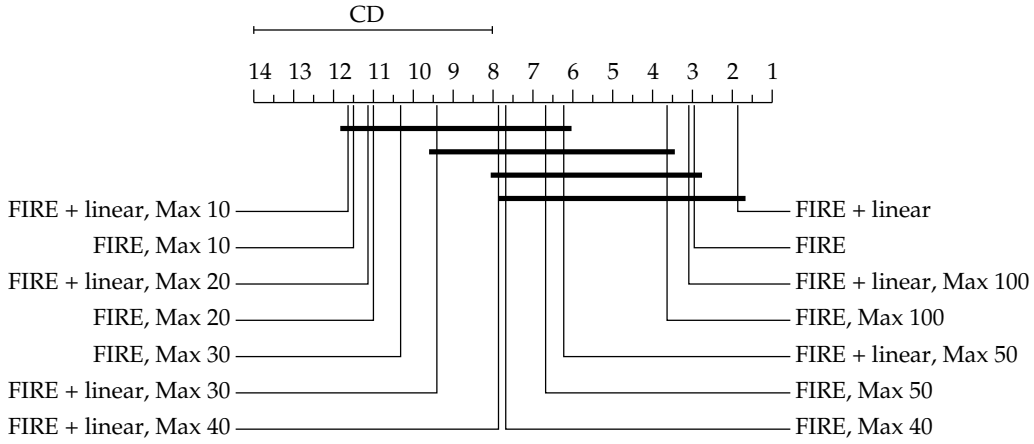
We also note that even if on single-target data linear terms always increase the accuracy of FIRE substantially, the effect is more complicated in the multi-target case. Surprisingly, for smaller size limit values (less than 30 terms and rules) the linear terms seem to decrease the accuracy. Moreover, even for larger FIRE models the effect is next to negligible. The sole exception in this is the unlimited version which clearly, although not significantly, benefits from the linear terms. We could, thus, conclude that in the multi-target case any size limit removes the gain from linear terms. On the other hand, for single-target case linear terms are always useful. We again refer to Section 5.5 for a more detailed analysis on linear terms.



(a) single-target problems



(b) multi-target problems, per target evaluation



(c) multi-target problems, per data set target average evaluation

Figure 3: The average ranks diagrams of FIRE with different model size limitations for *RRMSE* on *single-target data* (a), on *multi-target data evaluated on separate targets* (b), and on *multi-target data evaluated on target averages within data sets* (c). Better algorithms are on the right side, the ones that do not differ significantly by the Nemenyi test (p-value = 0.05) are connected with a horizontal bar. CD is the critical distance.

5.4 Running Times

The running times were measured with the GNU/Linux `time` command. The underlying environment was 64 bit Red Hat Linux with 4 processors of type 2.216 GHz Dual-Core AMD Opteron, and 16 GB of memory. We ran the experiments 5 times, omitted the highest and lowest times and took the average over the three remaining measurements. The regression trees, random forests and FIRE are implemented in Java within the CLUS machine learning toolbox (Blockeel and Struyf, 2002). The REGENDER and MTSMOTI algorithms are also implemented in Java, while RULEFIT and DIRTY are implemented in the R statistical language. For RULEFIT, the critical parts are implemented in a binary library. For the unlimited FIRE version using linear terms, we also tried to partly optimize the implementation: we implemented the most time critical part, that is, the optimization part (Algorithm 2), as a C++ dynamic library for Java.

The results are shown in Table 2. The overall trend is that more accurate models take more time to be generated. The FIRE, RULEFIT and DIRTY methods seem to be more time consuming. For DIRTY, the data sets with multiple targets require heavier computing, while for a small number of targets the algorithm is quite fast.

Adding linear terms in FIRE further increases the average time usage. However, the effect depends on the data set. On some data sets, like SIGMEA REAL and CPU ACTIVITY, linear terms increase the accuracy without increasing the running time.

Usually, however, adding more base models (rules and linear terms) to an ensemble also increases the amount of computing needed. As mentioned in Section 3, the total number of added linear terms is the number of numeric descriptive attributes times the number of target attributes. This is probably a reason for the long running time of FIRE with linear terms and why the difference between the two unlimited FIRE versions is greater for multi-target data sets than for single-target ones. The large difference between the times of the unlimited versions of FIRE for data sets like META LEARNING and AUTO-PRICE is caused by trying a different number of τ parameter values for optimization (Algorithm 1, lines 9–10).

The limited FIRE versions use on average about a tenth of the time used by the unlimited ones. The main reason for the decrease is that only a small portion of the covariance matrix is computed, as discussed in Section 3.3. Moreover, only a small number of weights is changed during each iteration.

By optimizing the implementation, we can surely decrease the time usage of FIRE: The unlimited version with linear terms was one fourth faster even with the suboptimal Java/C++ library solution shown in the rightmost column of Table 2. Nevertheless, as illustrated by the usually at most moderate difference between FIRE and the probably much more optimized RULEFIT implementation on single-target problems, there might not be a lot of room for further optimization. Nevertheless, we should be able to speed up FIRE by limiting the number of possibilities covered. For example, we could try out a smaller number of τ values (Algorithm 1, line 5).

DATA SET	TREE	RANDOM FOREST	FIRE	FIRE + LINEAR	FIRE, MAX 30	FIRE + LINEAR, MAX 30	MTSMOTI	DIRTY	RULEFIT	RULEFIT + LINEAR	REGENDER	FIRE + LINEAR (C++)
ABALONE	3.3	24.3	300.3	294.9	12.4	38.1	6.0	21.9	146.6	140.0	8.2	
AILERONS	2.9	10.1	258.4	315.4	5.6	13.0	15.9	157.6	52.0	52.7	15.2	
AUTO-MPG	2.1	4.5	301.3	336.6	4.2	5.4	1.2	7.3	12.2	10.4	1.5	
AUTO-PRICE	1.7	2.9	92.5	5.6	3.2	3.2	1.2	42.6	3.8	3.6	0.8	
BREAST CANCER	1.9	3.8	44.8	49.6	17.7	17.3	0.7	14.4	5.9	5.8	1.5	
CENSUS	9.8	185.7	531.0	551.1	43.5	47.4	44.0	84.0	1,131.7	1,133.1	108.5	
CLOUD	1.6	2.8	3.8	138.4	2.7	15.3	0.6	13.0	2.5	2.1	0.6	
CPU ACTIVITY	5.0	50.0	289.5	292.7	14.0	26.2	22.0	76.2	707.1	682.0	50.5	
COMPUTER HW	2.0	3.3	56.0	5.6	2.7	3.9	1.1	13.1	6.0	5.6	1.0	
DELTA-AILERONS	3.5	37.4	269.2	295.4	16.2	21.7	5.0	15.0	247.0	230.5	10.4	
DIABETES	1.4	2.1	33.1	33.4	2.7	2.8	0.3	2.5	0.8	0.7	0.5	
ECHOCARDIOGRAM	1.7	3.1	23.1	24.2	3.5	3.6	0.5	18.1	3.5	3.4	0.6	
HOUSING	2.2	4.9	267.1	316.7	4.4	5.2	2.5	33.5	14.7	14.7	1.4	
HOUSING CA	10.3	162.1	428.1	351.0	36.4	63.2	353.1	77.0	1,875.1	1,814.4	83.5	
KINEMATICS	4.5	56.3	405.1	423.4	14.9	20.0	16.2	37.0	720.8	742.7	23.1	
META-DATA	2.5	3.9	3.9	3.8	4.3	4.0	2.5	33.9	10.6	10.7	4.3	
PBC	2.2	4.9	54.8	64.7	7.7	9.1	2.4	28.3	13.3	13.4	1.4	
POLE TELECOMM	8.6	50.4	353.8	463.7	31.8	48.0	59.0	1,386.5	488.5	496.3	423.3	
PYRIMIDINES	1.7	2.3	2.8	5.0	2.7	3.4	1.0	64.5	1.1	1.0	0.7	
QUAKE	2.4	12.9	119.0	131.1	18.1	18.8	2.1	4.7	53.8	53.9	1.5	
SENSORY	2.1	4.7	31.2	31.3	6.3	5.9	1.1	13.1	28.2	28.2	1.6	
SERVO	1.6	2.9	32.1	30.4	2.9	2.8	0.4	4.5	4.9	4.9	0.7	
STRIKE	2.2	5.6	16.9	16.1	16.9	16.9	1.2	8.4	16.0	16.1	2.1	
VETERAN	1.5	2.8	156.1	172.9	13.7	17.0	0.5	9.3	2.6	2.6	0.6	
AVERAGE – SINGLE TARGET	3.3	26.8	169.7	181.4	12.0	17.2	22.5	90.3	231.2	227.9	31.0	
COLLEMBOLAN	1.9	6.1	92.1	162.6	6.6	7.7	5.0	367.0				91.0
EDM	1.3	2.0	43.8	58.2	3.3	3.9	2.2	70.2				40.4
FOREST KRAS	176.0	937.0	3,851.8	20,833.4	671.6	1,741.2	2,363.3	615,376.3				17,204.1
FOREST SLIVNICA	9.8	56.9	371.7	1,047.7	19.3	26.1	222.6	11,365.3				479.9
META LEARNING	1.5	2.2	50.9	698.5	13.4	17.2	1.9	1,218.9				293.6
MICROARTHROPODS	2.6	5.7	194.6	745.1	20.5	31.6	37.8	6,196.6				362.8
SIGMEA REAL	1.5	5.4	28.9	30.9	2.9	3.6	2.0	7.7				15.0
SIGMEA SIMULATED	4.8	25.0	257.4	291.9	16.4	18.4	52.3	150.1				199.7
SOLAR FLARE	1.3	2.4	10.0	9.9	13.3	12.7	0.8	25.3				6.9
VEGETATION	38.4	497.1	1,758.2	6,425.3	143.5	362.5	1,002.5	50,165.2				4,823.9
WATER QUALITY	2.2	14.2	437.6	829.6	35.7	38.9	7.8	294.4				467.8
AVERAGE – MULTI-TARGET	22.0	141.3	645.2	2,910.5	86.0	205.8	336.2	62,294.3				2,180.5

Table 2: Running times in seconds for processing single-target and multi-target regression data sets.

5.5 Analysis on the Effect of Linear Terms on FIRE

In this section, we analyze the effect that linear terms have on the performance of FIRE. As noted in Sections 5.2 and 5.3, small FIRE models do not seem to benefit from the linear terms in the multi-target setting. In single-target domains and for unlimited FIRE, however, the effect is always positive. Especially for the small size limited models in single-target case, the effect is very significant as mentioned in Section 5.1. The reason for this phenomenon can be understood when we remember the definition of the linear terms in the multi-target model:

$$\boldsymbol{x}_{(t,j)} = (0,\ldots,\underset{t-1}{0},\underset{t}{x_j},\underset{t+1}{0},\ldots,0).$$

The predictive power of a linear term clearly lowers with more targets and is the highest for single-target data. This is due to the fact that a linear term gives prediction only to a single target even if its effect on the ensemble size is the same as that of a rule. A rule, on the other hand, predicts all the targets at once and, thus, has a higher effect on the loss function. That is, in case of limited FIRE, each linear term occupies a slot of a rule but only has an effect corresponding to a proportion of $1/T$. In the unlimited FIRE we do not have such a contest between linear terms and rules during the optimization and, thus, the negative effect of adding a linear term is negligible.

The numbers of rules and linear terms presented in Tables 4 and 5 (Appendix D) support this line of reasoning. For the limited FIRE, the average number of linear terms lowers from 12% for single-target to 5% for multi-target. For the unlimited version, the values are respectively 2% and 24%. For the limited FIRE, the optimization process does not consider linear terms helpful in the multi-target case. Only a small number of linear terms is kept in the resulting model and they seem to have a negligible or even a negative effect on the accuracy. Moreover, apart from the WATER QUALITY data set, the data sets with most targets give rise to models with a greater portion of linear terms. Apparently, with many targets we need more linear terms to achieve a similar effect.

The results with different size limitations in Section 5.3 give an interesting view on the effect of linear terms on the models. In the multi-target case, only larger models benefit from adding the linear terms. The effect is negative for smaller models. The threshold in our case seems to be around the size limit of 50 terms. Clearly, these results further indicate that linear terms are of benefit only if we already have enough rules to cover the predictive problem well enough. With smaller limitations, they only take place away from the more powerful rules. However, for single-target rule ensembles the linear terms seem to help all the time as seen in Figure 3. A similar effect can be assumed for data sets with a small number of targets.

The issues presented above raise the question whether there is a better, more compressed way to include linear terms in the ensemble. We could, for instance, have a random combination of target dimensions covered for each linear term. It is an interesting question whether this would benefit the multi-target rule ensembles in the same way as the (single dimension) linear terms benefit the single-target rule ensembles.

We could also consider altering the normalization process for the linear terms described in Section 3.2. To add more effect to the linear terms, we could simply multiply the single nonzero prediction with the number of targets T. Intuitively this could bring the loss effect of a linear term to a level approximately equal to that of a rule. Unfortunately, in our experiments (not presented here) this seemed to stress linear terms too much and resulted in less accurate models.

Finally, it is not clear whether it is a good idea to count a linear term with the same weight as a rule when the model size is considered. After all, multiple linear terms can be united as shown

in Example 1 and are easier to interpret than a set of rules. This issue needs to be investigated in further work.

5.6 Summary of the Experimental Results

In our experiments, we first evaluated FIRE on single-target domains in order to show that our implementation of rule ensembles also works on standard regression problems. The results show that all versions of FIRE have accuracy that is comparable to the accuracy of the reference algorithms. The benefits of FIRE are even more evident when we take the model size into account: the most accurate models, random forests, are much larger.

Second and more important, we evaluated FIRE on multi-target domains. The results are somewhat similar to the ones obtained on single-target domains: random forests and the unlimited FIRE versions are more accurate than the limited FIRE, regression trees, and MTSMOTI. Nevertheless, the multi-task algorithm DIRTY performed much better now, being ranked right after the unlimited FIRE. As in the single-target case, the model size of MTSMOTI, regression trees, and the limited FIRE versions are statistically significantly smaller than the model size of the most accurate algorithms. Even though the difference in size between random forests and the unlimited FIRE is not statistically significant, the average size of a random forest is more than 300 times larger than the average size of FIRE models (with and without linear terms). In addition, although DIRTY performed well on average, its results were very unstable. Both the accuracy and computational resource usage varied highly from one data set to another.

Overall, the unlimited FIRE achieves good balance between accuracy and simplicity. Although it tends to generate somewhat less accurate models than the most accurate random forests, the size difference is very large. Therefore, we believe that the unlimited FIRE with linear terms is a very good choice for modeling multi-target regression problems.

Finally, the investigation of the influence of the maximum model size on the accuracy confirms that this parameter can be successfully used to control the accuracy-for-simplicity trade-off. The results show the general trend of larger models having better accuracy. The fact that the trend is more evident in the multi-target domains can be attributed to multi-target tasks being more complicated and demanding more complex models for maximal accuracy. In the case of linear terms, we can conclude that for larger (50 terms or more) multi-target rule ensembles adding linear terms should in general improve the accuracy. However, more importantly, we should rethink the way linear terms are created and handled in the multi-target environment, trying to achieve as great a benefit in a multi-target as in single-target setting.

6. Conclusions and Further Work

In many application areas there is a need for methods that can learn interpretable multi-target models, that is, models that predict several target attributes simultaneously. Rules are undoubtably one of the most interpretable model types. We have adopted the rule ensemble approach and generalized it to multi-target regression domains.

The initial implementation of our algorithm FIRE (Aho et al., 2009) was already able to learn multi-target regression rule ensembles. In this work, we have extended it so that in addition to rules, linear terms can also be added to the ensemble. The performance of the algorithm has also been significantly improved by modifying the normalization and optimization steps. We also include a much more in-depth experimental evaluation, including more data sets and more reference algorithms.

Our implementation has a simple parameter for limiting the size (the number of rules and linear terms) of the learned model. This enables us to trade accuracy for size (and interpretability) of learned models. We evaluated our algorithm with two parameter values: One that limits the number of rules to a maximum of 30, and one that has no size restrictions. We compared it with two other existing rule ensembles approaches RULEFIT and REGENDER, and to other similar multi-target prediction algorithms, namely regression trees, random forests, and model trees (MTSMOTI). In addition, we compared our multi-target algorithm with a recent multi-task algorithm DIRTY. We also investigated how the size limit and adding linear terms to the ensemble affect the accuracy.

In both, the single-target and multi-target settings, the unlimited FIRE with linear terms performed well, especially when the model size is taken into account. The model size limitation parameter can be used to tune the accuracy-for-simplicity trade-off. In sum, FIRE achieves a good balance of accuracy and simplicity in the context of multi-target regression.

Let us conclude with some ideas for further work. There are still some features of RULEFIT that have not yet been added to FIRE. Thus, a natural direction for further work is to add these features and study if they improve the performance of FIRE. This might include, for example, the use of tree ensembles based on importance-sampling. FIRE uses the gradient descent method for weight optimization and an ad-hoc approach to selecting the optimal τ value. We could combine both optimization problems into a single one. However, such a combined problem might no longer be convex and gradient descent optimization could get trapped in local optima. A possible solution would be to use metaheuristic optimization methods, such as differential evolution or ant-colony optimization instead.

On the other hand, it would be interesting to explore whether and how the recent multi-task optimization methods can be adapted to the multi-target environment. Moreover, our experiments suggest that we should further study the efficient application of linear terms to multi-target prediction. Finally, the automated exploration of the accuracy-for-simplicity trade-off by selecting an appropriate model size (the number of rules) deserves further attention.

Acknowledgments

We would like to thank the anonymous reviewers for insightful and helpful comments. We also want to thank Ali Jalali, Jerome H. Friedman and Annalisa Appice for the generous help with the implementations of their algorithms.

The work of T. Aho and T. Elomaa was supported by the Academy of Finland (through the research project REALM *Reactive learning and mining*). Moreover, the work of T. Aho was financially supported by Tampere Doctoral Programme in Information Science and Engineering.

The work of B. Ženko and S. Džeroski was supported by the Slovenian Research Agency (Grants P2-0103 and J2-2285), the European Commission (Grants ICT-2010-266722 and ICT-2011-287713), and Operation no. OP13.1.1.2.02.0005 financed by the European Regional Development Fund (85%) and the Ministry of Education, Science, Culture and Sport (15%).

Appendix A. Gradient Step Size

In this section, we present the reasoning behind the step size selection used in the FIRE gradient descent optimization procedure (Algorithm 2, line 17).

First of all, let us change the predictive function in Equation 2 to the following form:

$$\boldsymbol{f}(\boldsymbol{x}) = w_0\,\boldsymbol{avg} + \sum_{i=1}^{M} w_i \boldsymbol{r}_i(\boldsymbol{x}) + \sum_{j=1}^{J}\sum_{k=1}^{K} w_{(j,k)}\boldsymbol{x}_{(j,k)} = \boldsymbol{w}\left(\boldsymbol{P}_1^{\mathsf{T}}, \ldots, \boldsymbol{P}_J^{\mathsf{T}}\right).$$

Here J is the number of targets and each $\boldsymbol{P}_j^{\mathsf{T}}$ represents the predictions for the target dimension j:

$$\boldsymbol{P}_j = \left(avg_j, r_1(\boldsymbol{x})_j, r_2(\boldsymbol{x})_j, \ldots \left[\boldsymbol{x}_{(1,1)}\right]_j, \left[\boldsymbol{x}_{(1,2)}\right]_j, \ldots\right).$$

In our case, the loss function is presented by Equation 5 as

$$L(\boldsymbol{x};\boldsymbol{w}) = \frac{1}{2J}\sum_{j}^{J}(f_j(\boldsymbol{x}) - y_j)^2 = \frac{1}{2J}\sum_{j}^{J}(\boldsymbol{w}\boldsymbol{P}_j^{\mathsf{T}} - y_j)^2.$$

Let us denote the weights by $\boldsymbol{w}_k$ and the gradients by $\boldsymbol{g}_k$ at the end of the iteration k (Algorithm 2, at the beginning of line 17). The step we are taking to change the weight is defined by $\boldsymbol{w}_{k+1} = \boldsymbol{w}_k - \beta\boldsymbol{g}_k$, where β is the step size. Now, in the iteration k we want to minimize the one-dimensional loss function over step size

$$\Phi(\beta) = L(\boldsymbol{x}; \boldsymbol{w}_k - \beta\boldsymbol{g}_k),$$

where

$$\boldsymbol{g}_k^{\mathsf{T}} = \partial L/\partial \boldsymbol{w}_k = \frac{1}{J}\sum_{j}^{J}\left(\boldsymbol{w}_k\boldsymbol{P}_j^{\mathsf{T}} - y_j\right)\boldsymbol{P}_j^{\mathsf{T}}.$$

By finding the zero point of the derivative of the expected value

$$\mathbf{E}\,\Phi'(\beta) = \mathbf{E}\,\frac{1}{J}\sum_{j}^{J}\beta(\boldsymbol{g}_k\boldsymbol{P}_j^{\mathsf{T}})^2 - (\boldsymbol{w}_k\boldsymbol{P}_j^{\mathsf{T}} - y_j)\boldsymbol{g}_k\boldsymbol{P}_j^{\mathsf{T}}$$

we get the optimal step size β^* for minimizing the loss function Φ during each iteration k:

$$\beta^* = \frac{\mathbf{E}\,J\|\boldsymbol{g}_k\|^2}{\mathbf{E}\sum_j^J\left(\boldsymbol{g}_k\boldsymbol{P}_j^{\mathsf{T}}\right)^2}. \tag{6}$$

Unfortunately, computing the precise value for each iteration is very costly. Thus, let us try to find some approximation. By using the Cauchy-Schwarz inequality we get the following upper bound for the numerator:

$$(\boldsymbol{g}\boldsymbol{P}_j^{\mathsf{T}})^2 \leq \boldsymbol{g}\boldsymbol{g}^{\mathsf{T}}\boldsymbol{P}_j\boldsymbol{P}_j^{\mathsf{T}}.$$

Now, noting that $\|\boldsymbol{g}\|^2 = 0$ means a zero size step, we can write the following bound for the optimal step size β^*:

$$\beta^* \geq \frac{J}{\mathbf{E}\sum_j^J\|\boldsymbol{P}_j(\boldsymbol{x})\|^2} = \beta_{\text{low}}.$$

Because of the normalization explained in Section 3.1 and Appendix B, we know that during the optimization $\boldsymbol{avg} = \mathbf{1}$. Thus, we note that $\beta_{\text{low}} \leq 1$ always holds for the bound.

We could start with step size 1 and reduce the step size in a logarithmic fashion so that after a while we reach the lower bound. The optimal step size β^* depends on the gradient as seen in Equation 6. Reducing the step size like this would cause β to be near the optimal value most of time and end below it. However, in practice it seems that using β_{low} as a step size gives good results. Thus, we simply use this lower bound as a constant step size.

Appendix B. Justification for the Normalization Process

In this section, we give reasons for the decisions of the normalization process described in Section 3.1.

As we recall from Equation 2, our model is of the form

$$\boldsymbol{f}(\boldsymbol{x}) = w_0\,\boldsymbol{avg} + \sum_{i=1}^{M} w_i \boldsymbol{r}_i(\boldsymbol{x}) + \sum_{j=1}^{J}\sum_{k=1}^{K} w_{(j,k)} \boldsymbol{x}_{(j,k)}.$$

The linear terms in the last sum are optional and we concentrate on the rules part first.

First we rationalize why the first step, centering, in normalization presented in Section 3.1 is needed. We remember that our rules are transformed from a tree ensemble of form:

$$\boldsymbol{f}(\boldsymbol{x}) = \frac{1}{|D|}\sum_{i=1}^{|D|} \boldsymbol{d}_i(\boldsymbol{x}),$$

where $|D|$ is the number of trees in the ensemble. Here each tree prediction $\boldsymbol{d}_i$ in the ensemble is *global* in contrast to rule predictions $\boldsymbol{r}_i$ being *local*. That is, the tree prediction functions $\boldsymbol{d}_i$ give a prediction to all possible instances $\boldsymbol{x}$, while rules predict only the subset they cover. Otherwise the rule function $\boldsymbol{r}_i$ equals zero. In other words, the tree prediction functions cover all the instances.

Initially, right after converting the trees to rules, we know that out of the M rules, exactly $|D|$ cover an arbitrary instance $\boldsymbol{x}$. This consists of a rule from each of the trees in the tree ensemble. In this case, we can simply take an average of the predictive functions to get the overall prediction:

$$\boldsymbol{f}(\boldsymbol{x}) = \frac{1}{|D|}\sum_{i=1}^{M} \boldsymbol{r}_i(\boldsymbol{x}).$$

Now the rule predictions are used in the same scale in which they were created during the tree ensemble training.

Nevertheless, this is problematic if we omit a part of the initial rule set as is done in normalization. In this case, we do not know how many rules cover the given instance and, thus, can not simply take average of the covering instances. A simple solution to this problem is to remove the average offset that is included in all the predictions. That is, we replace each of the initial rules $\boldsymbol{r}''$ with a zero-centered rule $\boldsymbol{r}'$, which is defined as

$$\boldsymbol{r}'(\boldsymbol{x}) = \begin{cases} \boldsymbol{r}''(\boldsymbol{x}) - \boldsymbol{avg} & \text{if } \boldsymbol{x} \text{ is covered and} \\ 0 & \text{otherwise.} \end{cases}$$

Now we can define a rule set equivalent to the initial one with

$$\begin{aligned} &\frac{1}{\text{nb. of covering rules}}\sum_{i=1}^{M} \boldsymbol{r}_i''(\boldsymbol{x}) \\ &= \frac{\text{nb. of covering rules}}{\text{nb. of covering rules}}\,\boldsymbol{avg} + \sum_{i=1}^{M} \boldsymbol{r}_i'(\boldsymbol{x}) = \boldsymbol{avg} + \sum_{i=1}^{M} \boldsymbol{r}_i'(\boldsymbol{x}). \end{aligned}$$

This new form allows us to do the weight optimization freely without caring about the number of covering rules:

$$\boldsymbol{f}(\boldsymbol{x}) = w_0\,\boldsymbol{avg} + \sum_{i=1}^{M} w_i \boldsymbol{r}_i'(\boldsymbol{x}).$$

At the second stage of normalization, our aim is to equalize the rules with respect to the optimization problem presented in Equation 4:

$$\arg\min_{\boldsymbol{w}} \sum_{(\boldsymbol{x},\boldsymbol{y})\in E} L(\boldsymbol{f}(\boldsymbol{x}),\boldsymbol{y}) + \lambda \sum_{i=1}^{M} |w_i|^{\alpha}.$$

The optimization problem is not invariant to the scaling of rule predictions: If we scale the rule predictions as $\boldsymbol{r} = b\boldsymbol{r}'; b > 1$, the corresponding weight will not be simply decreased as $w = w'/b$, because the regularization part on the right only includes weights and not rule predictions. As a result, the rules with smaller (absolute) predicted values are penalized more during optimization than the ones with larger predicted values. We would like to have all the rules and targets to have equal initial importance.

The obvious approach of setting a constant value to the base model target predictions is suboptimal for multi-target problems. Let us illustrate this with an example. We are given two target features y_1, y_2 that are highly inversely linearly dependent so that linear base model of type $(1, -5)$ would give high predictive accuracy. It is clear that setting 1 to the rule predictions for all targets would discard all the discovered information on relations between the targets stored in the rules.

Thus, we choose to scale each rule $\boldsymbol{r}'(\boldsymbol{x})$ with a value that corresponds to its initial predictive size $\chi(\boldsymbol{r}')$:

$$\boldsymbol{r} = \frac{\boldsymbol{r'}}{\chi}. \tag{7}$$

But how should we choose the exact value of χ?

To simplify the relations of target prediction ranges it is useful to bound the scaled predictions to some closed interval, for example, $[-1, 1]$. In this case χ should be related to the largest predicted value of the rule $\boldsymbol{r'}$. However, we also have to note the differing scales of the target attributes r'_t. If σ_t were the standard deviation of a normally distributed target attribute r'_t, dividing a zero-centered attribute by $2\sigma_t$ should put 95% of all values within the $[-1, 1]$ interval.

Thus, when target prediction r'_m is the largest, we would use as the normalization factor χ in Equation 7 the value

$$\chi = \frac{r'_m}{2\sigma_m}.$$

More in detail, the index $m \in \{1, \ldots, T\}$ of the maximum target value r'_m of the rule $\boldsymbol{r'}$ is defined by:

$$m = \arg\max_t \left| \frac{r'_t}{2\sigma_t} \right|.$$

This way we make all the rules have equal maximal target prediction. We can now also give a strict bound to the predictions r_t: By the definition of m, after this second stage of normalization it holds that $r_m = 1$ and $|r_t| = |\sigma_m / r'_m r'_t / \sigma_t| \leq 1$ for all other targets t. Alternative choices for the normalization factor χ exist, but they may not behave as well in practice (Aho, 2012).

To sum up, at the second stage of normalization the target predictions r_t in a certain rule $\boldsymbol{r}$ are scaled by a factor χ that represents the the initial size of the predictive values of the particular rule. Thus, this stage roughly equalizes the rules before the optimization phase and affects the rules of the final model. After the two first normalization steps our rule predictions are of form:

$$\boldsymbol{r} = \frac{\boldsymbol{r'}}{\chi} = \frac{\boldsymbol{r''} - \boldsymbol{avg}}{\chi}.$$

In addition to equalizing the initial importance of rules, we also have to equalize the effect of differing target scales during the optimization. Otherwise, targets with large scales would dominate the rule selection. This is done in the third stage of the normalization. However, clearly our intention is to use this normalization only temporarily during optimization. Otherwise the resulting model would not anymore be applicable to real world data. As usual, we do this simply by dividing the target attribute prediction values r_t by twice their standard deviations $2\sigma_t$:

$$r_t^* = \frac{r_t}{2\sigma_t} = \frac{r_t'}{2\sigma_t\chi} = \frac{r_t'' - avg_t}{2\sigma_t\chi}.$$

The rationalization on the normalization of the linear terms is similar to what was presented here for the rules. Nevertheless, linear terms are global in nature.

We recall that single linear term is defined as

$$\boldsymbol{x}_{(t,j)} = (0, \ldots, \underset{t-1}{0}, \underset{t}{x_j}, \underset{t+1}{0}, \ldots, 0),$$

which depicts the influence of the descriptive attribute x_j on the target attribute x_t. We add linear terms for all possible combinations of numeric descriptive attributes and target attributes.

Unlike rules, linear terms are affected by *two* attributes and, thus, *two* attribute space scales: that of the j-th descriptive attribute space and that of t-th target space. In addition, linear terms with their single nonzero coordinate are multi-target only in principle. There is no sense in scaling with the maximum coordinate of linear terms as was done in the second normalization stage. These differences have to be taken into account.

We again shift the linear terms by the average $\overline{x_j''}$ of the j-th descriptive attribute:

$$\boldsymbol{x}'_{(t,j)} = (0, \ldots, x_j'' - \overline{x_j''}, \ldots, 0).$$

Here x_j'' is the original attribute x_j value. However, we continue by normalizing the terms to the target attribute scale

$$\boldsymbol{x}_{(t,j)} = \boldsymbol{x}'_{(t,j)} \frac{\sigma_t}{\sigma_j}.$$

Here we note the effect of two separate attribute spaces. Linear terms normalized like this appear in the final rule ensemble model.

However, analogously to the third stage of rule normalization we also scale the target dimension space out temporarily:

$$\boldsymbol{x}^*_{(t,j)} = \frac{\boldsymbol{x}_{(t,j)}}{2\sigma_t} = \frac{\boldsymbol{x}'_{(t,j)}}{2\sigma_j}.$$

This is, again, only intended to equalize the terms of different target attributes during the optimization procedure.

In Aho et al. (2009) we externally normalized the data. It is worth noting that the normalization process illustrated in Section 3.1 can not be trivially reduced to the external normalization. Let us concentrate on a single-target model and denote with † the functions and weights that result from the optimization process after external normalization. We now have the following form for the external

normalization results:

$$f^{\dagger}(\boldsymbol{x}^{\dagger}) = w_0^{\dagger} avg^{\dagger} + \sum_{i=1}^{M} w_i^{\dagger} r_i^{\dagger}(\boldsymbol{x}^{\dagger}) \overset{avg^{\dagger}=0}{=} \sum_{i=1}^{M} w_i^{\dagger} r_i^{\dagger}\left(\frac{1}{2}(\boldsymbol{x}-\overline{\boldsymbol{x}}) \odot \boldsymbol{\sigma}^{-1}\right)$$
$$= \sum_{i=1}^{M} w_i^{\dagger} I_i(\boldsymbol{x}) \frac{r_i''(\boldsymbol{x}) - avg}{2\sigma_i} = \sum_{i=1}^{M} \frac{w_i^{\dagger}}{2\sigma_i} r_i'(\boldsymbol{x}).$$

Here $\odot$ is the coordinate-wise Hadamard product and $\boldsymbol{\sigma}^{-1}$ is a vector which consists of the inverses of standard deviations: $1/\sigma$. Moreover, the indicator function I_i equals 1 if r_i' covers the instance and zero otherwise. The third equality follows from the fact that our tree ensemble training algorithm is invariant to scaling and behaves similarly with normalized and unprocessed data.

We notice that the form on last row can be expressed in the form of internal normalization

$$f(\boldsymbol{x}) = w_0\, avg + \sum_{i=1}^{M} w_i r_i'(\boldsymbol{x})$$

only if $avg = 0$, that is, the data is originally zero-centered. We realize that a lacking term affects the optimization of the remaining weights. Thus, in the general case transforming an externally normalized model to an equivalent internally normalized one is not trivially possible.

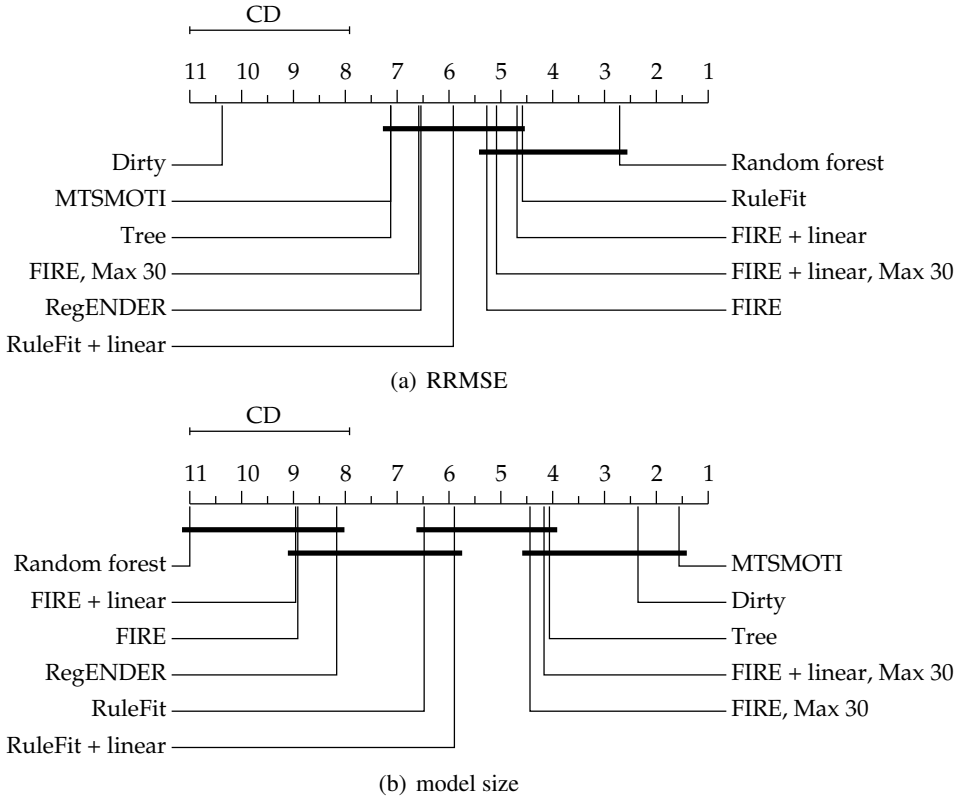
Appendix C. Comparison to RULEFIT

In this appendix, we study the differences between the FIRE and RULEFIT methods used in the experiments shown in Section 5.1. We used the RULEFIT settings that give the most accurate models as recommended by Friedman and Popescu (2008, 2004): these settings are presented in Section 4. However, an interesting question is prompted by the results in Section 5.1 and especially by Figure 1(a): Why does FIRE behave so much worse than RULEFIT on the single-target data? To what extent is this difference caused by the implementation and to what extent is it caused only by the different parameter settings implemented? In this section, we try to find an answer to these questions.

Figure 4 is analogous to Figure 1, the only difference is that now RULEFIT is using the parameter settings very similar to FIRE. As assumed, the behavior of the two algorithms is now much more alike. As seen in Figure 4(a), the accuracy of the unlimited FIRE versions is between those of the RULEFIT models. Surprisingly, however, FIRE now seems to gain more from linear terms while RULEFIT is more accurate without them.

There are several important RULEFIT features that could still be included in FIRE. The detailed list of differences is presented in Table 3. The most important difference is in the way the initial rule set is created. While FIRE (Algorithm 1, line 1) generates random forests, the best version of RULEFIT uses the best performing ISLE (Friedman and Popescu, 2003) tree ensembles. The major distinction here is that ISLE on each tree generation iteration takes into account the previously generated ensemble members when inducing the new tree. In this sense random forests are memoryless when compared with ISLE. As shown by Friedman and Popescu (2003), the difference in performance between ISLE and random forests may be very significant.

Another major difference is the robustness of RULEFIT. The used loss function and the linear terms are made robust against outliers. In addition, there are some minor differences. For example, for the best version of RULEFIT the size of each tree ensemble is larger and, instead of only leaves,



(a) RRMSE

(b) model size

Figure 4: Average ranks diagrams on *single-target* data with RULEFIT parameters similar to FIRE for *RRMSE* (a) and *model size* (b). Better algorithms are on the right-hand side, the ones that do not differ significantly by the Nemenyi test (p-value $= 0.05$) are connected with a horizontal bar. CD is the critical distance.

all tree nodes are turned into rules. This is possible because rule predictions are always set to 1 in the initial set of rules.

See Friedman and Popescu (2008, 2004) for more details about the parameter settings. We can conclude that we should be able to still improve the accuracy of FIRE by implementing some additional features present in RULEFIT.

Appendix D. Detailed Experimental Results

The detailed results discussed in Sections 5.1 and 5.2 are presented in Table 4 for single-target problems and in Table 5 for multi-target problems. In the table, DIRTY size is presented in number of nonzero weights as was done by Jalali et al. (2011). However, since small magnitude weights may disturb the view, we here also report average sizes where weights having absolute value under a threshold are removed. With thresholds $c \cdot \text{median}(B+S)$ where $c \in \{0, 0.01, 0.1, 0.3, 0.5\}$, we have the sizes 10.4, 10.2, 9.5, 8.5, 7.6 in single-target and 293, 264, 228, 198, 178 in multi-target cases. Thus, the amount of very small weights seems not to be very relevant. The amount of optimized weights altogether was on average 12.6 in single-target and 336.7 in multi-target.

SETTING	BEST RULEFIT	RULEFIT MOST SIMILAR TO FIRE	FIRE
LINEAR FUNC. TRIMMING	0.025	0.0	0.0
HUBER LOSS TRIMMING	0.9	(SQR LOSS) 1.0	(SQR LOSS) 1.0
MAX. NB OF RULES	5000	1000	ON AVG $\approx$700
MEAN MAX. NB. OF LEAVES	6	7	$\approx$7
INCENTIVE FACTOR	3*	(NOT USED) 1.0	(NOT USED) 1.0
TREE MEMORY FACTOR	0.01	(NOT USED) 0.0	(NOT USED) 0.0
TREE SAMPLING FRACTION	$1/5$	$1-1/e$	$1-1/e$
GD VALID. SET SIZE	3-FOLD XVAL	1/3	1/3
GD STEP SIZE	0.01*	0.01*	SEE APPENDIX A

Table 3: Setting differences for FIRE and RULEFIT in single-target experiments. The value is marked with an asterisk if the best value of RULEFIT was not mentioned in the original papers or if value similar to FIRE was not available. Default values were used in these cases.

	TREE		RANDOM FOREST		FIRE		FIRE + LINEAR		FIRE, MAX 30		FIRE + LINEAR, MAX 30		MTSMOTI		DIRTY		RULEFIT		RULEFIT + LINEAR		REGENDER	
DATA SET	RRMSE	#	RRMSE	#	RRMSE	#	RRMSE	# [%]	RRMSE	#	RRMSE	# [%]	RRMSE	#	RRMSE	#	RRMSE	#	RRMSE	# [%]	RRMSE	#
ABALONE	0.72	68	**0.67**	136,291	0.68	810	0.68	817 [1]	0.73	30	0.68	30 [23]	0.69	14	1.27	8	**0.67**	214	**0.67**	162 [2]	**0.67**	200
AILERONS	0.67	54	0.58	49,442	0.58	300	0.53	361 [9]	0.62	30	0.51	30 [77]	0.54	2	2.65	6	0.51	189	**0.50**	187 [3]	1.02	200
AUTO-MPG	0.45	16	**0.36**	14,310	0.37	780	0.37	768 [1]	0.39	30	0.37	30 [7]	0.46	11	1.73	7	**0.36**	164	**0.36**	116 [2]	0.39	200
AUTO-PRICE	0.49	9	0.35	5,554	**0.34**	629	0.36	120 [8]	0.43	30	**0.34**	30 [17]	0.39	6	1.02	15	0.38	125	0.38	125 [2]	0.37	200
BREAST CANCER	**0.96**	3	0.99	8,727	0.99	724	1.00	725 [0]	0.97	30	0.97	30 [0]	1.01	7	1.10	9	0.97	13	0.97	13 [0]	1.04	200
CENSUS	0.63	504	**0.55**	827,401	0.59	827	0.59	835 [1]	0.69	30	0.68	30 [10]	0.66	12	1.37	8	0.63	195	0.63	181 [2]	0.58	200
CLOUD	0.58	9	0.48	4,185	0.48	34	**0.42**	626 [1]	0.52	30	**0.42**	30 [10]	0.58	4	1.09	6	0.54	108	0.50	25 [8]	0.54	200
CPU ACTIVITY	0.19	317	**0.15**	270,473	0.18	718	0.17	730 [2]	0.24	30	0.22	30 [10]	0.19	6	1.06	12	0.16	257	0.20	163 [6]	0.18	200
COMPUTER HW	0.96	3	0.39	6,996	0.39	538	0.32	105 [6]	0.57	30	0.33	30 [13]	**0.22**	2	0.52	7	0.59	122	0.57	90 [6]	0.45	200
DELTA-AILERONS	0.60	106	**0.53**	210,808	0.54	600	0.54	549 [1]	0.59	30	0.55	30 [10]	0.57	16	0.87	5	0.55	217	0.55	148 [1]	0.95	200
DIABETES	1.02	3	0.89	1,610	0.89	385	0.89	387 [1]	**0.81**	30	0.84	30 [7]	0.99	1	2.47	2	0.88	22	0.90	22 [0]	1.05	200
ECHOCARDIOGRAM	**0.68**	3	0.76	4,472	0.85	579	0.85	537 [1]	0.86	30	0.87	30 [3]	0.76	3	1.08	9	0.78	87	0.78	86 [2]	0.92	200
HOUSING	0.43	32	**0.36**	18,864	0.38	379	0.38	777 [2]	0.45	30	0.44	30 [10]	0.47	4	1.18	13	0.37	190	0.37	175 [2]	0.39	200
HOUSING CA	0.51	745	**0.42**	782,382	0.54	745	0.52	827 [1]	0.63	30	0.58	30 [10]	0.60	13	1.44	8	0.48	257	0.47	202 [2]	0.47	200
KINEMATICS	0.72	290	**0.53**	318,707	0.63	722	0.63	619 [1]	0.74	30	0.72	30 [13]	0.77	4	2.83	8	0.58	220	0.58	205 [0]	0.54	200
META-DATA	1.03	1	**0.96**	5,613	0.99	54	1.00	0 [0]	0.99	30	1.00	0 [0]	1.00	15	0.98	21	0.99	66	0.99	66 [0]	1.59	200
PBC	0.94	6	**0.82**	13,687	0.83	748	0.83	758 [1]	**0.82**	30	**0.82**	30 [10]	0.84	2	1.62	18	**0.82**	130	**0.82**	133 [2]	0.84	200
POLE TELECOMM	0.17	377	0.20	96,380	0.30	757	0.30	783 [3]	0.38	30	0.38	30 [23]	0.26	21	0.97	30	0.19	123	0.19	125 [3]	**0.16**	200
PYRIMIDINES	0.84	6	0.77	1,971	0.81	26	0.82	199 [8]	0.81	26	0.89	8 [25]	1.77	2	1.15	26	0.66	45	**0.65**	26 [15]	0.71	200
QUAKE	**0.99**	3	1.03	71,100	1.03	91	1.03	91 [2]	1.01	30	1.01	30 [3]	1.00	2	24.7	3	1.08	0	1.08	0 [0]	1.01	200
SENSORY	0.94	6	**0.85**	17,972	**0.85**	128	**0.85**	128 [0]	**0.85**	30	**0.85**	30 [0]	0.91	16	6.29	11	**0.85**	167	**0.85**	167 [0]	0.89	200
SERVO	0.42	11	**0.31**	5,555	0.38	436	0.38	436 [0]	0.42	30	0.42	30 [0]	0.55	7	0.89	4	0.38	188	0.38	188 [0]	**0.31**	200
STRIKE	0.98	12	0.93	23,379	0.95	404	0.96	410 [1]	0.93	30	0.93	30 [0]	0.95	2	1.00	6	**0.88**	135	**0.88**	140 [0]	0.98	200
VETERAN	0.99	2	**0.92**	4,872	0.93	418	0.93	417 [0]	0.94	30	0.93	30 [3]	0.93	4	0.94	7	0.96	54	0.95	51 [2]	1.16	200
AVERAGE	0.68	108	**0.62**	120,865	0.65	493	0.64	500 [2.1]	0.68	30	0.66	28 [12]	0.71	7.3	2.51	10	0.64	137	0.63	116 [2.5]	0.72	200

Table 4: Comparison of *RRMSE* for *single-target* regression. In each row, the smallest error is typeset in bold. Size is given either as the number of rules and linear terms or as the number of terminal nodes in trees. Besides the model sizes (#), we also give in brackets the percentage of linear terms within the total model size ([%] if present). The final row gives the average errors and rule set sizes over all data sets.

DATA SET	TREE		RANDOM FOREST		FIRE		FIRE + LINEAR		FIRE, MAX 30		FIRE + LINEAR, MAX 30		MTSMOTI		DIRTY	
TARGET ATTRIBUTES	RRMSE	#	RRMSE	#	RRMSE	#	RRMSE	# [%]	RRMSE	#	RRMSE	# [%]	RRMSE	#	RRMSE	#
COLLEMBOLAN	0.97	3	0.92	13,145	0.93	438	0.93	397 [15]	0.96	30	0.95	30 [3]	1.04	16	**0.62**	141
SPECIES-NB	0.98		0.94		0.95		0.95		0.98		0.97		1.01		**0.37**	
ABUD-TOTAL	0.96		**0.93**		0.94		0.94		0.97		0.95		1.10		1.04	
ABUD-F.QUAD	0.96		0.89		0.90		0.90		0.94		0.93		1.01		**0.44**	
EDM	0.72	11	0.69	2,923	**0.68**	676	**0.68**	708 [5]	0.70	30	0.70	30 [3]	0.85	2	0.99	32
D-FLOW	0.68		0.66		0.66		**0.65**		0.71		0.71		0.92		0.99	
D-GAP	0.75		0.71		0.71		0.71		**0.69**		**0.69**		0.77		0.99	
FOREST KRAS	0.62	1,142	0.55	1,530,421	0.65	830	0.64	2,590 [68]	0.71	30	0.71	30 [0]	0.69	10	**0.21**	1,760
CC	0.53		**0.47**		0.56		0.54		0.61		0.61		0.57		0.48	
FSH	0.49		0.42		0.53		0.52		0.59		0.60		0.59		**0.18**	
DELVEG	0.53		**0.47**		0.55		0.54		0.61		0.61		0.56		**0.47**	
VPV1-HMX	0.60		0.52		0.63		0.62		0.70		0.71		0.68		**0.28**	
VPV1-H99	0.58		0.51		0.62		0.61		0.70		0.71		0.68		**0.22**	
VPV1-H95	0.57		0.50		0.61		0.60		0.69		0.70		0.67		**0.20**	
VPV1-H75	0.57		0.51		0.60		0.60		0.69		0.69		0.67		**0.16**	
VPV1-H50	0.60		0.54		0.63		0.63		0.71		0.71		0.70		**0.13**	
VPV1-H25	0.66		0.60		0.70		0.69		0.76		0.76		0.75		**0.09**	
VPV1-H10	0.76		0.69		0.78		0.78		0.83		0.83		0.82		**0.06**	
VPV1-H05	0.83		0.76		0.83		0.83		0.87		0.87		0.86		**0.04**	
FOREST SLIVNICA	0.54	321	0.49	227,198	0.50	618	**0.47**	872 [25]	0.58	30	0.58	30 [10]	0.51	4	0.49	298
HEIGHT	0.56		0.51		0.53		**0.49**		0.61		0.61		0.52		0.90	
COVER	0.52		0.48		0.47		0.45		0.54		0.55		0.50		**0.08**	

Continued on the next two pages.

Table 5: Comparison of *RRMSE* for *multi-target* regression. For each data set, we first give the *average RRMSE over all targets*, and then the *RRMSE for each target separately*. In each row, the smallest error is typeset in bold. Size is given either as the number of rules and linear terms or as the number of terminal nodes in trees. Besides the model sizes (#), we also give the percentage of linear terms within the total model size in brackets ([%] if present). The two final rows give the average RRMSEs over all targets, over data set target averages and average model size over all data sets.

DATA SET	TREE		RANDOM FOREST		FIRE		FIRE + LINEAR		FIRE, MAX 30		FIRE + LINEAR, MAX 30		MTSMOTI		DIRTY	
TARGET ATTRIBUTES	RRMSE	#	RRMSE	#	RRMSE	#	RRMSE	# [%]	RRMSE	#	RRMSE	# [%]	RRMSE	#	RRMSE	#
META LEARNING	1.36	2	0.92	1,415	**0.86**	464	0.88	472 [38]	0.99	30	0.96	30 [0]	1.20	2	1.23	10
LTREE	1.47		0.93		**0.85**		0.87		0.97		0.93		1.20		1.09	
C50-RULES	1.46		0.92		**0.82**		0.84		0.97		0.93		1.17		1.10	
LINDISCR	1.12		0.87		**0.79**		**0.79**		0.94		0.91		0.96		1.35	
MLCIB1	1.49		0.95		**0.88**		0.89		1.02		0.99		1.28		1.25	
MLCNB	1.29		**0.92**		**0.92**		0.96		0.99		0.97		1.08		1.41	
RIPPER	1.33		0.93		**0.81**		0.82		0.85		0.83		1.13		1.19	
CLEM-RBFN	1.11		0.89		**0.88**		0.89		1.06		1.05		1.20		1.30	
C50-TREE	1.47		0.93		**0.85**		**0.85**		0.97		0.94		1.24		1.15	
CLEM-MLP	1.22		0.94		**0.93**		0.96		1.07		1.07		1.54		1.32	
C50-BOOST	1.51		0.95		**0.87**		**0.87**		1.00		0.96		1.20		1.13	
MICROARTHROPODS	0.77	52	0.74	12,411	0.75	642	0.75	992 [42]	0.85	30	0.85	30 [13]	0.83	18	**0.63**	417
ACARI	0.76		**0.71**		0.72		0.73		0.82		0.82		0.79		0.96	
COLLEMBOLAN	**0.74**		**0.74**		**0.74**		**0.74**		0.81		0.82		0.75		0.92	
SH-BIODIV	0.81		**0.75**		0.78		0.78		0.90		0.90		0.94		**0.00**	
SIGMEA REAL	**0.61**	12	0.65	22,506	0.69	147	0.65	209 [4]	0.74	14	0.72	30 [10]	0.62	7	1.00	8
MFO	**0.62**		0.67		0.74		0.72		0.75		0.77		0.64		0.99	
MSO	0.61		0.62		0.64		**0.59**		0.73		0.66		**0.59**		1.01	
SIGMEA SIMULATED	0.03	146	**0.02**	78,750	0.04	658	0.03	676 [3]	0.08	30	0.08	30 [17]	0.05	17	1.09	22
DISP-RATE	0.03		**0.02**		0.04		0.03		0.09		0.09		0.05		1.06	
DISP-SEEDS	0.03		**0.02**		0.03		0.03		0.08		0.07		0.04		1.12	
SOLAR FLARE	0.99	2	1.05	3,974	1.00	74	1.00	74 [0]	1.00	30	1.00	30 [0]	1.02	14	**0.93**	30
C-CLASS	0.99		1.03		1.00		1.00		1.00		1.00		1.03		**0.97**	
M-CLASS	**0.97**		1.04		0.99		0.99		0.98		0.98		1.00		**0.97**	
X-CLASS	1.01		1.07		1.01		1.01		1.03		1.03		1.04		**0.84**	

Table 5: Continued from the previous page and continued on the next page.

DATA SET	TREE		RANDOM FOREST		FIRE		FIRE + LINEAR		FIRE, MAX 30		FIRE + LINEAR, MAX 30		MTSMOTI		DIRTY	
TARGET ATTRIBUTES	RRMSE	#	RRMSE	#	RRMSE	#	RRMSE	# [%]	RRMSE	#	RRMSE	# [%]	RRMSE	#	RRMSE	#
VEGETATION	0.82	349	**0.72**	1,102,508	0.80	826	0.79	1,541 [46]	0.88	30	0.88	30 [0]	0.89	8	**0.32**	715
NUMBER-SPP	0.80		0.67		0.74		0.73		0.87		0.88		0.94		**0.02**	
DEM	0.68		0.52		0.65		0.63		0.78		0.79		0.81		**0.45**	
TWI	0.70		0.60		0.67		0.65		0.74		0.74		0.80		**0.13**	
SOLAR	0.99		0.97		0.99		0.98		0.99		0.99		0.99		**0.17**	
THINVK	0.96		**0.88**		0.94		0.94		0.98		0.98		0.96		**0.88**	
THK	0.96		**0.88**		0.94		0.94		0.98		0.98		0.96		**0.88**	
EFF-RAIN	0.68		**0.53**		0.66		0.64		0.78		0.79		0.81		0.57	
MINT-JUL	0.72		0.58		0.69		0.67		0.81		0.82		0.83		**0.29**	
MAX-FEB	0.68		0.53		0.67		0.65		0.82		0.83		0.85		**0.14**	
GRND-DPTH	0.81		0.71		0.78		0.77		0.86		0.87		0.89		**0.00**	
SALINITY	0.94		0.89		0.92		0.92		0.97		0.97		0.97		**0.00**	
WATER QUALITY	0.96	5	**0.90**	41,568	0.94	801	0.93	954 [19]	0.94	30	0.94	30 [0]	0.96	8	0.93	224
CLAD-SP	0.99		**0.94**		0.95		0.95		0.96		0.96		0.98		0.96	
GONG-INC	1.00		**0.97**		1.00		0.99		0.99		0.99		1.00		**0.97**	
OEDO-SP	1.00		0.94		0.96		0.95		0.97		0.97		0.99		**0.88**	
TIGE-TEN	0.95		**0.88**		0.93		0.91		0.93		0.93		0.95		**0.88**	
MELO-VAR	0.98		**0.90**		0.93		0.93		0.95		0.95		0.97		0.95	
NITZ-PAL	0.90		**0.83**		0.87		0.86		0.85		0.85		0.89		0.92	
AUDO-CHA	0.98		0.96		0.98		0.98		0.97		0.97		0.98		**0.82**	
ERPO-OCT	0.97		**0.91**		0.94		0.94		0.94		0.94		0.97		0.95	
GAMM-FOSS	0.93		**0.80**		0.83		0.83		0.89		0.89		0.91		0.91	
BAET-RHOD	0.97		**0.89**		0.95		0.94		0.95		0.95		0.96		0.98	
HYDRO-SP	0.98		**0.90**		0.95		0.94		0.95		0.95		0.97		0.96	
RHYA-SP	0.96		0.90		0.92		0.92		0.94		0.93		0.94		**0.85**	
SIMU-SP	0.99		**0.94**		0.98		0.97		0.99		0.99		1.00		0.97	
TUBI-SP	0.89		**0.84**		0.90		0.90		0.87		0.88		0.91		0.97	
AVERAGE – TARGETS	0.87		0.75		0.78		0.78		0.84		0.84		0.88		**0.71**	
AVERAGE – DATA SETS	0.76	185.9	**0.70**	276,075	0.71	561.3	**0.70**	862.3 [24]	0.77	28.5	0.76	30.0 [5.1]	0.79	9.6	0.77	293.3

Table 5: Continued from the previous two pages.

References

Timo Aho. *Steps on Multi-Target Prediction and Adaptability to Dynamic Input*. PhD thesis, Tampere University of Technology, Department of Software Systems, Tampere, Finland, 2012.

Timo Aho, Bernard Ženko, and Sašo Džeroski. Rule ensembles for multi-target regression. In Wei Wang, Hillol Kargupta, Sanjay Ranka, Philip S. Yu, and Xindong Wu, editors, *Proceedings of the Ninth IEEE International Conference on Data Mining (ICDM 2009)*, pages 21–30. IEEE Computer Society, 2009.

Annalisa Appice and Sašo Džeroski. Stepwise induction of multi-target model trees. In Joost N. Kok, Jacek Koronacki, Ramon Lopez de Mantaras, Stan Matwin, Dunja Mladenić, and Andrzej Skowron, editors, *Proceedings of the 18th European Conference on Machine Learning (ECML 2007)*, LNCS, pages 502–509. Springer, 2007.

Andreas Argyriou, Theodoros Evgeniou, and Massimiliano Pontil. Convex multi-task feature learning. *Machine Learning*, 73(3):243–272, 2008.

Arthur Asuncion and David J. Newman. UCI machine learning repository, 2011. URL `http://archive.ics.uci.edu/ml/`.

Steffen Bickel, Jasmina Bogojeska, Thomas Lengauer, and Tobias Scheffer. Multi-task learning for HIV therapy screening. In William W. Cohen, Andrew McCallum, and Sam T. Roweis, editors, *Proceedings of the 25th International Conference on Machine Learning (ICML 2008)*, AICPS, pages 56–63. ACM, 2008.

Hendrik Blockeel. *Top-down Induction of First Order Logical Decision Trees*. PhD thesis, Katholieke Universiteit Leuven, Department of Computer Science, Leuven, Belgium, 1998.

Hendrik Blockeel and Jan Struyf. Efficient algorithms for decision tree cross-validation. *Journal of Machine Learning Research*, 3:621–650, 2002.

Hendrik Blockeel, Luc De Raedt, and Jan Ramon. Top-down induction of clustering trees. In Jude W. Shavlik, editor, *Proceedings of the 15th International Conference on Machine Learning (ICML 1998)*, pages 55–63. Morgan Kaufmann, 1998.

Leo Breiman. Bagging predictors. *Machine Learning*, 24(2):123–140, 1996.

Leo Breiman. Random forests. *Machine Learning*, 45(1):5–32, 2001.

Leo Breiman, Jerome H. Friedman, Richard A. Olshen, and Charles J. Stone. *Classification and Regression Trees*. Wadsworth International Group, Belmont, CA, 1984.

Rich Caruana. Multitask learning. *Machine Learning*, 28(1):41–75, 1997.

Olivier Chapelle, Pannagadatta Shivaswamy, Srinivas Vadrevu, Kilian Weinberger, Ya Zhang, and Belle Tseng. Boosted multi-task learning. *Machine Learning*, 85(1):1–25, 2010.

Delve. Data for evaluating learning in valid experiments, 2011. URL `http://www.cs.toronto.edu/~delve/index.html`.

Krzysztof Dembczyński, Wojciech Kotłowski, and Roman Słowiński. Solving regression by learning an ensemble of decision rules. In Leszek Rutkowski, Ryszard Tadeusiewicz, Lotfi A. Zadeh, and Jacek M. Zurada, editors, *Proceedings of the Ninth International Conference on Artificial Intelligence and Soft Computing (ICAISC 2008)*, LNCS, pages 533–544. Springer, 2008.

Krzysztof Dembczyński, Wojciech Kotłowski, and Roman Słowiński. Maximum likelihood rule ensembles. In William W. Cohen, Andrew McCallum, and Sam T. Roweis, editors, *Proceedings of the 25th International Conference on Machine Learning (ICML 2008)*, AICPS, pages 224–231. ACM, 2008.

Damjan Demšar, Marko Debeljak, Claire Lavigne, and Sašo Džeroski. Modelling pollen dispersal of genetically modified oilseed rape within the field. In *Abstracts, the 90th ESA Annual Meeting*, page 152. The Ecological Society of America, 2005.

Damjan Demšar, Sašo Džeroski, Thomas Larsen, Jan Struyf, Jørgen Axelsen, Marianne Bruus Pedersen, and Paul Henning Krogh. Using multi-objective classification to model communities of soil microarthropods. *Ecological Modelling*, 191(1):131–143, 2006.

Janez Demšar. Statistical comparisons of classifiers over multiple data sets. *Journal of Machine Learning Research*, 7:1–30, 2006.

Thomas Dietterich. Ensemble methods in machine learning. In Josef Kittler and Fabio Roli, editors, *Proceedings of the First International Workshop on Multiple Classifier Systems (MCS 2000)*, LNCS, pages 1–15. Springer, 2000.

Sašo Džeroski, Ljupčo Todorovski, and Hendrik Blockeel. Relational ranking with predictive clustering trees. In *Proceedings of the Workshop on Active Mining (in ICDM 2002)*, pages 9–15. IEEE Computer Society, 2002.

Sašo Džeroski, Andrej Kobler, Valentin Gjorgjioski, and Panče Panov. Using decision trees to predict forest stand height and canopy cover from LANDSAT and LIDAR data. In Klaus Tochtermann and Arno Scharl, editors, *Proceedings of the 20th International Conference on Informatics for Environmental Protection (EnviroInfo 2006)*, pages 125–133. Shaker, 2006.

Sašo Džeroski, Damjan Demšar, and Jasna Grbović. Predicting chemical parameters of river water quality from bioindicator data. *Applied Intelligence*, 13(1):7–17, 2000.

Sašo Džeroski, Nathalie Colbach, and Antoine Messéan. Analysing the effect of field character on gene flow between oilseed rape varieties and volunteers with regression trees. In Antoine Messéan, editor, *Proceedings of the Second International Conference on Co-existence between GM and non-GM based Agricultural Supply Chains*, pages 207–211. Agropolis Productions, 2005.

Peter Flach and Nada Lavrač. Rule induction. In Michael R. Berthold and David J. Hand, editors, *Intelligent Data Analysis*, pages 229–267. Springer, 2003. Second edition.

Jerome H. Friedman and Bogdan E. Popescu. Importance sampled learning ensembles. Technical report, Stanford University, Department of Statistics, Stanford, CA, 2003.

Jerome H. Friedman and Bogdan E. Popescu. Gradient directed regularization for linear regression and classification. Technical report, Stanford University, Department of Statistics, Stanford, CA, 2004.

Jerome H. Friedman and Bogdan E. Popescu. Predictive learning via rule ensembles. Technical report, Stanford University, Department of Statistics, Stanford, CA, 2005.

Jerome H. Friedman and Bogdan E. Popescu. Predictive learning via rule ensembles. *The Annals of Applied Statistics*, 2(3):916–954, 2008.

Valentin Gjorgjioski, Sašo Džeroski, and Matt White. Clustering analysis of vegetation data. Technical Report 10065, Jožef Stefan Institute, Ljubljana, Slovenia, 2008.

Nitin Indurkhya and Sholom M. Weiss. Solving regression problems with rule-based ensemble classifiers. In *Proceedings of the Seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD 2001)*, pages 287–292. ACM, 2001.

Ali Jalali, Pradeep Ravikumar, Sujay Sanghavi, and Chao Ruan. A dirty model for multi-task learning. In J. Lafferty, C. K. I. Williams, J. Shawe-Taylor, R.S. Zemel, and A. Culotta, editors, *Advances in Neural Information Processing Systems (NIPS 2010)*, pages 964–972. MIT Press, 2011.

Minwoo Jeong and Gary Geunbae Lee. Multi-domain spoken language understanding with transfer learning. *Speech Communication*, 51(5):412–424, 2009.

Christian Kampichler, Sašo Džeroski, and Ralf Wieland. Application of machine learning techniques to the analysis of soil ecological data bases: relationships between habitat features and collembolan community characteristics. *Soil Biology and Biochemistry*, 32(2):197–209, 2000.

Aram Karalič. Employing linear regression in regression tree leaves. In B. Neumann, editor, *Proceedings of the Tenth European Conference on Artificial intelligence (ECAI 1992)*, pages 440–441. John Wiley & Sons, 1992.

Aram Karalič and Ivan Bratko. First order regression. *Machine Learning*, 26(2-3):147–176, 1997.

Dragi Kocev, Celine Vens, Jan Struyf, and Sašo Džeroski. Ensembles of multi-objective decision trees. In Joost N. Kok, Jacek Koronacki, Ramon Lopez de Mantaras, Stan Matwin, Dunja Mladenić, and Andrzej Skowron, editors, *Proceedings of the 18th European Conference on Machine Learning (ECML 2007)*, LNCS, pages 624–631. Springer, 2007.

Qi Liu, Qian Xu, Vincent W. Zheng, Hong Xue, Zhiwei Cao, and Qiang Yang. Multi-task learning for cross-platform siRNA efficacy prediction: an in-silico study. *BMC Bioinformatics*, 11(1): 181–196, 2010.

Karim Lounici, Massimiliano Pontil, Alexandre B. Tsybakov, and Sara A. van de Geer. Taking advantage of sparsity in multi-task learning. In *Proceedings of the 22nd Conference on Learning Theory (COLT 2009)*, 2009.

Ryszard S. Michalski. On the quasi-minimal solution of the general covering problem. In *Proceedings of the Fifth International Symposium on Information Processing (FCIP 1969)*, volume A3, Switching Circuits, pages 125–128, 1969.

Shibin Parameswaran and Kilian Weinberger. Large margin multi-task metric learning. In J. Lafferty, C. K. I. Williams, J. Shawe-Taylor, R.S. Zemel, and A. Culotta, editors, *Advances in Neural Information Processing Systems (NIPS 2010)*, pages 1867–1875. MIT Press, 2011.

Beau Piccart, Jan Struyf, and Hendrik Blockeel. Empirical asymmetric selective transfer in multi-objective decision trees. In T. Horvath, J.-F. Boulicaut, and M. Berthold, editors, *Proceedings of the Eleventh International Conference on Discovery Science (DS 2008)*, LNAI, pages 64–75. Springer, 2008.

Ross J. Quinlan. Learning with continuous classes. In A. Adams and L. Sterling, editors, *Proceedings of the Fifth Australian Joint Conference on Artificial Intelligence (AI 1992)*, pages 343–348. World Scientific, 1992.

Alain Rakotomamonjy, Rémi Flamary, Gilles Gasso, and Stéphane Canu. $\ell_p - \ell_q$ penalty for sparse linear and sparse multiple kernel multitask learning. *IEEE Transactions on Neural Networks*, 22 (8):1307–1320, 2011.

StatLib. Data sets archive, 2011. URL `http://stat.cmu.edu/datasets/`.

Daniela Stojanova. Estimating forest properties from remotely sensed data by using machine learning. Master's thesis, Jožef Stefan International Postgraduate School, Ljubljana, Slovenia, 2009.

Jan Struyf and Sašo Džeroski. Constraint based induction of multi-objective regression trees. In F. Bonchi and J. Boulicaut, editors, *Proceedings of the Fourth International Workshop on Knowledge Discovery in Inductive Databases (KDID 2005)*, LNCS, pages 222–233. Springer, 2006.

Einoshin Suzuki, Masafumi Gotoh, and Yuta Choki. Bloomy decision tree for multi-objective classification. In Luc De Raedt and Arno Siebes, editors, *Proceedings of the Fifth European Conference on Principles of Data Mining and Knowledge Discovery (PKDD 2001)*, LNCS, pages 436–447. Springer, 2001.

Robert Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society. Series B (Methodological)*, 58(1):267–288, 1996.

Luís Torgo. Regression DataSets, 2011. URL `http://www.liaad.up.pt/~ltorgo/Regression/DataSets.html`.

Vladimir N. Vapnik. *The Nature of Statistical Learning Theory*. Springer, New York, NY, 1995.

Yong Wang and Ian H. Witten. Inducing model trees for continuous classes. In *Proceedings of the Ninth European Conference on Machine Learning (ECML 1997) Poster Papers*, 1997.

Weka. Collections of datasets, 2011. URL `http://www.cs.waikato.ac.nz/~ml/weka/index_datasets.html`.

Guo-Xun Yuan, Kai-Wei Chang, Cho-Jui Hsieh, and Chih-Jen Lin. A comparison of optimization methods and software for large-scale L1-regularized linear classification. *Journal of Machine Learning Research*, 11:3183–3234, 2010.

Bernard Ženko. *Learning predictive clustering rules*. PhD thesis, University of Ljubljana, Faculty of computer and information science, Ljubljana, Slovenia, 2007.

Bernard Ženko and Sašo Džeroski. Learning classification rules for multiple target attributes. In Takashi Washio, Einoshin Suzuki, Kai Ming Ting, and Akihiro Inokuchi, editors, *Proceedings of the 12th Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD 2008)*, LNCS, pages 454–465. Springer, 2008.

Journal of Machine Learning Research 13 (2012) 2409-2464 Submitted 4/11; Revised 2/12; Published 8/12

Characterization and Greedy Learning of Interventional Markov Equivalence Classes of Directed Acyclic Graphs

Alain Hauser HAUSER@STAT.MATH.ETHZ.CH
Peter Bühlmann BUHLMANN@STAT.MATH.ETHZ.CH
Seminar für Statistik
ETH Zürich
8092 Zürich, Switzerland

Editor: Max Chickering

Abstract

The investigation of directed acyclic graphs (DAGs) encoding the same Markov property, that is the same conditional independence relations of multivariate observational distributions, has a long tradition; many algorithms exist for model selection and structure learning in Markov equivalence classes. In this paper, we extend the notion of Markov equivalence of DAGs to the case of interventional distributions arising from *multiple* intervention experiments. We show that under reasonable assumptions on the intervention experiments, interventional Markov equivalence defines a finer partitioning of DAGs than observational Markov equivalence and hence improves the identifiability of causal models. We give a graph theoretic criterion for two DAGs being Markov equivalent under interventions and show that each interventional Markov equivalence class can, analogously to the observational case, be uniquely represented by a chain graph called *interventional essential graph* (also known as *CPDAG* in the observational case). These are key insights for deriving a generalization of the Greedy Equivalence Search algorithm aimed at structure learning from interventional data. This new algorithm is evaluated in a simulation study.

Keywords: causal inference, interventions, graphical model, Markov equivalence, greedy equivalence search

1. Introduction

Directed acyclic graphs (or DAGs for short) are commonly used to model causal relationships between random variables; in such models, parents of some vertex in the graph are understood as "causes", and edges have the meaning of "causal influences". The causal influences between random variables imply conditional independence relations among them. However, those independence relations, or the corresponding Markov properties, do *not* identify the corresponding DAG completely, but only up to Markov equivalence. To put it simple, the skeleton of an underlying DAG is completely determined by its Markov property, whereas the *direction* of the arrows (which is crucial for causal interpretation) is in general not encoded in the Markov property for the observational distribution.

Interventions can help to overcome those limitations in identifiability. An *intervention* is realized by forcing the value of one or several random variables of the system to chosen values, destroying their original causal dependencies. The ensemble of both the observational and interventional distributions can greatly improve the identifiability of the causal structure of the system, the underlying DAG.

This paper has two main contributions. The first one is an algorithmically tractable graphical representation of Markov equivalence classes under a given set of interventions (possibly affecting several variables) from which the identifiability of causal models can be read off. This is of general interest for computation and algorithms dealing with structure (DAG) learning from an ensemble of observational and interventional data such as MCMC. The second contribution is a generalization of the Greedy Equivalence Search (GES) algorithm of Chickering (2002b), yielding an algorithm called Greedy Interventional Equivalence Search (GIES) which can be used for regularized maximum likelihood estimation in such an interventional setting.

In Section 2, we establish a criterion for two DAGs being Markov equivalent under a given intervention setting. We then generalize the concept of essential graphs, a graph theoretic representation of Markov equivalence classes, to the interventional case and characterize the properties of those graphs in Section 3. In Section 4, we elaborate a set of algorithmic operations to efficiently traverse the search space of interventional essential graphs and finally present the GIES algorithm. An experimental evaluation thereof is given in Section 5. We postpone all proofs to Appendix B, while Appendix A contains a review on graph theoretic concepts and definitions. An implementation of the GIES algorithm will be available in the next release of the R package `pcalg` (Kalisch et al., 2012); meanwhile, a prerelease version is available upon request from the first author.

1.1 Related Work

The investigation of Markov equivalence classes of directed graphical models has a long tradition, perhaps starting with the criterion for two DAGs being Markov equivalent by Verma and Pearl (1990) and culminating in the graph theoretic characterization of essential graphs (also called *CPDAGs*, "completed partially directed acyclic graphs") representing Markov equivalence classes by Andersson et al. (1997). Several algorithms for estimating essential graphs from observational data exist, such as the PC algorithm (Spirtes et al., 2000) or the Greedy Equivalence Search (GES) algorithm (Meek, 1997; Chickering, 2002b); a more complete overview is given in Brown et al. (2005) and Murphy (2001).

Different approaches to incorporate interventional data for learning causal models have been developed in the past. The Bayesian procedures of Cooper and Yoo (1999) or Eaton and Murphy (2007) address the problem of calculating a posterior (and also a likelihood) of an ensemble of observational and interventional data but do not address questions of identifiability or Markov equivalence: allowing different posteriors for Markov equivalent models can be intended in Bayesian methods (and realized by giving the corresponding models different priors). Since the number of DAGs with p variables grows super-exponentially with p (Robinson, 1973), the computation of a *full* posterior is intractable. For this reason, the mentioned Bayesian approaches are limited to computing posterior probabilities for certain features of a DAG; such a feature could be an edge from a vertex a to another vertex b, or a directed path from a to b visiting additional vertices. Approaches based on active learning (He and Geng, 2008; Tong and Koller, 2001; Eberhardt, 2008) propose an iterative line of action, estimating the essential graph with observational data in a first step and using interventional data in a second step to orient beforehand unorientable edges. He and Geng (2008) present a greedy procedure in which interventional data is uniquely used for deciding about edge orientations; this is not favorable from a statistical point of view since interventional data can also help to improve the estimation of the skeleton (or, more generally, the observational essential graph). Tong and Koller (2001) avoid this problem by using a Bayesian framework, but do not

address the issue of Markov equivalence therewith. Eberhardt et al. (2005) and Eberhardt (2008) provide algorithms for choosing intervention targets that *completely* identify *all* causal models of p variables uniformly, but neither address the question of partial identifiability under a limited number of interventions nor provide an algorithm for learning the causal structure from data. Eberhardt et al. (2010) present an algorithm for learning *cyclic* linear causal models, but focus on complete identifiability; identifiability results for cyclic models only imply *sufficient*, but not *necessary*, conditions for the identifiability of acyclic models.

Probably the most advanced result concerning identifiability of causal models under single-variable interventions so far is given in the work of Tian and Pearl (2001). Although they do not provide a characterization of equivalence classes as a whole (as this paper does), they present a necessary and sufficient graph theoretic criterion for two models being indistinguishable under a set of single-variable interventions as well as a learning algorithm based on the detection of changes in marginal distributions.

2. Model

We consider p random variables $(X_1,\ldots,X_p) =: X$ which take values in some product measure space $(\mathcal{X},\mathcal{A},\mu) = (\prod_{i=1}^p \mathcal{X}_i, \bigotimes_{i=1}^p \mathcal{A}_i, \bigotimes_{i=1}^p \mu_i)$ with $\mathcal{X}_i \subset \mathbb{R}\ \forall\ i$. Each σ-algebra $\mathcal{A}_i$ is assumed to contain at least two disjoint sets of positive measure to avoid pathologies, and X is assumed to have a strictly positive joint density w.r.t. the measure μ on $\mathcal{X}$. We denote the set of all positive densities on $\mathcal{X}$ by $\mathcal{M}$. For any subset of component indices $A \subset [p] := \{1,\ldots,p\}$, we use the notation $\mathcal{X}_A := \prod_{a\in A} \mathcal{X}_a$, $X_A := (X_a)_{a\in A}$ and the convention $X_\emptyset \equiv 0$. Lowercase symbols like x_A represent a value in $\mathcal{X}_A$.

The model we are considering is built upon Markov properties with respect to DAGs. By convention, all graphs appearing in the paper shall have the vertex set $[p]$, representing the p random variables $X_1,\ldots,X_p$. Our notation and definitions related to graphs are summarized in Appendix A.1.

2.1 Causal Calculus: A Short Review

We start by summarizing important facts and fixing our notation concerning Markov properties and intervention calculus.

Definition 1 (Markov property; Lauritzen, 1996) *Let D be a DAG. Then we say that a probability density $f \in \mathcal{M}$ **obeys the Markov property of** D if $f(x) = \prod_{i=1}^p f(x_i|x_{\mathrm{pa}_D(i)})$. The set of all positive densities obeying the Markov property of D is denoted by $\mathcal{M}(D)$.*

Definition 1 is the most straightforward translation of independence relations induced from structural equations, the historical origin of directed graphical models (Wright, 1921). Related notions like local and global Markov properties exist and are equivalent to the factorization property of Definition 1 for positive densities (Lauritzen, 1996).

Definition 2 (Markov equivalence; Andersson et al., 1997) *Let D_1 and D_2 be two DAGs. D_1 and D_2 are called **Markov equivalent** (notation: $D_1 \sim D_2$) if $\mathcal{M}(D_1) = \mathcal{M}(D_2)$.*

Theorem 3 (Verma and Pearl, 1990) *Two DAGs D_1 and D_2 are Markov-equivalent if and only if they have the same skeleton and the same v-structures.*

Directed graphical models allow for an obvious causal interpretation. For a density f that obeys the Markov properties of some DAG D, we can think of a random variable X_a being the *direct cause* of another variable X_b if a is a parent of b in D.

Definition 4 (Causal model) *A **causal model** is a pair (D, f), where D is a DAG on the vertex set $[p]$ and $f \in \mathcal{M}(D)$ is a density obeying the Markov property of D: D is called the **causal structure** of the model, and f the **observational density**.*

Causality is strongly linked to interventions. We consider **stochastic interventions** (Korb et al., 2004) modeling the effect of setting or forcing one or several random variables X_I, where $I \subset [p]$ is called the **intervention target**, to the value of *independent* random variables U_I, called **intervention variables**. The joint product density of U_I on $\mathcal{X}_I$, called **level density**, is denoted by $\tilde{f}$. Extending the do() operator (Pearl, 1995) to stochastic interventions, we denote the density of X under such an intervention by $f(x|\operatorname{do}_D(X_I = U_I))$. Using truncated factorization and the assumption of independent intervention variables, this **interventional density** can be written as

$$f(x \mid \operatorname{do}_D(X_I = U_I)) = \prod_{i \notin I} f(x_i | x_{\mathrm{pa}_D(i)}) \prod_{i \in I} \tilde{f}(x_i) \ . \tag{1}$$

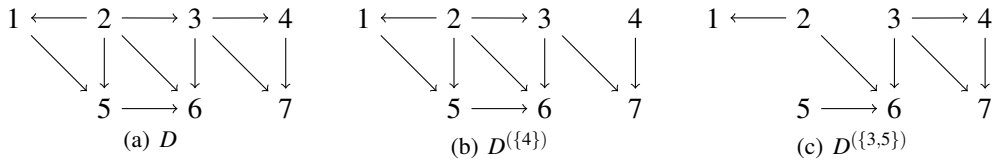
By denoting with $I = \emptyset$ and using the convention $f(x|\operatorname{do}(X_\emptyset = U_\emptyset)) = f(x)$, we also encompass the observational case as an intervention target.

Definition 5 (Intervention graph) *Let $D = ([p], E)$ be a DAG with vertex set $[p]$ and edge set E (see Appendix A.1), and $I \subset [p]$ an intervention target. The **intervention graph** of D is the DAG $D^{(I)} = ([p], E^{(I)})$, where $E^{(I)} := \{(a,b) \mid (a,b) \in E, b \notin I\}$.*

For a causal model (D, f), an interventional density $f(\cdot|\operatorname{do}_D(X_I = U_I))$ obeys the Markov property of $D^{(I)}$: the Markov property of the observational density is inherited. Figure 1 shows an example of a DAG and two corresponding intervention graphs.

As foreshadowed in the introduction, we are interested in causal inference based on data sets originating from *multiple* interventions, that means from a set of the form $\mathcal{S} = \{(I_j, \tilde{f}_j)\}_{j=1}^J$, where $I_j \subset [p]$ is an intervention target and $\tilde{f}_j$ a level density on $\mathcal{X}_{I_j}$ for $1 \le j \le J$. We call such a set an **intervention setting**, and the corresponding (multi)set of intervention targets $\mathcal{I} = \{I_j\}_{j=1}^J$ a **family of targets**. We often use the family of targets as an index set, for example to write a corresponding intervention setting as $\mathcal{S} = \{(I, \tilde{f}_I)\}_{I \in \mathcal{I}}$.

We consider **interventional data** of sample size n produced by a causal model (D, f) under an intervention setting $\mathcal{S} = \{(I, \tilde{f}_I)\}_{I \in \mathcal{I}}$. We assume that the n samples $X^{(1)}, \ldots, X^{(n)}$ are independent, and write them as usual as rows of a **data matrix X**. However, they are *not* identically distributed



(a) D (b) $D^{(\{4\})}$ (c) $D^{(\{3,5\})}$

Figure 1: A DAG D and the corresponding intervention graphs $D^{(\{4\})}$ and $D^{(\{3,5\})}$.

as they arise from *different* interventions. The interventional data set is fully specified by the pair $(\mathcal{T}, \mathbf{X})$,

$$\mathcal{T} = \begin{pmatrix} T^{(1)} \\ \vdots \\ T^{(n)} \end{pmatrix} \in \mathcal{I}^n, \quad \mathbf{X} = \begin{pmatrix} —X^{(1)}— \\ \vdots \\ —X^{(n)}— \end{pmatrix}, \tag{2}$$

where for each $i \in [n]$, $T^{(i)}$ denotes the intervention target under which the sample $X^{(i)}$ was produced. This data set can potentially contain observational data as well, namely if $\emptyset \in \mathcal{I}$. To summarize, we consider the statistical model

$$\begin{aligned} &X^{(1)}, X^{(2)}, \ldots, X^{(n)} \text{ independent,} \\ &X^{(i)} \sim f\big(\cdot \mid \mathrm{do}_D(X^{(i)}_{T^{(i)}} = U_{T^{(i)}})\big),\ U_{T^{(i)}} \sim \tilde{f}_{T^{(i)}}, \quad i = 1, \ldots, n\,, \end{aligned} \tag{3}$$

and we assume that each target $I \in \mathcal{I}$ appears at least once in the sequence $\mathcal{T}$.

2.2 Interventional Markov Equivalence: New Concepts and Results

An intervention at some target $a \in [p]$ destroys the original causal influence of other variables of the system on X_a. Interventional data thereof can hence not be used to determine the causal parents of X_a in the (undisturbed) system. To be able to estimate at least the complete skeleton of a causal structure (as in the observational case), an intervention experiment has to be performed based on a *conservative* family of targets:

Definition 6 (Conservative family of targets) *A family of targets $\mathcal{I}$ is called **conservative** if for all $a \in [p]$, there is some $I \in \mathcal{I}$ such that $a \notin I$.*

In this paper, we restrict our considerations to *conservative* families of targets; see Section 2.3 for a more detailed discussion. Note that every experiment in which we also measure observational data corresponds to a conservative family of targets.

If a family of targets $\mathcal{I}$ contains more than one target, interventional data as in Equation (3) are *not* identically distributed. Whereas the distribution of observational data is determined by a *single* density, we need *tuples* of densities as in the following definition to specify the distribution of interventional data.

Definition 7 *Let D be a DAG on $[p]$, and let $\mathcal{I}$ be a family of targets. Then we define*

$$\begin{aligned} \mathcal{M}_{\mathcal{I}}(D) := \big\{ (f^{(I)})_{I \in \mathcal{I}} \in \mathcal{M}^{|\mathcal{I}|} \,\big|\, &\forall\, I \in \mathcal{I} : f^{(I)} \in \mathcal{M}(D^{(I)}),\ \textit{and} \\ &\forall\, I, J \in \mathcal{I},\ \forall\, a \notin I \cup J : f^{(I)}(x_a | x_{\mathrm{pa}_D(a)}) = f^{(J)}(x_a | x_{\mathrm{pa}_D(a)}) \big\}\,. \end{aligned}$$

Although the do() operator does not appear in Definition 7, the elements in $\mathcal{M}_{\mathcal{I}}(D)$ are exactly the tuples $(f(\cdot|\,\mathrm{do}_D(X_I = U_I)))_{I \in \mathcal{I}}$ that can be realized as interventional densities of some causal model (D, f). The first condition in the definition reflects the fact that an intervention at a target I generates a density obeying the Markov property of $D^{(I)}$; the second condition is a consequence of the truncated factorization in Equation (1). These considerations are formalized in the following lemma and motivate Definition 9 of interventional Markov equivalence in analogy to the observational case. Note that for $\mathcal{I} = \{\emptyset\}$, Definition 7 equals its observational counterpart: $\mathcal{M}_{\{\emptyset\}}(D) = \mathcal{M}(D)$ (see Definition 1).

Lemma 8 *Let D be a DAG on $[p]$, and $\mathcal{I}$ a conservative family of targets.*

(i) Let (D, f) be a causal model (that is, $f \in \mathcal{M}(D)$), $\mathcal{S} = \{(I, \tilde{f}_I)\}_{I\in\mathcal{I}}$ an intervention setting and $U_I \sim \tilde{f}_I$ intervention variables for $I \in \mathcal{I}$. Then, we have

$$\big(f(\cdot \mid \mathrm{do}(X_I = U_I))\big)_{I\in\mathcal{I}} \in \mathcal{M}_{\mathcal{I}}(D) \,.$$

(ii) Let $(f^{(I)})_{I\in\mathcal{I}} \in \mathcal{M}_{\mathcal{I}}(D)$. Then there is some positive density $f \in \mathcal{M}(D)$ and an intervention setting $\mathcal{S} = \{(I, \tilde{f}_I)\}_{I\in\mathcal{I}}$ such that $f(\cdot|\,\mathrm{do}(X_I = U_I)) = f^{(I)}(\cdot)$ for random variables U_I with density $\tilde{f}_I$, for all $I \in \mathcal{I}$.

Definition 9 (Interventional Markov equivalence) *Let D_1 and D_2 be DAGs, and $\mathcal{I}$ a family of targets. D_1 and D_2 are called **$\mathcal{I}$-Markov equivalent** (notation: $D_1 \sim_{\mathcal{I}} D_2$) if $\mathcal{M}_{\mathcal{I}}(D_1) = \mathcal{M}_{\mathcal{I}}(D_2)$. The $\mathcal{I}$-Markov equivalence class of a DAG D is denoted by $[D]_{\mathcal{I}}$.*

Alternatively, we will also use the term "interventionally Markov equivalent" when it is clear which family of targets is meant. For the simplest conservative family of targets, $\mathcal{I} = \{\emptyset\}$, we get back Definition 2 for the observational case. We now generalize Theorem 3 for the interventional case in order to get a purely graph theoretic criterion for interventional Markov equivalence of two given DAGs, the main result of this section.

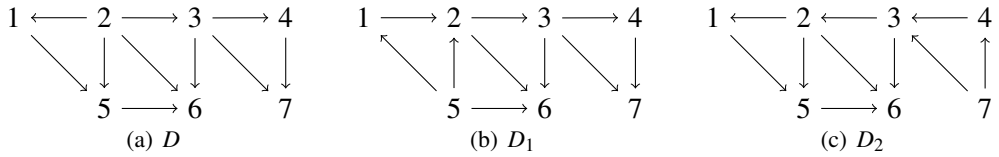
Theorem 10 *Let D_1 and D_2 be two DAGs on $[p]$, and $\mathcal{I}$ a conservative family of targets. Then, the following statements are equivalent:*

(i) $D_1 \sim_{\mathcal{I}} D_2$;
(ii) for all $I \in \mathcal{I}$, $D_1^{(I)} \sim D_2^{(I)}$ (in the observational sense);
(iii) for all $I \in \mathcal{I}$, $D_1^{(I)}$ and $D_2^{(I)}$ have the same skeleton and the same v-structures;
(iv) D_1 and D_2 have the same skeleton and the same v-structures, and $D_1^{(I)}$ and $D_2^{(I)}$ have the same skeleton for all $I \in \mathcal{I}$.

2.3 Discussion

Throughout this paper, we always assume the observational density f of a causal model to be strictly positive. This assumption makes sure that the conditional densities in Equation (1) are well-defined. The requirement of a strictly positive density can, however, be a restriction for example for discrete models (where the density is with respect to the counting measure). In the observational case, the notion of Markov equivalence remains the same when we also allow densities that are not strictly positive (Lauritzen, 1996). We conjecture that the notion of interventional Markov equivalence (Definition 9 and Theorem 10) also remains valid for such densities; corresponding proofs would, however, require more caution to avoid the aforementioned problems with (truncated) factorization.

To illustrate the importance of a conservative family of targets for structure identification, let us consider the simplest non-trivial example of a causal model with 2 variables X_1 and X_2. Under observational data, we can distinguish two Markov equivalence classes: one in which the variables are independent (represented by the empty DAG D_0), and one in which they are not independent (represented by the DAGs $D_1 := 1 \longrightarrow 2$ and $D_2 := 1 \longleftarrow 2$). D_1 and D_2 can be distinguished if we can measure data from an intervention at one of the vertices in addition to observational data; this experimental setting corresponds to the (conservative) family of targets $\mathcal{I} = \{\emptyset, \{1\}\}$. However, an



(a) D (b) D_1 (c) D_2

Figure 2: Three DAGs having equal skeletons and a single v-structure, $3 \longrightarrow 6 \longleftarrow 5$, hence being observationally Markov equivalent. For $\mathcal{I} = \{\emptyset, \{4\}\}$, we have $D \sim_{\mathcal{I}} D_1$, but $D \not\sim_{\mathcal{I}} D_2$ since the skeletons of $D^{(\{4\})}$ (Figure 1(b)) and $D_2^{(\{4\})}$ do not coincide.

intervention at, say, X_1 *alone* (that is, in the *absence* of observational data), corresponding to the non-conservative family $\mathcal{I} = \{\{1\}\}$, only allows a distinction between the models D_2 and D_0 on the one hand (which do not show dependence between X_1 and X_2 under the intervention) and D_1 on the other hand (which does show dependence between X_1 and X_2 under the intervention). Note that the two indistinguishable models D_0 and D_2 do not even have the same skeleton, and that it is impossible to determine the influence of X_2 on X_1 in the undisturbed system. In this setting, it would be more natural to consider the intervened variable X_1 as an external parameter rather than a random variable of the system, and to perform regression to detect or determine the influence of X_1 on X_2. Note, however, that full identifiability of the models does *not* require observational data; interventions at X_1 and X_2 (corresponding to the conservative family $\mathcal{I} = \{\{1\}, \{2\}\}$ in our notation) are also sufficient.

Theorem 10 is of great importance for the description of Markov equivalence classes under interventions. It shows that two DAGs which are interventionally Markov equivalent under some conservative family of targets are also observationally Markov equivalent:

$$D_1 \sim_{\mathcal{I}} D_2 \Rightarrow D_1 \sim D_2. \tag{4}$$

This implication is *not* true anymore for non-conservative families of targets. This is an explanation for the term "conservative": a conservative family of targets yields a finer partitioning of DAGs into equivalence classes compared to observational Markov equivalence, but it preserves the "borders" of observational Markov equivalence classes. Figure 2 shows three DAGs that are observationally Markov equivalent, but which fall into two different interventional Markov equivalence classes under the family of targets $\mathcal{I} = \{\emptyset, \{4\}\}$.

Theorem 10 agrees with Theorem 3 of Tian and Pearl (2001) for single-variable interventions. While we also make a statement about interventions at *several* variables, they prove their theorem for perturbations of the system at single variables only, but for a wider class of perturbations called **mechanism changes** that go beyond our notion of interventions. While an *intervention* destroys the causal dependence of a variable from its parents (and hence replaces a conditional density by a marginal one in the Markov factorization, see Equation (1)), a *mechanism change* (also known as "imperfect" or "soft" interventions; see Eaton and Murphy, 2007) alters the functional form of this dependence (and hence replaces a Markov factor by a different one which is still a conditional distribution). The fact that Theorem 10 is true for mechanism changes on single variables motivates the conjecture that it also holds for mechanism changes on *several* variables.

3. Essential Graphs

Theorem 10 represents a computationally fast criterion for deciding whether two DAGs are interventionally Markov equivalent or not. However, given some DAG D, it does not provide a possibility for quickly finding *all* equivalent ones, and hence does not specify the equivalence class as a whole. In this section, we give a characterization of graphs that uniquely represent an interventional Markov equivalence class (Theorem 18). Our characterization of these *interventional essential graphs* is inspired by and similar to the one developed by Andersson et al. (1997) for the observational case and allows for handling equivalence classes algorithmically. Furthermore, we present a linear time algorithm for constructing a representative of the equivalence class corresponding to an interventional essential graph (Proposition 16 and discussion thereafter), as well as a polynomial time algorithm for constructing the interventional essential graph of a given DAG (Algorithm 1). Throughout this section, $\mathcal{I}$ always stands for a conservative family of targets.

3.1 Definitions and Motivation

All DAGs in an $\mathcal{I}$-Markov equivalence class share the same skeleton; however, arrow orientations may vary between different representatives (Theorem 10). Varying and common arrow orientations are represented by undirected and directed edges, respectively, in $\mathcal{I}$-essential graphs.

Definition 11 ($\mathcal{I}$-essential graph) *Let D be a DAG. The **$\mathcal{I}$-essential graph** of D is defined as $\mathcal{E}_{\mathcal{I}}(D) := \bigcup_{D' \in [D]_{\mathcal{I}}} D'$. (The union is meant in the graph theoretic sense, see Appendix A.1).*

When the family of targets $\mathcal{I}$ in question is clear from the context, we will also use the term **interventional essential graph**, while "observational essential graph" shall refer to the concept of essential graphs as introduced by Andersson et al. (1997) in the observational case. Simply speaking of "essential graphs", we mean interventional or observational essential graphs in the following.

Definition 12 ($\mathcal{I}$-essential arrow) *Let D be a DAG. An edge $a \longrightarrow b \in D$ is **$\mathcal{I}$-essential** in D if $a \longrightarrow b \in D' \; \forall \; D' \in [D]_{\mathcal{I}}$.*

An $\mathcal{I}$-essential graph typically contains directed as well as undirected edges. Directed ones correspond to arrows that are $\mathcal{I}$-essential in every representative of the equivalence class; in other words, $\mathcal{I}$-essential arrows are those whose direction is identifiable. A first sufficient criterion for an edge to be $\mathcal{I}$-essential follows immediately from Lemma 47 (Appendix B.1).

Corollary 13 *Let D be a DAG with $a \longrightarrow b \in D$. If there is an intervention target $I \in \mathcal{I}$ such that $|\{a,b\} \cap I| = 1$, then $a \longrightarrow b$ is $\mathcal{I}$-essential.*

The investigation of essential graphs has a long tradition in the observational case (Andersson et al., 1997; Chickering, 2002a). Due to increased identifiability of causal structures, Markov equivalence classes shrink in the interventional case; Equation (4) implies $\mathcal{E}_{\mathcal{I}}(D) \subset \mathcal{E}_{\{\emptyset\}}(D)$ for any conservative family of targets $\mathcal{I}$ (see also Figure 8 in Section 5). Essential graphs, interventional as well as observational ones, are mainly interesting because of two reasons:

- It is important to know which arrow directions of a causal model are identifiable and which are not since arrow directions are relevant for the causal interpretation.

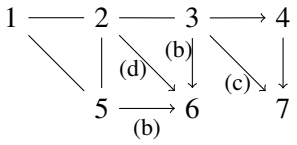
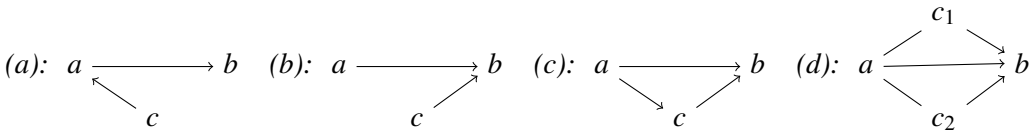


Figure 3: A graph with six arrows. Four of them are strongly $\mathcal{I}$-protected for any conservative family of targets $\mathcal{I}$ (in parentheses: arrow configurations according to Definition 14). Arrows $3 \longrightarrow 4$ and $4 \longrightarrow 7$ are strongly $\mathcal{I}$-protected for $\mathcal{I} = \{\emptyset, \{4\}\}$, but not for $\mathcal{I} = \{\emptyset\}$.

- Markov equivalent DAGs encode the same statistical model. Hence the space of DAGs is no suitable "parameter" or search space for statistical inference and computation. The natural search space is given by the set of the equivalence classes, the objects that can be distinguished from data. Essential graphs uniquely represent these equivalence classes and are efficiently manageable in algorithms.

The characterization of $\mathcal{I}$-essential graphs (Theorem 18) relies on the notion of strongly $\mathcal{I}$-protected arrows (Definition 14) which reproduces the corresponding definition of Andersson et al. (1997) for $\mathcal{I} = \{\emptyset\}$; an illustration is given in Figure 3.

Definition 14 (Strong protection) *Let G be a graph. An arrow $a \longrightarrow b \in G$ is **strongly $\mathcal{I}$-protected** in G if there is some $I \in \mathcal{I}$ such that $|I \cap \{a,b\}| = 1$, or the arrow $a \longrightarrow b$ occurs in at least one of the following four configurations as an induced subgraph of G:*



We will see in Theorem 18 that every arrow of an $\mathcal{I}$-essential graph (that is, every edge corresponding to an $\mathcal{I}$-essential arrow in the representative DAGs) is strongly $\mathcal{I}$-protected. The configurations in Definition 14 guarantee the identifiability of the edge orientation between a and b: if there is a target $I \in \mathcal{I}$ such that $|I \cap \{a,b\}| = 1$, turning the arrow would change the skeleton of the intervention graph $D^{(I)}$ (see also Corollary 13); in configuration (a), reversal would create a new v-structure; in (b), reversal would destroy a v-structure; in (c), reversal would create a cycle; an in (d) finally, at least one of the arrows between a and c_1 or c_2 must point away from a in each representative, hence turning the arrow $a \longrightarrow b$ would create a cycle. We refer to Andersson et al. (1997) for a more detailed discussion of the configurations (a) to (d).

3.2 Characterization of Interventional Essential Graphs

As in the observational setting, we can show that interventional essential graphs are chain graphs with chordal chain components (see Appendix A.1). For the observational case $\mathcal{I} = \{\emptyset\}$, Propositions 15 and 16 below correspond to Propositions 4.1 and 4.2 of Andersson et al. (1997).

Proposition 15 *Let D be a DAG on $[p]$. Then:*

(i) $\mathcal{E}_{\mathcal{I}}(D)$ is a chain graph.

(ii) For each chain component $T \in \mathbf{T}(\mathcal{E}_\mathcal{I}(D))$, the induced subgraph $\mathcal{E}_\mathcal{I}(D)[T]$ is chordal.

Proposition 16 *Let D be a DAG. A digraph D' is acyclic and $\mathcal{I}$-equivalent to D if and only if D' can be constructed by orienting the edges of every chain component of $\mathcal{E}_\mathcal{I}(D)$ according to a perfect elimination ordering.*

This proposition is not only of theoretic, but also of algorithmic interest. According to the explanation in Appendix A.2, perfect elimination orderings on the (chordal) chain components of $\mathcal{E}_\mathcal{I}(D)$ can be generated with LexBFS (Algorithm 6); doing this for all chain components yields computational complexity $O(|E|+p)$, where E denotes the edge set of $\mathcal{E}_\mathcal{I}(D)$ (see Appendix A.2).

As an immediate consequence of Proposition 16, interventional essential graphs are in one-to-one correspondence with interventional Markov equivalence classes. We will therefore also speak about "representatives of $\mathcal{I}$-essential graphs", where we mean representatives (that is, DAGs) of the corresponding equivalence class. Propositions 15 and 16 give the justification for the following definition; note that in order to generate a representative of some $\mathcal{I}$-essential graph, the family of targets $\mathcal{I}$ need not be known.

Definition 17 *Let G be the $\mathcal{I}$-essential graph of some DAG. The set of representatives of G is denoted by $\mathbf{D}(G)$:*

$$\mathbf{D}(G) := \{D \text{ a DAG} \mid D \subset G, D^u = G^u, D[T] \text{ oriented according to some perfect elimination ordering for each chain component } T \in \mathbf{T}(G)\}.$$

Here, D^u denotes the skeleton of D (Appendix A.1). We can now state the main result of this section, a graph theoretic characterization of $\mathcal{I}$-essential graphs. For the observational case $\mathcal{I} = \{\emptyset\}$, this theorem corresponds to Theorem 4.1 of Andersson et al. (1997).

Theorem 18 *A graph G is the $\mathcal{I}$-essential graph of a DAG D if and only if*

(i) G is a chain graph;
(ii) for each chain component $T \in \mathbf{T}(G)$, $G[T]$ is chordal;
(iii) G has no induced subgraph of the form $a \longrightarrow b - c$;
(iv) G has no line $a - b$ for which there exists some $I \in \mathcal{I}$ such that $|I \cap \{a,b\}| = 1$;
(v) every arrow $a \longrightarrow b \in G$ is strongly $\mathcal{I}$-protected.

The graph G of Figure 3 satisfies points (i) to (iii) of Theorem 18. For $\mathcal{I} = \{\emptyset, \{4\}\}$, it also fulfills points (iv) and (v); in this case, it is the $\mathcal{I}$-essential graph $\mathcal{E}_\mathcal{I}(D)$ of the DAG D of Figure 1(a) by Proposition 16.

3.3 Construction of Interventional Essential Graphs

In this section, we show that there is a simple way to construct the $\mathcal{I}$-essential graph $\mathcal{E}_\mathcal{I}(D)$ of a DAG D: we need to successively convert arrows that are not strongly $\mathcal{I}$-protected into lines (Algorithm 1). By doing this, we get a sequence of partial $\mathcal{I}$-essential graphs.

Definition 19 (Partial $\mathcal{I}$-essential graph) *Let D be a DAG. A graph G with $D \subset G \subset \mathcal{E}_\mathcal{I}(D)$ is called a **partial $\mathcal{I}$-essential graph** of D if $a \longrightarrow b - c$ does not occur as an induced subgraph of G.*

The following lemma can be understood as a motivation for looking at such graphs. Note that due to the condition $G \subset \mathcal{E}_{\mathcal{I}}(D)$, and because G and $\mathcal{E}_{\mathcal{I}}(D)$ have the same skeleton, every arrow of $\mathcal{E}_{\mathcal{I}}(D)$ is also present in G, hence statement (ii) below makes sense.

Lemma 20 *Let D be a DAG. Then:*

(i) D and $\mathcal{E}_{\mathcal{I}}(D)$ are partial $\mathcal{I}$-essential graphs of D.
(ii) Let G be a partial $\mathcal{I}$-essential graph of D. Every arrow $a \longrightarrow b \in \mathcal{E}_{\mathcal{I}}(D)$ is strongly $\mathcal{I}$-protected in G.
(iii) Let G be a partial $\mathcal{I}$-essential graph of two DAGs D_1 and D_2. Then, $D_1 \sim_{\mathcal{I}} D_2$.

Algorithm 1 constructs the $\mathcal{I}$-essential graph G from a partial $\mathcal{I}$-essential graph of any DAG $D \in \mathbf{D}(G)$. The algorithm is indeed valid and calculates $\mathcal{E}_{\mathcal{I}}(D)$, since the graph produced in each iteration is a partial $\mathcal{I}$-essential graph of D (Lemma 21), and the only partial $\mathcal{I}$-essential graph that has only strongly $\mathcal{I}$-protected arrows is $\mathcal{E}_{\mathcal{I}}(D)$ (Lemma 22).

Lemma 21 *Let D be a DAG and G a partial $\mathcal{I}$-essential graph of D. Assume that $a \longrightarrow b \in G$ is not strongly $\mathcal{I}$-protected in G, and let $G' := G + (b,a)$ (that is, the graph we get by replacing the arrow $a \longrightarrow b$ by a line $a - b$; see Appendix A.1). Then G' is also a partial $\mathcal{I}$-essential graph of D.*

Lemma 22 *Let D be a DAG. There is exactly one partial $\mathcal{I}$-essential graph of D in which every arrow is strongly $\mathcal{I}$-protected, namely $\mathcal{E}_{\mathcal{I}}(D)$.*

To construct $\mathcal{E}_{\mathcal{I}}(D)$ from some DAG $D = ([p], E)$, we must, in the worst case, execute the iteration of Algorithm 1 for every arrow in the DAG; at each step, we must check every 4-tuple of vertices to see whether some arrow occurs in configuration (d) of Definition 14. Therefore Algorithm 1 has at most complexity $O(|E| \cdot p^4)$; by exploiting the partial order $\preceq_G$ on $\mathbf{T}(G)$ (see Appendix A.1), more efficient implementations are possible. Note that some checks only need to be done once. If, for example, an edge $a \longrightarrow b$ is part of a v-structure (configuration (b) of Definition 14), or if there is some $I \in \mathcal{I}$ such that $|I \cap \{a,b\}| = 1$ in the first iteration of Algorithm 1, this will also be the case in every later iteration.

3.4 Example: Identifiability under Interventions

A simple example illustrates how much identifiability can be gained with a single intervention. We consider a linear chain as observational essential graph:

$$G = \mathcal{E}_{\{\emptyset\}}(D) : 1 - 2 - 3 - \cdots - p \ .$$

We can easily count the number of representatives of G using the following lemma.

Input : G: partial $\mathcal{I}$-essential graph of some DAG D (not known)
Output: $\mathcal{E}_{\mathcal{I}}(D)$
while $\exists\ a \longrightarrow b \in G$ *s.t.* $a \longrightarrow b$ *not strongly $\mathcal{I}$-protected in G* **do**
  $G \leftarrow G + (b,a)$;
return G;

Algorithm 1: REPLACEUNPROTECTED$(\mathcal{I}, G)$. Iterative construction of an $\mathcal{I}$-essential graph

Lemma 23 (Source lemma) *Let G be a connected, chordal, undirected graph, and let $D \subset G$ be a DAG without v-structures and with $D^u = G$. Then D has exactly one source.*

Proof Let σ be a topological ordering of D; then, $\sigma(1)$ is a source, see Appendix A.1. It remains to show that there is at most one such source. Assume, for the sake of contradiction, that there are two different sources u and v. Since G is connected, there is a shortest u-v-path $\gamma = (a_0 \equiv u, a_1, \ldots, a_k \equiv v)$. Let $a_i \leftarrow\!\!- a_{i+1} \in D$ be the first arrow that points away from v in the chain γ in D (note $i \geq 1$ since $u \longrightarrow a_1 \in D$ by assumption). The v-structure $a_{i-1} \longrightarrow a_i \leftarrow\!\!- a_{i+1}$ is not allowed as an induced subgraph of D, hence a_{i-1} and a_{i+1} must be adjacent in D and in G; however, γ is then no *shortest* u-v-path, a contradiction. ■

For our linear chain G and any $s \in [p]$, there is exactly one DAG $D \in \mathbf{D}(G)$ that has the (unique) source s, namely the DAG we get by orienting *all* edges of G away from s; other edge orientations would produce a v-structure. We conclude G has p representatives.

Assume that the true causal model producing the data is (D, f), and denote the source of D by $s \in [p]$. Consider the conservative family of targets $\mathcal{I} = \{\emptyset, \{v\}\}$ with $v \in [p]$. If $v < s$, the interventional essential graph $\mathcal{E}_\mathcal{I}(D)$ is

$$1 \leftarrow\!\!- 2 \leftarrow\!\!- \ldots \leftarrow\!\!- v+1 \,\text{—}\, \ldots \,\text{—}\, p \,,$$

and $|\mathbf{D}(\mathcal{E}_\mathcal{I}(D))| = p - v$ by the same arguments as above; analogously, if $v > s$, we find $|\mathbf{D}(\mathcal{E}_\mathcal{I}(D))| = v - 1$. On the other hand, if $v = s$, all edges of D are strongly $\mathcal{I}$-protected: those incident to s because of the intervention target, all others because they are in configuration (a) of Definition 14; therefore, we have $\mathcal{E}_\mathcal{I}(D) = D$.

In the best case, all edge orientations in the chain can be identified by a single intervention, while the observational essential graph $\mathcal{E}_{\{\emptyset\}}(D)$ that is identifiable from observational data alone contains p representatives. However, this needs an intervention at the a priori unknown source s. Choosing the central vertex $\lceil \frac{p}{2} \rceil$ as intervention target ensures that at least half of the edges become directed in $\mathcal{E}_\mathcal{I}(D)$, independent of the position s of the source.

4. Greedy Interventional Equivalence Search

Different algorithms have been proposed to estimate essential graphs from observational data. One of them, the Greedy Equivalence Search (GES) (Meek, 1997; Chickering, 2002b), is particularly interesting because of two properties:

- It is score-based; it greedily maximizes some score function for given data over essential graphs. It uses no tuning-parameter; the score function alone measures the quality of the estimate. Chickering (2002b) chose the BIC score because of consistency; technically, any score equivalent and decomposable function (see Definition 24) is adequate.

- It traverses the space of essential graphs which is the natural search space for model inference (see Section 3). We will see in Section 5 that a greedy search over *equivalence classes* yields much better estimation results than a naïve greedy search over *DAGs*.

GES greedily optimizes the score function in two phases (Chickering, 2002b):

- In the **forward phase**, the algorithm starts with the empty essential graph, $G_0 := ([p], \emptyset)$. It then sequentially steps from one essential graph G_i to a *larger* one, G_{i+1}, for which there are representatives $D_i \in \mathbf{D}(G_i)$ and $D_{i+1} \in \mathbf{D}(G_{i+1})$ such that D_{i+1} has exactly one arrow more than D_i.

- In the **backward phase**, the sequence $(G_i)_i$ is continued by gradually stepping from one essential graph G_i to a *smaller* one, G_{i+1}, for which there are representatives $D_i \in \mathbf{D}(G_i)$ and $D_{i+1} \in \mathbf{D}(G_{i+1})$ such that D_{i+1} has exactly one arrow less than D_i.

In both phases, the respective candidate with maximal score is chosen, or the phase is aborted if no candidate scores higher than the current essential graph G_i.

We introduce in addition a new turning phase which proved to enhance estimation (see Section 5). Here, the sequence $(G_i)_i$ is elongated by gradually stepping from one essential graph G_i to a new one with the same number of edges, denoted by G_{i+1}, for which there are representatives $D_i \in \mathbf{D}(G_i)$ and $D_{i+1} \in \mathbf{D}(G_{i+1})$ such that D_{i+1} can be constructed from D_i by turning exactly one arrow. As before, we choose the highest scoring candidate. Such a turning phase had already been proposed, but not characterized or implemented, by Chickering (2002b).

Because GES is an optimization algorithm working on the space of observational essential graphs, and because the characterization of *interventional* essential graphs is similar to that of observational ones (Theorem 18), GES can indeed be generalized to handle interventional data as well by operating on interventional instead of observational essential graphs. We call this generalized algorithm *Greedy Interventional Equivalence Search* or GIES. An overview is shown in Algorithm 2: the forward, backward and turning phase are repeatedly executed in this order until none of them can augment the score function any more.

A naïve search strategy would perhaps traverse the space of DAGs instead of essential graphs, greedily adding, removing or turning single arrows from DAGs. It is well-known in the observational case that such an approach performs markedly worse than one accounting for Markov equivalence (Chickering, 2002b; Castelo and Kočka, 2003), and we will see in our simulations (Section 5.2) that the same is true in the interventional case as long as few interventions are made. Ignoring Markov equivalence cuts down the search space of successors at haphazard; since all DAGs in a Markov equivalence class represent the same statistical model, there is no justification for considering neighbors (that is, DAGs that can be reached by adding, removing or turning an arrow) of one of the representatives but not of the other ones.

GIES can be used with general score functions. It goes without saying that the chosen score function should be a "reasonable" one which has favorable statistical properties such as consistency. We denote the score of a DAG D given interventional data $(\mathcal{T}, \mathbf{X})$ by $S(D; \mathcal{T}, \mathbf{X})$, and we assume that S is **score equivalent**, that is, it assigns the same score to $\mathcal{I}$-equivalent DAGs; $\mathcal{I}$ always stands for a conservative family of targets in this section. Furthermore, we require S to be decomposable.

Definition 24 *A score function S is called **decomposable** if for each DAG D, S can be written as a sum*

$$S(D; \mathcal{T}, \mathbf{X}) = \sum_{i=1}^{p} s(i, \mathrm{pa}_D(i); \mathcal{T}, \mathbf{X}),$$

*where the **local score** s depends on $\mathbf{X}$ only via $\mathbf{X}_{\cdot i}$ and $\mathbf{X}_{\cdot \mathrm{pa}_D(i)}$, with $\mathbf{X}_{\cdot i}$ denoting the i^{th} column of $\mathbf{X}$ and $\mathbf{X}_{\cdot \mathrm{pa}_D(i)}$ the submatrix of $\mathbf{X}$ corresponding to the columns with index in $\mathrm{pa}_D(i)$.*

Throughout the rest of this section, S always denotes a score equivalent and decomposable score function. Such a score function needs only be evaluated at one single representative of some interventional Markov equivalence class. Indeed, a key ingredient for the efficiency of the observational GES as well as our interventional GIES is an implementation that computes the greedy steps to the next equivalence class in a local fashion without enumerating all corresponding DAG members. Chickering (2002b) found a clever way to do that in the forward and backward phase of the observational GES. In Sections 4.1 and 4.2, we generalize his methods to the interventional case, and in Section 4.3, we propose an efficient implementation of the new turning phase.

4.1 Forward Phase

A step in the forward phase of GIES can be formalized as follows: for an $\mathcal{I}$-essential graph G_i, find the next one $G_{i+1} := \mathcal{E}_\mathcal{I}(D_{i+1})$, where

$$D_{i+1} := \operatorname*{arg\,max}_{D' \in \mathbf{D}^+(G_i)} S(D'; \mathcal{T}, \mathbf{X}), \text{ and}$$
$$\mathbf{D}^+(G_i) := \{D' \text{ a DAG} \mid \exists \text{ an arrow } u \longrightarrow v \in D' : D' - (u,v) \in \mathbf{D}(G_i)\} .$$

If no candidate DAG $D' \in \mathbf{D}^+(G_i)$ scores higher than G_i, abort the forward phase.

We denote the set of candidate *$\mathcal{I}$-essential graphs* by $\boldsymbol{\mathcal{E}}^+_\mathcal{I}(G_i) := \{\mathcal{E}_\mathcal{I}(D') \mid D' \in \mathbf{D}^+(G_i)\}$. In the next proposition, we show that each graph $G' \in \boldsymbol{\mathcal{E}}^+_\mathcal{I}(G_i)$ can be characterized by a triple (u,v,C), where $u \longrightarrow v$ is the arrow that has to be added to a representative D of G_i in order to get a representative D' of G', and C specifies the edge orientations of D within the chain component of v in G.

```
Input  : (T, X): interventional data for family of targets I
Output: I-essential graph
G ← ([p], ∅);
repeat
    DoContinue ← FALSE;
    repeat
        G_old ← G;
        G ← ForwardStep(G; T, X) ;                          // See Algorithm 3
    until G_old = G;
    repeat
        G_old ← G;
        G ← BackwardStep(G; T, X) ;                         // See Algorithm 4
        if G_old ≠ G then DoContinue ← TRUE;
    until G_old = G;
    repeat
        G_old ← G;
        G ← TurningStep(G; T, X) ;                          // See Algorithm 5
        if G_old ≠ G then DoContinue ← TRUE;
    until G_old = G;
until ¬DoContinue;
```

Algorithm 2: GIES$(\mathcal{T}, \mathbf{X})$. Greedy Interventional Equivalence Search. The steps of the different phases of the algorithms are described in Algorithms 3–5.

Proposition 25 *Let G be an $\mathcal{I}$-essential graph, let u and v be two non-adjacent vertices of G, and let $C \subset \mathrm{ne}_G(v)$. Then there is a DAG $D \in \mathbf{D}(G)$ with $\{a \in \mathrm{ne}_G(v) \mid a \longrightarrow v \in D\} = C$ such that $D' := D + (u,v) \in \mathbf{D}^+(G)$ if and only if*

(i) C is a clique in $G[T_G(v)]$;
(ii) $N := \mathrm{ne}_G(v) \cap \mathrm{ad}_G(u) \subset C$;
(iii) and every path from v to u in G has a vertex in C.

For given G, u, v and C determine D' uniquely up to $\mathcal{I}$-equivalence.

Note that points (i) and (ii) imply in particular that N is a clique in $G[T_G(v)]$. Proposition 25 has already been proven for the case of observational data (Chickering, 2002b, Theorem 15); it is not obvious, however, to see that this characterization of a forward step is also valid for interventional essential graphs, so we give a new proof in Appendix B.3 using the results developed in Sections 2 and 3.

The DAGs D and D' in Proposition 25 only differ in the edge (u,v); v is the only vertex whose parents are different in D and D'. Since the score function S is assumed to be decomposable, the score difference between D and D' can be expressed by the local score change at vertex v, as stated in the following corollary.

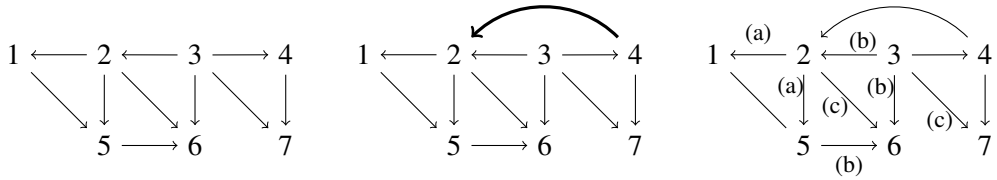
Corollary 26 *Let G, u, v, C, D and D' be as in Proposition 25. The score difference $\Delta S := S(D';\mathcal{T},\mathbf{X}) - S(D;\mathcal{T},\mathbf{X})$ can be calculated as follows:*

$$\Delta S = s(v, \mathrm{pa}_G(v) \cup C \cup \{u\}; \mathcal{T}, \mathbf{X}) - s(v, \mathrm{pa}_G(v) \cup C; \mathcal{T}, \mathbf{X}).$$

In the observational case, this corollary corresponds to Corollary 16 of Chickering (2002b).

Input : $G = ([p], E)$: $\mathcal{I}$-essential graph; $(\mathcal{T}, \mathbf{X})$: interventional data for $\mathcal{I}$
Output: $G' \in \boldsymbol{\mathcal{E}}^+_{\mathcal{I}}(G)$, or G
$\Delta S_{\max} \leftarrow 0$;
foreach $v \in [p]$ **do**
  foreach $u \in [p] \setminus \mathrm{ad}_G(v)$ **do**
    $N \leftarrow \mathrm{ne}_G(v) \cap \mathrm{ad}_G(u)$;
    foreach *clique* $C \subset \mathrm{ne}_G(v)$ *with* $N \subset C$ **do** `// Proposition 25(i) and (ii)`
      if $\nexists$ *path from v to u in* $G[[p] \setminus C]$ **then** `// Proposition 25(iii)`
        $\Delta S \leftarrow s(v, \mathrm{pa}_G(v) \cup C \cup \{u\}; \mathcal{T}, \mathbf{X}) - s(v, \mathrm{pa}_G(v) \cup C; \mathcal{T}, \mathbf{X})$;
        if $\Delta S > \Delta S_{\max}$ **then**
          $\Delta S_{\max} \leftarrow \Delta S$;
          $(u_{\max}, v_{\max}, C_{\max}) \leftarrow (u, v, C)$;
if $\Delta S_{\max} > 0$ **then**
  $\sigma \leftarrow \mathrm{LexBFS}((C_{\max}, v_{\max}, \ldots), E[T_G(v_{\max})])$;
  Orient edges of $G[T_G(v_{\max})]$ according to σ;
  Insert edge $(u_{\max}, v_{\max})$ into G;
  return ReplaceUnprotected$(\mathcal{I}, G)$; `// See Algorithm 1`
else return G;

Algorithm 3: ForwardStep$(G; \mathcal{T}, \mathbf{X})$. One step of the forward phase of GIES.



(a) D (b) D' (c) $\mathcal{E}_{\mathcal{I}}(D')$

Figure 4: DAGs D, D' and $\mathcal{E}_{\mathcal{I}}(D')$ illustrating a possible forward step of GIES for the family of targets $\mathcal{I} = \{\emptyset, \{4\}\}$, applied to the $\mathcal{I}$-essential graph G of Figure 3 for the parameters $(u, v, C) = (4, 2, \{3\})$ (notation according to Proposition 25). In parentheses in Figure (c): arrow configurations according to Definition 14; arrows incident to 4 are strongly $\mathcal{I}$-protected by the intervention target $\{4\}$.

The most straightforward way to construct an $\mathcal{I}$-essential graph $G' \in \boldsymbol{\mathcal{E}}^+_{\mathcal{I}}(G)$ characterized by the triple (u, v, C) as defined in Proposition 25 would be to create a representative $D \in \mathbf{D}(G)$ by orienting the edges of $T_G(v)$ as indicated by the set C, add the arrow $u \longrightarrow v$ to get D', and finally construct $\mathcal{E}_{\mathcal{I}}(D')$ with Algorithm 1. The next lemma suggests a novel shortcut to this procedure: it is sufficient to orient the edges of the chain component $T_G(v)$ *only* to get a partial $\mathcal{I}$-essential graph of D' after adding the arrow $u \longrightarrow v$.

Lemma 27 *Let G, u, v, C, D and D' be as in Proposition 25. Let H be the graph that we get by orienting all edges of $T_G(v)$ as in D (leaving other chain components unchanged) and inserting the arrow (u, v). Then H is a partial $\mathcal{I}$-essential graph of D'.*

Algorithm 3 shows our implementation of the forward phase of GIES, summarizing the results of Proposition 25, Corollary 26 and Lemma 27. Figure 4 illustrates one forward step, applied to the $\mathcal{I}$-essential graph G (for $\mathcal{I} = \{\emptyset, \{4\}\}$) of Figure 3 and characterized by the triple $(u, v, C) = (4, 2, \{3\})$. Note that this triple is indeed valid in the sense of Proposition 25: $\{3\}$ is clearly a clique (point (i)), $\mathrm{ne}_G(2) \cap \mathrm{ad}_G(4) = \{3\}$ (point (ii)), and there is no path from 2 to 4 in $G[[p] \setminus C]$ (point (iii)).

4.2 Backward Phase

In analogy to the forward phase, one step of the backward phase can be formalized as follows: for an $\mathcal{I}$-essential graph G_i, find its successor $G_{i+1} := \mathcal{E}_{\mathcal{I}}(D_{i+1})$, where

$$D_{i+1} := \operatorname*{arg\,max}_{D' \in \mathbf{D}^-(G_i)} S(D'; \mathbf{X}), \text{ and}$$
$$\mathbf{D}^-(G_i) := \{D' \text{ a DAG} \mid \exists\, D \in \mathbf{D}(G_i), u \longrightarrow v \in D : D' = D - (u, v)\}\,.$$

If no candidate DAG $D' \in \mathbf{D}^+(G_i)$ scores higher than G_i, the backward phase is aborted.

Whenever we have some representative $D \in \mathbf{D}(G)$ of an $\mathcal{I}$-essential graph G, we get a DAG in $\mathbf{D}^-(G)$ by removing any arrow of D. This is in contrast to the forward phase where we do not necessarily get a DAG in $\mathbf{D}^+(G)$ by adding an arbitrary arrow to D. By adding arrows, new directed cycles could be created, something which is not possible by removing arrows. This is the reason why the backward phase is generally simpler to implement than the forward phase.

In Proposition 28 (corresponding to Theorem 17 of Chickering (2002b) for the observational case), we show that we can, similarly to the forward phase, characterize an $\mathcal{I}$-essential graph of $\boldsymbol{\mathcal{E}}_{\mathcal{I}}^{-}(G) := \{\mathcal{E}_{\mathcal{I}}(D') \mid D' \in \mathbf{D}^{-}(G)\}$ by a triple (u,v,C), where C is a clique in $\mathrm{ne}_G(v)$. As in the forward phase, we see that the score difference of D and D' is determined by the local score change at the vertex v (Corollary 29), and that lines in chain components other than $T_G(v)$ remain lines in $G' = \mathcal{E}_{\mathcal{I}}(D')$ (Lemma 30). Algorithm 4 summarizes the results of the propositions in this section.

Proposition 28 *Let $G = ([p], E)$ be an $\mathcal{I}$-essential graph with $(u,v) \in E$ (that is, $u \mathrel{-\!\!-} v \in G$ or $u \longrightarrow v \in G$), and let $C \subset \mathrm{ne}_G(v)$. There is a DAG $D \in \mathbf{D}(G)$ with $u \longrightarrow v \in D$ and $\{a \in \mathrm{ne}_G(v) \setminus \{u\} \mid a \longrightarrow v \in D\} = C$ such that $D' := D - (u,v) \in \mathbf{D}^{-}(G)$ if and only if*

(i) C is a clique in $G[T_G(v)]$;
(ii) $C \subset N := \mathrm{ne}_G(v) \cap \mathrm{ad}_G(u)$.

Moreover, u, v and C determine D' uniquely up to $\mathcal{I}$-equivalence for a given G.

Corollary 29 *Let G, u, v, C, D and D' be as in Proposition 28. The score difference $\Delta S := S(D'; \mathcal{T}, \mathbf{X}) - S(D; \mathcal{T}, \mathbf{X})$ is:*

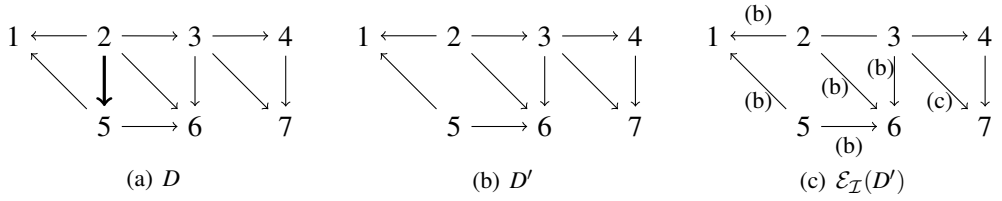
$$\Delta S = s(v, (\mathrm{pa}_G(v) \cup C) \setminus \{u\}; \mathcal{T}, \mathbf{X}) - s(v, \mathrm{pa}_G(v) \cup C \cup \{u\}; \mathcal{T}, \mathbf{X}).$$

In the observational case, this corresponds to Corollary 18 in Chickering (2002b). The analogue to Lemma 27 for a computational shortcut in the forward phase reads as follows:

Lemma 30 *Let G, u, v, C, D and D' be as in Proposition 28. Let H be the graph that we get by orienting all edges of $T_G(v)$ as in D and removing the arrow (u,v). Then H is a partial $\mathcal{I}$-essential graph of D'.*

Input : $G = ([p], E)$: $\mathcal{I}$-essential graph; $(\mathcal{T}, \mathbf{X})$: interventional data for $\mathcal{I}$
Output: $G' \in \boldsymbol{\mathcal{E}}_{\mathcal{I}}^{-}(G)$, or G
$\Delta S_{\max} \leftarrow 0$;
foreach $v \in [p]$ **do**
  foreach $u \in \mathrm{ne}_G(v) \cup \mathrm{pa}_G(v)$ **do**
    $N \leftarrow \mathrm{ne}_G(v) \cap \mathrm{ad}_G(u)$;
    foreach *clique* $C \subset N$ **do**
      $\Delta S \leftarrow s(v, (\mathrm{pa}_G(v) \cup C) \setminus \{u\}; \mathcal{T}, \mathbf{X}) - s(v, \mathrm{pa}_G(v) \cup C \cup \{u\}; \mathcal{T}, \mathbf{X})$;
      if $\Delta S > \Delta S_{\max}$ **then**
        $\Delta S_{\max} \leftarrow \Delta S$;
        $(u_{\max}, v_{\max}, C_{\max}) \leftarrow (u, v, C)$;
if $\Delta S_{\max} > 0$ **then**
  if $u_{\max} \in \mathrm{ne}_G(v_{\max})$ **then** $\sigma \leftarrow \mathrm{LexBFS}((C_{\max}, u_{\max}, v_{\max}, \ldots), E[T_G(v_{\max})])$;
  else $\sigma \leftarrow \mathrm{LexBFS}((C_{\max}, v_{\max}, \ldots), E[T_G(v_{\max})])$;
  Orient edges of $G[T_G(v_{\max})]$ according to σ;
  Remove edge $(u_{\max}, v_{\max})$ from G;
  return $\mathrm{ReplaceUnprotected}(\mathcal{I}, G)$; `// See Algorithm 1`
else return G;

Algorithm 4: $\mathrm{BackwardStep}(G; \mathcal{T}, \mathbf{X})$. One step of the backward phase of GIES.



(a) D (b) D' (c) $\mathcal{E}_{\mathcal{I}}(D')$

Figure 5: DAGs D, D' and $\mathcal{E}_{\mathcal{I}}(D')$ illustrating a possible backward step of GIES for the family of targets $\mathcal{I} = \{\emptyset, \{4\}\}$, applied to the $\mathcal{I}$-essential graph G of Figure 3 for the parameters $(u, v, C) = (2, 5, \emptyset)$ (notation according to Proposition 28). Figure (c), in parentheses: arrow configurations according to Definition 14.

A backward step of GIES is summarized in Algorithm 4 and illustrated in Figure 5. The triple $(u, v, C) = (2, 5, \emptyset)$ used there to characterize the backward step obviously fulfills the requirements of Proposition 28.

4.3 Turning Phase

Finally, we characterize a step of the turning phase of GIES, in which we want to find the successor $G_{i+1} := \mathcal{E}_{\mathcal{I}}(D_{i+1})$ for an $\mathcal{I}$-essential graph G_i by the rule

$$\begin{aligned} D_{i+1} :=& \operatorname*{arg\,max}_{D' \in \mathbf{D}^{\circlearrowleft}(G_i)} S(D'; \mathcal{T}, \mathbf{X}), \text{ where} \\ \mathbf{D}^{\circlearrowleft}(G_i) :=& \{D' \text{ a DAG } | D' \notin \mathbf{D}(G_i), \text{ and } \exists \text{ an arrow } u \longrightarrow v \in D' : \\ & D' - (u, v) + (v, u) \in \mathbf{D}(G_i)\} \,. \end{aligned}$$

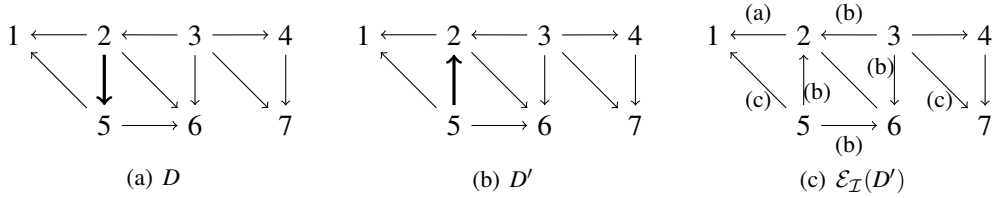
When the score cannot be augmented anymore, the turning phase is aborted. The additional condition "$D' \notin \mathbf{D}(G_i)$" is not necessary in the definitions of $\mathbf{D}^{+}(G_i)$ and $\mathbf{D}^{-}(G_i)$; when adding or removing an arrow from a DAG, the skeleton changes, hence the new DAG is certainly not $\mathcal{I}$-equivalent to the previous one. However, when *turning* an arrow, the skeleton remains the same, and the danger of staying in the same equivalence class exists.

Again, we are looking for an efficient method to find a representative D' for each $G' \in \boldsymbol{\mathcal{E}}^{\circlearrowleft}_{\mathcal{I}}(G_i) := \{\mathcal{E}_{\mathcal{I}}(D') \mid D' \in \mathbf{D}^{\circlearrowleft}(G_i)\}$. It makes sense to distinguish whether the arrow that should be turned in a representative $D \in \mathbf{D}(G_i)$ is $\mathcal{I}$-essential or not. We start with the case where we want to turn an arrow which is *not* $\mathcal{I}$-essential.

Proposition 31 *Let G be an $\mathcal{I}$-essential graph with $u - v \in G$, and let $C \subset \mathrm{ne}_G(v) \setminus \{u\}$. Define $N := \mathrm{ne}_G(v) \cap \mathrm{ad}_G(u)$. Then there is a DAG $D \in \mathbf{D}(G)$ with $u \longleftarrow v \in D$ and $\{a \in \mathrm{ne}_G(v) \mid a \longrightarrow v \in D\} = C$ such that $D' := D - (v, u) + (u, v) \in \mathbf{D}^{\circlearrowleft}(G)$ if and only if*

(i) C is a clique in $G[T_G(v)]$;
(ii) $C \setminus N \neq \emptyset$;
(iii) $C \cap N$ separates $C \setminus N$ and $N \setminus C$ in $G[\mathrm{ne}_G(v)]$.

For a given G, u, v and C determine D' up to $\mathcal{I}$-equivalence.



(a) D (b) D' (c) $\mathcal{E}_{\mathcal{I}}(D')$

Figure 6: DAGs D, D' and $\mathcal{E}_{\mathcal{I}}(D')$ illustrating a possible turning step of GIES applied to the $\mathcal{I}$-essential graph G ($\mathcal{I} = \{\emptyset, \{4\}\}$) of Figure 3 for the parameters $(u, v, C) = (5, 2, \{3\})$ (notation of Proposition 31). The arrow $2 \longrightarrow 5$ is not $\mathcal{I}$-essential in D. Figure (c): arrow configurations in parentheses, see Definition 14.

There are now *two* vertices that have different parents in the DAGs D and D', namely u and v; thus the calculation of the score difference between D and D' involves two local scores instead of one.

Corollary 32 *Let G, u, v, C, D and D' be as in Proposition 31. Then the score difference $\Delta S := S(D'; \mathcal{T}, \mathbf{X}) - S(D; \mathcal{T}, \mathbf{X})$ can be calculated as follows:*

$$\begin{aligned}\Delta S = {} & s(v, \mathrm{pa}_G(v) \cup C \cup \{u\}; \mathcal{T}, \mathbf{X}) + s(u, \mathrm{pa}_G(u) \cup (C \cap N); \mathcal{T}, \mathbf{X}) \\ & - s(v, \mathrm{pa}_G(v) \cup C; \mathcal{T}, \mathbf{X}) - s(u, \mathrm{pa}_G(u) \cup (C \cap N) \cup \{v\}; \mathcal{T}, \mathbf{X}).\end{aligned}$$

Lemma 33 *Let G, u, v, C, D and D' be as in Proposition 31. Let H be the graph that we get by orienting all edges of $T_G(v)$ as in D and turning the arrow (v, u). Then H is a partial $\mathcal{I}$-essential graph of D'.*

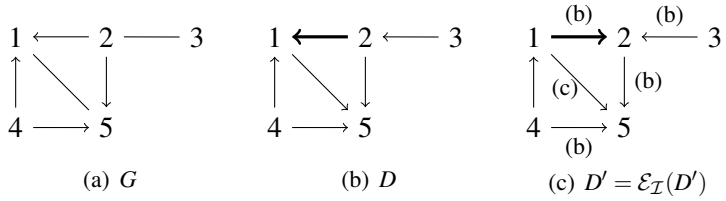
A possible turning step is illustrated in Figure 6, where a non-$\mathcal{I}$-essential arrow (for $\mathcal{I} = \{\emptyset, \{4\}\}$) of a representative of the graph G of Figure 3 is turned. The step is characterized by the triple $(u, v, C) = (5, 2, \{3\})$ which satisfies the conditions of Proposition 31: $\{3\}$ is obviously a clique (point (i)), $C \setminus N = C$ since $N = \{1\}$ (point (ii)), and $C \setminus N = \{3\}$ and $N \setminus C = \{1\}$ are separated in $G[\mathrm{ne}_G(2)]$ (point (iii)). In contrast, the triple $(u, v, C) = (5, 2, \{1\})$ fulfills points (i) and (iii) of Proposition 31, but not point (ii). There is a DAG $D \in \mathbf{D}(G)$ with $\{a \in \mathrm{ne}_G(2) \mid a \longrightarrow 2 \in D\} = \{1\}$, and turning the arrow $2 \longrightarrow 5$ in D yields another DAG D' (that is, does not create a new cycle). This new DAG D', however, is $\mathcal{I}$-equivalent to D, and hence not a member of $\mathbf{D}^{\circlearrowleft}(G)$ (see the discussion above).

We now proceed to the case where an $\mathcal{I}$-essential arrow of a representative of G is turned; here there is no danger to remain in the same Markov equivalence class. The characterization of this case is similar to the forward phase.

Proposition 34 *Let G be an $\mathcal{I}$-essential graph with $u \longleftarrow v \in G$, and let $C \subset \mathrm{ne}_G(v)$. Then there is a DAG $D \in \mathbf{D}(G)$ with $\{a \in \mathrm{ne}_G(v) \mid a \longrightarrow v \in D\} = C$ such that $D' := D - (v, u) + (u, v) \in \mathbf{D}^{\circlearrowleft}(G)$ if and only if*

(i) C is a clique;
(ii) $N := \mathrm{ne}_G(v) \cap \mathrm{ad}_G(u) \subset C$;
(iii) every path from v to u in G except (v, u) has a vertex in $C \cup \mathrm{ne}_G(u)$.

Moreover, u, v and C determine D' up to $\mathcal{I}$-equivalence.



(a) G (b) D (c) $D' = \mathcal{E}_{\mathcal{I}}(D')$

Figure 7: Graphs G, D, D' and $\mathcal{E}_{\mathcal{I}}(D')$ illustrating a possible turning step of GIES for the family of targets $\mathcal{I} = \{\emptyset, \{4\}\}$ and the parameters $(u, v, C) = (1, 2, \{3\})$ (notation of Proposition 34). The arrow $2 \longrightarrow 1$ is $\mathcal{I}$-essential in D. Figure (c): arrow configurations in parentheses, see Definition 14.

Chickering (2002a) has already proposed a turning step for essential arrows in the observational case; however, he did not provide necessary and sufficient conditions specifying all possible turning steps as Proposition 34 does.

Lemma 35 *Let G, u, v, C, D and D' be as in Proposition 34, and let H be the graph that we get by orienting all edges of $T_G(v)$ and $T_G(u)$ as in D and by turning the edge (v,u). Then H is a partial $\mathcal{I}$-essential graph of D'.*

To construct a $G' \in \boldsymbol{\mathcal{E}}^{\circlearrowleft}_{\mathcal{I}}(G)$ out of G, we must possibly orient *two* chain components of G instead of one (Lemma 35). In the example of Figure 7, we see that it is indeed not sufficient to orient the edges of $T_G(v)$ alone in order to get a partial $\mathcal{I}$-essential graph of G'. The arrow $1 \longrightarrow 5$ is not $\mathcal{I}$-essential in D, hence $5 \in T_G(1)$. However, the same arrow is $\mathcal{I}$-essential in D' and hence also present in $\mathcal{E}_{\mathcal{I}}(D')$.

Despite the fact that we need to orient the edges of $T_G(v)$ and $T_G(u)$ to get a partial $\mathcal{I}$-essential graph of D', $\mathcal{E}_{\mathcal{I}}(D')$ is nevertheless determined by the orientation of edges adjacent to v (determined by the clique C) alone. This comes from the fact that in D, defined as in Proposition 34, all arrows of $D[T_G(u)]$ must point away from u.

Corollary 36 *Let G, u, v, C, D and D' be as in Proposition 34. Then the score difference $\Delta S := S(D'; \mathcal{T}, \mathbf{X}) - S(D; \mathcal{T}, \mathbf{X})$ can be calculated as follows:*

$$\begin{aligned} \Delta S \;=\; & s(v, \mathrm{pa}_G(v) \cup C \cup \{u\}; \mathcal{T}, \mathbf{X}) + s(u, \mathrm{pa}_G(u) \setminus \{v\}; \mathcal{T}, \mathbf{X}) \\ & - s(v, \mathrm{pa}_G(v) \cup C; \mathcal{T}, \mathbf{X}) - s(u, \mathrm{pa}_G(u); \mathcal{T}, \mathbf{X}). \end{aligned}$$

The entire turning step, for essential and non-essential arrows, is shown in Algorithm 5.

4.4 Discussion

Every step in the forward, backward and turning phase of GIES is characterized by a triple (u, v, C), where u and v are different vertices and C is a clique in the neighborhood of v. To identify the highest scoring movement from one $\mathcal{I}$-essential graph G to a potential successor in $\boldsymbol{\mathcal{E}}^{+}_{\mathcal{I}}(G)$, $\boldsymbol{\mathcal{E}}^{-}_{\mathcal{I}}(G)$ or $\boldsymbol{\mathcal{E}}^{\circlearrowleft}_{\mathcal{I}}(G)$, respectively, one potentially has to examine all cliques in the neighborhood $\mathrm{ne}_G(v)$ of all vertices $v \in [p]$. The time complexity of any (forward, backward or turning) step applied to an $\mathcal{I}$-essential graph G hence highly depends on the size of the largest clique in the chain components

of G. By restricting GIES to $\mathcal{I}$-essential graphs with a bounded vertex degree, the time complexity of a step of GIES is polynomial in p; otherwise, it is in the worst case exponential. We believe, however, that GIES is in practice much more efficient than this worst-case complexity suggests. Some evidence for this claim is provided by the runtime analysis of our simulation study, see Section 5.2.

A heuristic approach to guarantee polynomial runtime of a greedy search has been proposed by Castelo and Kočka (2003) for the observational case. Their Hill Climber Monte Carlo (HCMC) algorithm operates in DAG space, but to account for Markov equivalence, the neighborhood of a number of randomly chosen DAGs equivalent to the current one is scanned in each greedy step.

Input : $G = ([p], E)$: $\mathcal{I}$-essential graph; $(\mathcal{T}, \mathbf{X})$: interventional data for $\mathcal{I}$
Output: $G' \in \boldsymbol{\mathcal{E}}^{\circlearrowleft}_{\mathcal{I}}$, or G
$\Delta S_{\max} \leftarrow 0$;
foreach $v \in [p]$ **do**
  foreach $u \in \mathrm{ne}_G(v)$ **do** `// Consider arrows that are not` $\mathcal{I}$`-essential for turning`
    $N \leftarrow \mathrm{ne}_G(u) \cap \mathrm{ad}_G(v)$;
    foreach *clique* $C \subset \mathrm{ne}_G(v) \setminus \{u\}$ **do** `// Proposition 31(i)`
      if $C \setminus N \neq \emptyset$ *and* $\{u, v\}$ *separates* C *and* $N \setminus C$ *in* $G[T_G(v)]$ **then**
        `// Proposition 31(ii) and (iii)`
        $\Delta S \leftarrow s(v, \mathrm{pa}_G(v) \cup C \cup \{u\}; \mathcal{T}, \mathbf{X}) + s(u, \mathrm{pa}_G(u) \cup (C \cap N); \mathcal{T}, \mathbf{X})$;
        $\Delta S \leftarrow \Delta S - s(v, \mathrm{pa}_G(v) \cup C; \mathcal{T}, \mathbf{X}) - s(u, \mathrm{pa}_G(u) \cup (C \cap N) \cup \{v\}; \mathcal{T}, \mathbf{X})$;
        if $\Delta S > \Delta S_{\max}$ **then**
          $\Delta S_{\max} \leftarrow \Delta S$;
          $(u_{\max}, v_{\max}, C_{\max}) \leftarrow (u, v, C)$;
  foreach $u \in \mathrm{ch}_G(v)$ **do** `// Consider` $\mathcal{I}$`-essential arrows for turning`
    $N \leftarrow \mathrm{ne}_G(v) \cap \mathrm{ad}_G(u)$;
    foreach *clique* $C \subset \mathrm{ne}_G(v)$ *with* $N \subset C$ **do** `// Proposition 34(i) and (ii)`
      if $\nexists$ *path from* v *to* u *in* $G[[p] \setminus (C \cup \mathrm{ne}_G(u))] - (v, u)$ **then** `// Proposition 34(iii)`
        $\Delta S \leftarrow s(v, \mathrm{pa}_G(v) \cup C \cup \{u\}; \mathcal{T}, \mathbf{X}) + s(u, \mathrm{pa}_G(u) \setminus \{v\}; \mathcal{T}, \mathbf{X})$;
        $\Delta S \leftarrow \Delta S - s(v, \mathrm{pa}_G(v) \cup C; \mathcal{T}, \mathbf{X}) - s(u, \mathrm{pa}_G(u); \mathcal{T}, \mathbf{X})$;
        if $\Delta S > \Delta S_{\max}$ **then**
          $\Delta S_{\max} \leftarrow \Delta S$;
          $(u_{\max}, v_{\max}, C_{\max}) \leftarrow (u, v, C)$;
if $\Delta S_{\max} > 0$ **then**
  if $v_{\max} \longrightarrow u_{\max} \in G$ **then**
    $\sigma_u := \mathrm{LEXBFS}((u_{\max}, \ldots), E[T_G(u_{\max})])$;
    Orient edges of $G[T_G(u_{\max})]$ according to σ;
    $\sigma_v := \mathrm{LEXBFS}((C_{\max}, v_{\max}, \ldots), E[T_G(v_{\max})])$;
  else $\sigma_v := \mathrm{LEXBFS}((C_{\max}, v_{\max}, u_{\max}, \ldots), E[T_G(v_{\max})])$;
  Orient edges of $G[T_G(v_{\max})]$ according to σ_v;
  Turn edge $(v_{\max}, u_{\max})$ in G;
  return $\mathrm{REPLACEUNPROTECTED}(\mathcal{I}, G)$; `// See Algorithm 1`
else return G;

Algorithm 5: $\mathrm{TURNINGSTEP}(G; \mathcal{T}, \mathbf{X})$. One step of the turning phase of GIES.

The equivalence class of the current DAG is explored by randomly turning "covered arrows", that is, arrows whose reversal does not change the Markov property. In our (interventional) notation, an arrow is covered if and only if it is *not* strongly $\mathcal{I}$-protected (Definition 14). By limiting the number of covered arrow reversals, a polynomial runtime is guaranteed at the cost of potentially lowering the probability of investigating a particular successor in $\boldsymbol{\mathcal{E}}^{+}_{\mathcal{I}}(G)$, $\boldsymbol{\mathcal{E}}^{-}_{\mathcal{I}}(G)$ or $\boldsymbol{\mathcal{E}}^{\circlearrowleft}_{\mathcal{I}}(G)$, respectively. HCMC hence enables a fine tuning of the trade-off between exploration of the search space and runtime, or between greediness and randomness.

The order of executing the backward and the turning phase seems somewhat arbitrary. In the analysis of the steps performed by GIES in our simulation study (Section 5.2), we saw that the turning phase can generally only augment the score when very few backward steps were executed before. For this reason, we believe that changing the order of the backward and the turning phase would have little effect on the overall performance of GIES.

As already discussed by Chickering (2002b) for the observational case, caching techniques can markedly speed up GES; the same holds for GIES. The basic idea is the following: in a forward step, the algorithm evaluates a lot of triples (u,v,C) to choose the best one, $(u_{\max},v_{\max},C_{\max})$ (lines 1 to 9 in Algorithm 3). After performing the forward move corresponding to $(u_{\max},v_{\max},C_{\max})$, many of the triples evaluated in the step before are still valid candidates for next step in the sense of Proposition 25 and lead to the same score difference as before (see Corollary 26). Caching those values avoids unnecessary reevaluation of possible forward steps. The same holds for the backward and the turning phase; since the forward step is most frequently executed, a caching strategy in this phase yields the highest speed-up though.

We emphasize that the characterization of "neighboring" $\mathcal{I}$-essential graphs in $\boldsymbol{\mathcal{E}}^{+}_{\mathcal{I}}(G)$, $\boldsymbol{\mathcal{E}}^{-}_{\mathcal{I}}(G)$ or $\boldsymbol{\mathcal{E}}^{\circlearrowleft}_{\mathcal{I}}(G)$, respectively, by triples (u,v,C) is of more general interest for structure learning algorithms, for example for the design of sampling steps of an MCMC algorithm. Also the beforementioned HCMC algorithm could be extended to interventional data by generalizing the notion of "covered arcs" using Definition 14.

The prime example of a score equivalent and decomposable score function is the Bayesian information criterion (BIC) (Schwarz, 1978) which we used in our simulations (Section 5). It penalizes the complexity of causal models by their number of free parameters (ℓ_0 penalization); this number is the sum of free parameters of the conditional densities in the Markov factorization (Definition 1), which explains the decomposability of the score. Using different penalties, for example, ℓ_2 penalization, can lead to a non-decomposable score function. GIES can also be adapted to such score functions; the calculation of score differences becomes computationally more expensive in this case since it cannot be done in a local fashion as in Corollaries 26, 29, 32 and 36.

GIES only relies on the notion of interventional Markov equivalence, and on a score function that can be evaluated for a given class of causal models. As we mentioned in Section 2.1, we believe that interventional Markov equivalence classes remain unchanged for models that do not have a strictly positive density. For this reason it should be safe to also apply GIES to such a model class.

5. Experimental Evaluation

We evaluated the GIES algorithm on simulated interventional data (Section 5.2) and on *in silico* gene expression data sets taken from the DREAM4 challenge (Marbach et al., 2010) (Section 5.3).

In both cases, we restricted our considerations to *Gaussian* causal models as summarized in Section 5.1.

5.1 Gaussian Causal Models

Consider a causal model (D, f) with a Gaussian density of the form $\mathcal{N}(0, \Sigma)$. The observational Markov property of such a model translates to a set of *linear* structural equations

$$X_i = \sum_{j=1}^{p} \beta_{ij} X_j + \varepsilon_i,\ \varepsilon_i \overset{\text{indep.}}{\sim} \mathcal{N}(0, \sigma_i^2), \quad 1 \leq i \leq p\ , \tag{5}$$

where $\beta_{ij} = 0$ if $j \notin \mathrm{pa}_D(i)$. When the DAG structure D is known, the covariance matrix Σ can be parameterized by the **weight matrix**

$$B := (\beta_{ij})_{i,j=1}^{p} \in \mathbf{B}(D) := \{A = (\alpha_{ij}) \in \mathbb{R}^{p \times p} \mid \alpha_{ij} = 0 \text{ if } j \notin \mathrm{pa}_D(i)\}$$

that assigns a weight β_{ij} to each arrow $j \longrightarrow i \in D$, and the vector of error covariances $\sigma^2 := (\sigma_1^2, \ldots, \sigma_p^2)$:

$$\Sigma = \mathrm{Cov}(X) = (\mathbb{1} - B)^{-1} \,\mathrm{diag}(\sigma^2)(\mathbb{1} - B)^{-\mathrm{T}}\ .$$

This is a consequence of Equation (5).

We always assume Gaussian intervention variables U_I (see Section 2.1). In this case, not only the observational density f is Gaussian, but also the interventional densities $f(x \mid \mathrm{do}_D(X_I = U_I))$. An interventional data set $(\mathcal{T}, \mathbf{X})$ as defined in Equation (2) then consists of n independent, but not identically distributed Gaussian samples.

We use the **Bayesian information criterion** (BIC) as score function for GIES:

$$S(D; \mathcal{T}, \mathbf{X}) := \sup\{\ell_D(B, \sigma^2; \mathcal{T}, \mathbf{X}) \mid B \in \mathbf{B}(D), \sigma^2 \in \mathbb{R}_{>0}^p\} - \frac{k_D}{2} \log(n)\ ,$$

where ℓ_D denotes the log-likelihood of the density in Equation (3):

$$\begin{aligned}
\ell_D(B, \sigma^2; \mathcal{T}, \mathbf{X}) &:= \sum_{i=1}^{n} \log f\big(X^{(i)} \mid \mathrm{do}_D(X^{(i)}_{T^{(i)}} = U_{T^{(i)}})\big) \\
&= \sum_{i=1}^{n} \Big[\sum_{j \notin T^{(i)}} \log f(X_j^{(i)} \mid X^{(i)}_{\mathrm{pa}_D(j)}) + \sum_{j \in T^{(i)}} \log \tilde{f}(X_j^{(i)}) \Big] \\
&= -\frac{1}{2} \sum_{i=1}^{n} \sum_{j \notin T^{(i)}} \Big[\log \sigma_j^2 + \frac{1}{\sigma_j^2} \Big(X_j^{(i)} - B_{j\bullet} X^{(i)} \Big)^2 \Big] + C \\
&= -\frac{1}{2} \sum_{j=1}^{p} \Big[|\{i \mid j \notin T^{(i)}\}| \log \sigma_j^2 + \frac{1}{\sigma_j^2} \sum_{i : j \notin T^{(i)}} \Big(X_j^{(i)} - B_{j\bullet} X^{(i)} \Big)^2 \Big] + C\ ,
\end{aligned} \tag{6}$$

where the constant C is independent of the parameters (B, σ^2) of the model. Since Gaussian causal models with structure D are parameterized by $B \in \mathbf{B}(D)$ and $\sigma^2 \in \mathbb{R}_{>0}^p$, we have $k_D = p + |E|$ free parameters, where E denotes the edge set of D. It can be seen in Equation (6) that the **maximum likelihood estimator** (MLE) $(\hat{B}, \hat{\sigma}^2)$, the maximizer of ℓ_D, minimizes the residual sum of squares for the different structural equations; for more details we refer to Hauser and Bühlmann (2012).

The DAG $\hat{D}$ maximizing the BIC yields a *consistent* estimator for the true causal structure D in the sense that $P[\hat{D} \sim_{\mathcal{I}} D] \to 1$ in the limit $n \to \infty$ as long as the true density f is **faithful** with respect to D, that is, *every* conditional independence relation of f is encoded in the Markov property of D (Hauser and Bühlmann, 2012). Note that the BIC score is even defined in the high-dimensional setting $p > n$; however, we only consider low-dimensional settings here.

5.2 Simulations

We simulated interventional data from 4000 randomly generated Gaussian causal models as described in Section 5.2.1. In Sections 5.2.2 and 5.2.3, we present our methods for evaluating GIES; the results are discussed in Section 5.2.4. As a rough summary, GIES markedly beat the conceptually simpler greedy search over the space of DAGs as well as the original GES of Chickering (2002b) ignoring the interventional nature of the simulated data sets. Its learning performance could keep up with a provably consistent exponential time dynamic programming algorithm at much lower computational cost.

5.2.1 GENERATION OF GAUSSIAN CAUSAL MODELS

For some number p of vertices, we randomly generated Gaussian causal models parameterized by a structure D, a weight matrix $B \in \mathbf{B}(D)$ and a vector of error covariances $\sigma^2 \in \mathbb{R}^p_{>0}$ by a procedure slightly adapted from Kalisch and Bühlmann (2007):

1. For a given **sparseness parameter** $s \in (0,1)$, draw a DAG D with topological ordering $(1,\dots,p)$ and binomially distributed vertex degrees with mean $s(p-1)$.

2. Shuffle the vertex indices of D to get a random topological ordering.

3. For each arrow $j \longrightarrow i \in D$, draw $\beta'_{ij} \sim \mathcal{U}([-1,-0.1] \cup [0.1,1])$ using independent realizations; for other pairs of (i,j), set $\beta'_{ij} = 0$ (see Equation (5)). This yields a weight matrix $B' = (\beta'_{ij})_{i,j=1}^p \in \mathbf{B}(D)$ with positive as well as negative entries which are bounded away from 0.

4. Draw error variances $\sigma'^2_i \overset{\text{i.i.d.}}{\sim} \mathcal{U}([0.5,1])$.

5. Calculate the corresponding covariance matrix $\Sigma' = (\mathbb{1} - B')^{-1} \operatorname{diag}(\sigma'^2)(\mathbb{1} - B')^{-\mathrm{T}}$.

6. Set $H := \operatorname{diag}((\Sigma'_{11})^{-1/2}, \dots, (\Sigma'_{pp})^{-1/2})$, and normalize the weights and error variances as follows:
$$B := HB'H^{-1}, \quad (\sigma_1^2, \dots, \sigma_p^2)^{\mathrm{T}} := H^2(\sigma'^2_1, \dots, \sigma'^2_p)^{\mathrm{T}} .$$
It can easily be seen that the corresponding covariance matrix fulfills
$$\Sigma = (\mathbb{1} - B)^{-1} \operatorname{diag}(\sigma^2)(\mathbb{1} - B)^{-\mathrm{T}} = H\Sigma' H ,$$
ensuring the desired normalization $\Sigma_{ii} = 1$ for all i.

Steps 1 and 3 are provided by the function `randomDAG()` of the R-package `pcalg` (Kalisch et al., 2012).

We considered families of targets of the form $\mathcal{I} = \{\emptyset, I_1, \dots, I_k\}$, where $I_1, \dots, I_k$ are k different, randomly chosen intervention targets of size m; the target size m had values between 1 and 4.

For a fixed sample size n, we produced approximately the same number of data samples for each target in the family $\mathcal{I}$ by using a level density $\mathcal{N}((2,\dots,2),(0.2)^2\mathbb{1}_m)$ in each case (see the model in Equation (1)). With this choice and the aforementioned normalization of Σ, the mean values of the intervention levels lay 2 standard deviations above the mean values of the observational marginal distributions. In total, we considered 4000 causal models and simulated 128 observational or interventional data sets from each of them by combining the following simulation parameters:

- $(p,s) \in \{(10,0.2),(20,0.1),(30,0.1),(40,0.1)\}$ with 1000 DAGs each.
- $k = 0, 0.2p, 0.4p, \dots, p$ for each value of p; the first setting is purely observational.
- $m \in \{1,2,4\}$.
- $n \in \{50,100,200,500,1000,2000,5000,10000\}$.

In addition, we generated causal models with $p \in \{50,100,200\}$ (100 DAGs each) and $p = 500$ (20 DAGs) with an expected vertex degree of 4 (which corresponds to a sparseness parameter of $s = 4/(p-1)$) and simulated 6 data sets for the parameters $k = 0.4$ and $n \in \{1000,2000,5000,10000,20000,50000\}$ from each of these models. We only used these additional data sets for the investigation of the runtime of GIES.

5.2.2 ALTERNATIVE STRUCTURE LEARNING ALGORITHMS

We compare GIES with three alternative greedy search algorithms. The first one is the original GES of Chickering (2002b) which regards the complete interventional data set as observational (that is, ignores the list $\mathcal{T}$ of an interventional data set $(\mathcal{T},\mathbf{X})$ as defined in Equation (2)). The second one, which we call GIES-NT (for "no turning"), is a variant of GIES that stops after the first forward and backward phase and lacks the turning phase. The third algorithm, called GDS for "greedy DAG search", is a simple greedy algorithm optimizing the same score function as GIES, but working on the space of DAGs instead of the space of $\mathcal{I}$-essential graphs; GDS simply adds, removes or turns arrows of DAGs in the forward, backward and turning phase, respectively. Furthermore, for $p \leq 20$, we compare with a dynamic programming (DP) approach proposed by Silander and Myllymäki (2006), an algorithm that finds a global optimum of any decomposable score function on the space of DAGs. Because of the exponential growth in time and memory requirements, we could not calculate DP estimates for models with $p \geq 30$ variables. For GDS and DP, we examine the $\mathcal{I}$-essential graph of the returned DAGs.

5.2.3 QUALITY MEASURES FOR ESTIMATED ESSENTIAL GRAPHS

The **structural Hamming distance** or SHD (Tsamardinos et al., 2006; we use the slightly adapted version of Kalisch and Bühlmann, 2007) is used to measure the distance between an estimated $\mathcal{I}$-essential graph $\hat{G}$ and a true $\mathcal{I}$-essential graph or DAG G. If A and $\hat{A}$ denote the adjacency matrices of G and $\hat{G}$, respectively, the SHD between G and $\hat{G}$ reads

$$\mathrm{SHD}(\hat{G},G) := \sum_{1\leq i<j\leq p} \left(1 - \mathbb{1}_{\{(A_{ij}=\hat{A}_{ij})\wedge(A_{ji}=\hat{A}_{ji})\}}\right).$$

The SHD between $\hat{G}$ and G is the sum of the numbers of false positives of the skeleton, false negatives of the skeleton, and wrongly oriented edges. Those quantities are defined as follows. Two

vertices which are adjacent in $\hat{G}$ but not in G count as one false positive, two vertices which are adjacent in G but not in $\hat{G}$ as one false negative. Two vertices which are adjacent in both G and $\hat{G}$, but connected with different edge types (that is, by a directed edge in one graph, by an undirected one in the other; or by directed edges with different orientations in both graphs) constitute a **wrongly oriented** edge.

5.2.4 RESULTS AND DISCUSSION

As we mentioned in Section 3.1, the undirected edges in the $\mathcal{I}$-essential graph $\mathcal{E}_{\mathcal{I}}(D)$ of some causal structure D are the edges with unidentifiable orientation. The number of undirected edges in $\mathcal{E}_{\mathcal{I}}(D)$ analyzed in the next paragraph is therefore a good measure for the identifiability of D. Later on, we study the performance of GIES and compare it to the other algorithms mentioned in Section 5.2.2.

*I*dentifiability under Interventions

In Figure 8, the number of non-$\mathcal{I}$-essential arrows is plotted as a function of the number k of non-empty intervention targets ($k = |\mathcal{I}| - 1$, see Section 5.2.1). With single-vertex interventions at 80% of the vertices, the majority of the DAGs used in the simulation are completely identifiable; with target size $m = 2$ or $m = 4$, this is already the case for $k = 0.6p$ or $k = 0.4p$, respectively. For the small target sizes used, the identifiability under k targets of size m is similar to the identifiability under $k \cdot m$ single-vertex targets.

A certain prudence is advisable when interpreting Figure 8 since the number of orientable edges also reflects the characteristics of the generated DAGs. Nevertheless, the plots show that the iden-

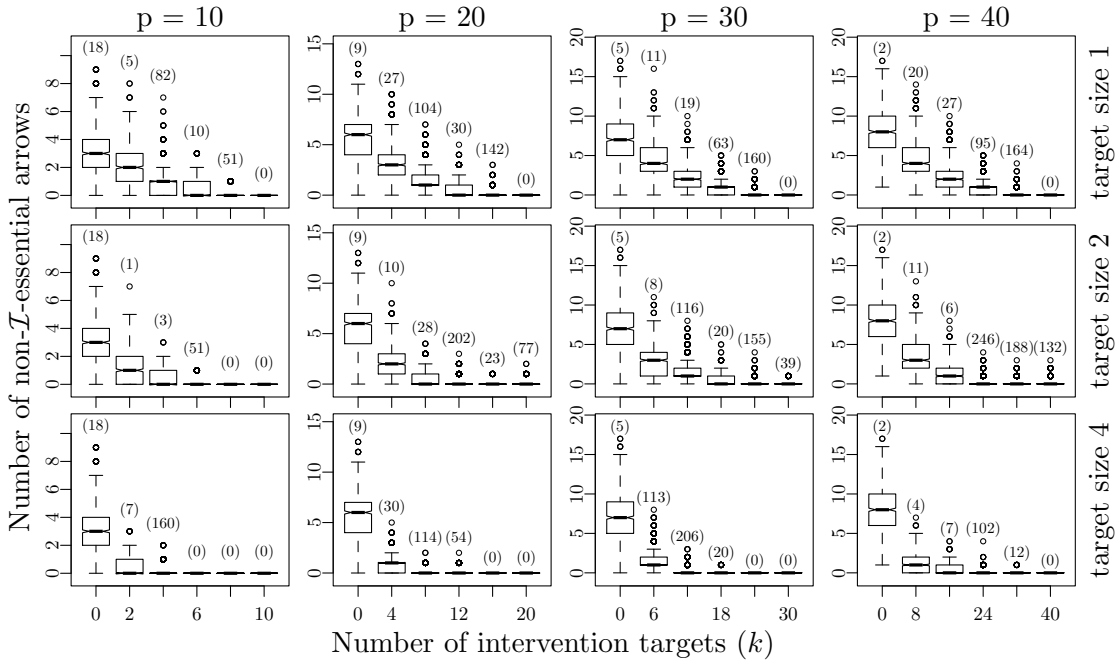


Figure 8: Number of non-$\mathcal{I}$-essential arrows as a function of the number k of intervention vertices. In parentheses: number of outliers in the corresponding boxplot.

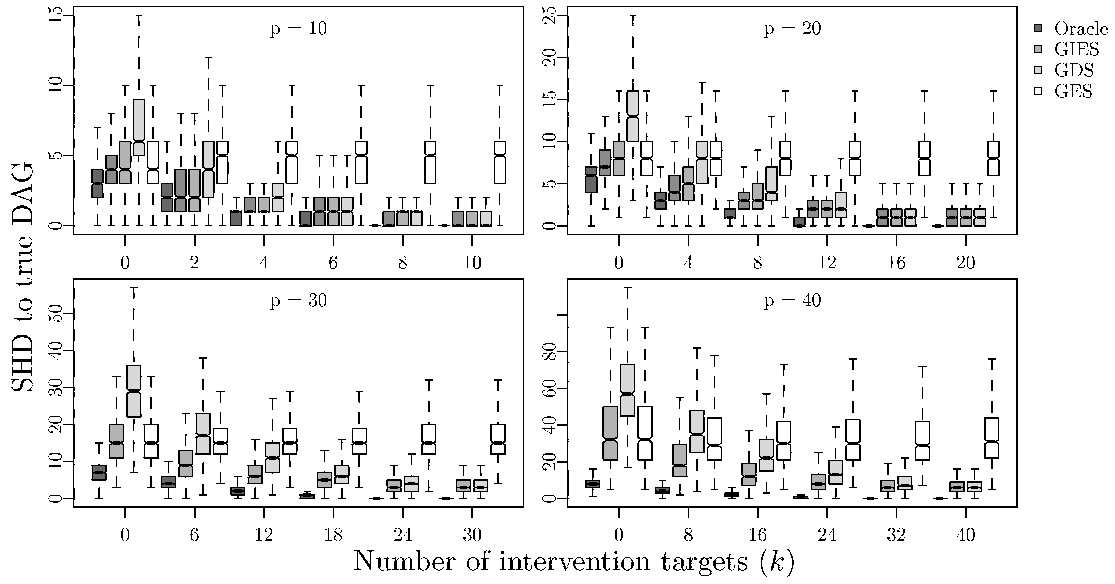


Figure 9: SHD between $\mathcal{I}$-essential graph $\hat{G}$ estimated from $n = 1000$ data points and true DAG D as a function of the number k of single-vertex intervention targets. "Oracle estimates" denote the respective true $\mathcal{I}$-essential graph $\mathcal{E}_{\mathcal{I}}(D)$, the best possible estimate under some family of targets $\mathcal{I}$ (see also Figure 8). DP estimates are missing in the two lower plots.

tifiability of causal models increases quickly even with few intervention targets. In regard of applications this is an encouraging finding since it illustrates that even a small number of intervention experiments can strongly increase the identifiability of causal structures.

*P*erformance of GIES

Figure 9 shows the structural Hamming distance between true DAG D and estimated $\mathcal{I}$-essential graph $\hat{G}$ for different algorithms as a function of the number k of intervention targets. Single-vertex interventions are considered; for larger targets, the overall picture is comparable (data not shown). In 10 out of 12 cases for $p \leq 20$, the median SHD values of GIES and DP estimates are equal; in the remaining cases, too, GIES yields estimates of comparable quality—at much lower computational costs.

In parallel with the identifiability, the estimates produced by the different algorithms improve for growing k. This illustrates that interventional data arising from different intervention targets carry *more* information about the underlying causal model than observational data of the same sample size.

For complete interventions, that is, $k = p$, every DAG is completely identifiable and hence its own $\mathcal{I}$-essential graph. Therefore, GDS and GIES are exactly the same algorithm in this case. With shrinking k, the performance of GDS compared to that of GIES gets worse. On the other hand, GES coincides with GIES in the observational case ($k = 0$). For growing k, the estimation performance of GES stays approximately constant; it can, as opposed to GIES, not make use of the additional information coming from interventions. To sum up, both the price of ignoring interventional Markov equivalence (GDS) and ignoring the interventional nature of the provided data sets (GES) are apparent in Figure 9.

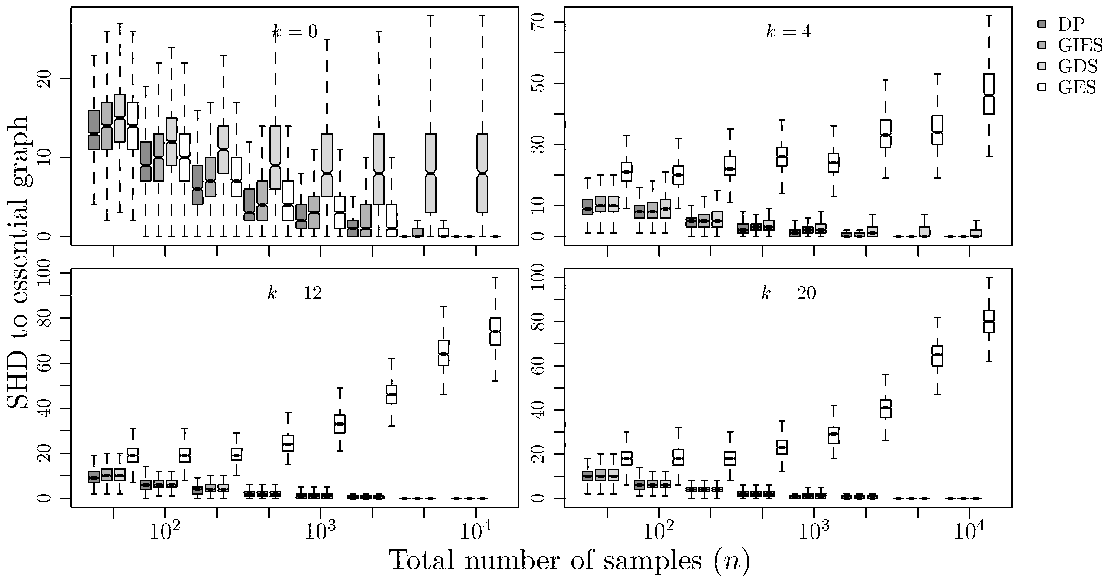


Figure 10: SHD between estimated and true $\mathcal{I}$-essential graph for different numbers k of intervention targets of size $m = 4$ for the DAGs with $p = 20$ vertices. The abscissa denotes the *total* sample size n. For example, a data set with $n = 1000$ and $k = 4$ consists of 200 observational samples and 200 interventional samples each arising from interventions at four different targets, see Section 5.2.1.

The performance of GIES as a function of the sample size n is plotted in Figure 10 for the DAGs with $p = 20$ vertices and intervention targets of size $m = 4$. The quality of the GIES estimates is comparable to that of the DP estimates. The behavior of the SHD values for growing n is a strong hint for the consistency of GIES in the limit $n \to \infty$ (note that the DP algorithm is consistent; Hauser and Bühlmann, 2012). In contrast, the plots for $k = 0$ and $k = 4$ again reveal the weak performance of GDS for small numbers of intervention vertices; the plots suggest that GDS, in contrast to GIES, does not yield a consistent estimator of the $\mathcal{I}$-essential graph due to being stuck in a bad local optimum.

The most striking result in Figure 10 is certainly the fact that the estimation performance of GES heavily decreases with growing n as long as the data is not observational ($k > 0$). This is not an artifact of GES, but a problem of model-misspecification: running DP for an *observational* model (that is, considering all data as observational as GES does) yields SHD values maximally 14% below that of GES (data not shown). For single-vertex interventions, the SHD values of the GES estimates stay approximately constant with growing n; for target size $m = 2$, its SHD values also increase, but not to the same extent as for $m = 4$.

In Figure 11, we compare the SHD between true and estimated $\mathcal{I}$-essential graphs with $p = 30$ vertices for estimates produced by different greedy algorithms; other vertex numbers give a similar picture. In most settings, GIES beats both GDS and GIES-NT. It combines both the advantage of GIES-NT, using the space of interventional Markov equivalence classes as search space, and GDS, the turning phase apparently reducing the risk of getting stuck in local maxima of the score function.

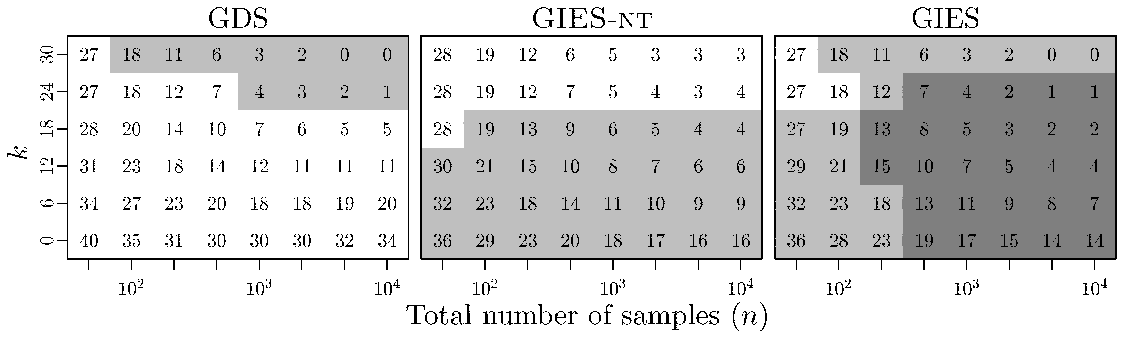


Figure 11: Mean SHD between estimated and true $\mathcal{I}$-essential graph for different greedy algorithms as a function of n and k; data for DAGs with $p = 30$ and single-vertex interventions. Shading: algorithm yielded significantly better estimates than one (■) or two (■) of its competitors, respectively (paired t-test on a significance level of $\alpha = 5\%$).

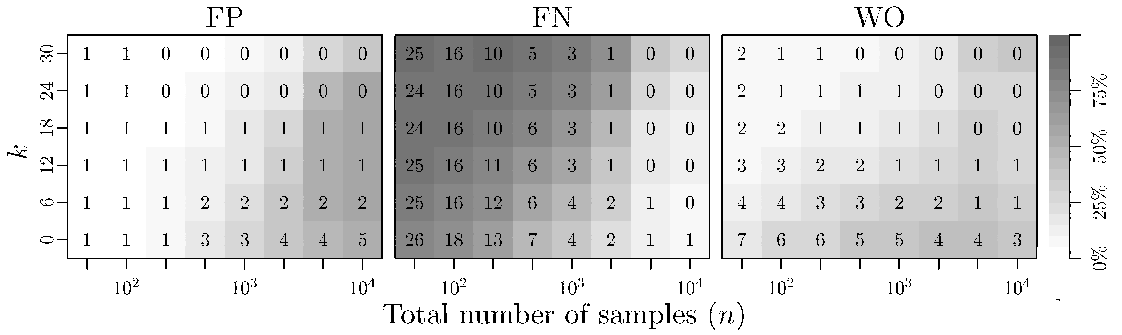


Figure 12: False positives (FP) and false negatives (FN) of the skeleton and wrongly oriented edges (WO; Section 5.2.3) of the GIES estimates compared to the true $\mathcal{I}$-essential graphs with $p = 30$ vertices; mean values as a function of k and n for single-vertex interventions. Shading: ratio of each quantity and the SHD between estimated and true $\mathcal{I}$-essential graph (dark means a large contribution to the SHD).

As noted in Section 5.2.3, the SHD between true and estimated interventional essential graphs can be written as the sum of false positives of the skeleton, false negatives of the skeleton and wrongly oriented edges. Those numbers are shown in Figure 12, again for GIES estimates under single-vertex interventions for DAGs with $p = 30$ vertices. False positives of the skeleton are the main contribution to the SHD values. In 60% of the cases, especially for large n and small k, wrongly oriented edges represent the second-largest contribution.

*R*untime Analysis

All algorithms evaluated in this section were implemented in C++ and compiled into a library using the GNU compiler g++ 4.6.1. The simulations—that is, the generation of data and the library calls—were performed using R 2.13.1. All simulations were run on an AMD Opteron 8380 CPU with 2.5 GHz and 2 GB RAM.

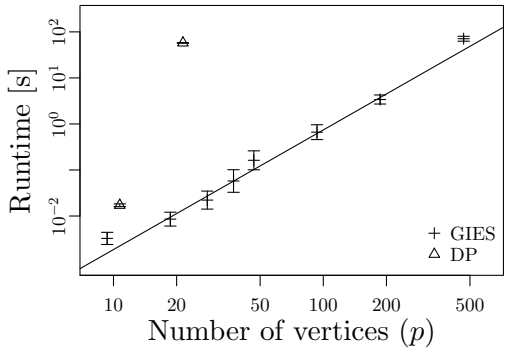


Figure 13: Runtime of GIES and DP as a function of the vertex number.

Figure 13 shows the running times of GIES and DP as a function of the number p of vertices. GDS had running times of the same order of magnitude as GIES; they were actually up to 50% higher since we used a basic implementation of GDS compared to an optimized version of GIES (running times of GDS are not plotted for this reason). The linearity of the GIES values in the log-log plot (see the solid line in Figure 13) indicate a polynomial time complexity of the approximate order $O(p^{2.8})$, in contrast to the exponential complexity of DP; note that GIES also has an exponential *worst case* complexity (see Section 4.4). The multiple linear regression $\log(t) = \beta_0 + \beta_1 \log(p) + \beta_2 \log(|E|) + \varepsilon$, where t denotes the runtime and E the edge set of the true DAG, yields coefficients $\hat{\beta}_1 = 1.01$ and $\hat{\beta}_2 = 0.94$.

5.3 DREAM4 Challenge

We also measured the performance of GIES on synthetic gene expression data sets from the DREAM4 *in silico* challenge (Marbach et al., 2010; Prill et al., 2010). Our goal here was to evaluate predictions of expression levels of gene knockout or knockdown experiments by cross-validation based on the provided interventional data.

5.3.1 DATA

The DREAM4 challenge provides five data sets with an ensemble of interventional and observational data simulated from five biologically plausible, possibly *cyclic* gene regulatory networks with 10 genes (Marbach et al., 2009). The data set of each network consists of

- 11 observational measurements, simulated from random fluctuations of the system parameters (resembling observational data measured in different individuals);
- 10 measurements from single-gene knockdowns, one knockdown per gene;
- 10 measurements from single-gene knockouts, one knockout per gene;
- five time series with 21 time points each, simulated from an unknown change of parameters in the first half (corresponding to measurements under a perturbed chemical environment having unknown effects on the gene regulatory network) and from the unperturbed system in the second half.

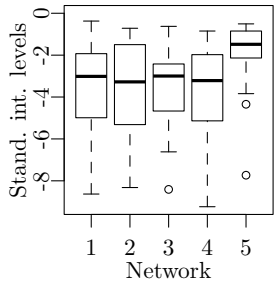


Figure 14: Standardized intervention levels in the different DREAM4 data sets. Data is scaled such that the observational samples have empirical mean 0 and standard deviation 1.

Since our framework can not cope with uncertain interventions (that is, interventions with unknown target), we only used the 50 observational measurements of the second half of the time series. Altogether, we have, from each network, a total of 81 data points, 61 observational and 20 interventional ones. We normalized the data such that the observational samples of each gene have mean 0 and standard deviation 1. In this normalization, 95% of the intervention levels (that is, the expression levels of knocked out or knocked down genes) lie between -8.37 and -0.62 with a mean of -3.30 (Figure 14).

5.3.2 METHODS

We used each interventional measurement (20 per network) as one test data point and predicted its value from a network estimated with training data consisting either of the 80 remaining data points, or the 61 observational measurements alone. We used GIES, GES and PC (Spirtes et al., 2000) to estimate the causal models and evaluated the prediction accuracy by the mean squared error (MSE). We will use abbreviations like "GES(80)" or "PC(61)" to denote GES estimates based on a training set of size 80 or PC estimates based on an observational training set of size 61, respectively.

For a given DAG, we predicted interventional gene expression levels based on the estimated structural equation model after replacing the structural equation of the intervened variable by a constant one; see Section 5.1 for connection between Gaussian causal models and structural equations, especially Equation (5). GES and PC regard all data as observational and yield an observational essential graph. For those algorithms, we enumerated all representative DAGs of the estimated equivalence class using the function `allDags()` of the R package `pcalg` (Kalisch et al., 2012), calculated an expression level with each of them, and took the mean of those predictions. GIES(80) yields a single DAG in each case since the 19 interventional measurements in the training data ensure complete identifiability.

Furthermore, we used the evaluation script provided by the DREAM4 challenge to assess the quality of our network predictions to those sent in to the challenge by participating teams. This evaluation is based on the area under the ROC curve (AUROC) of the true and false positive rate of the edge predictions.

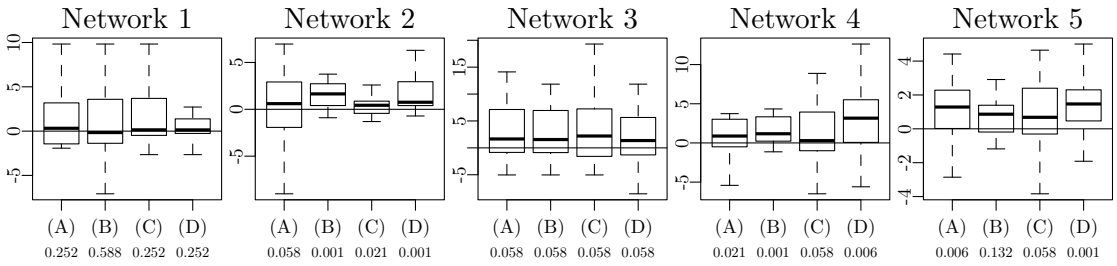


Figure 15: Upper row: MSE values of GIES and competitors; lower row: differences of MSE values as defined in Equation (7); large values indicate a good performance of GIES. (A) GIES(80), (B) PC(80), (C) PC(61), (D) GES(80), (E) GES(61). Numbers below the boxplots: p-values of a one-sided sign test.

5.3.3 RESULTS

Figure 15 shows boxplots of MSE differences between GIES(80) and its competitors; that is, we consider quantities of the form

$$\Delta\text{MSE}_{\text{comp}} := \text{MSE}_{\text{comp}} - \text{MSE}_{\text{GIES(80)}}, \tag{7}$$

where comp stands for one of the competitors. Since the MSE differences showed a skewed distribution in general, we used a sign test for calculating their p-values.

Except for one case (PC(61) in network 1), GIES(80) always yielded the best predictions of all competitors. Although all data sets are dominated by observational data (61 observational measurements versus 20 interventional ones), GIES can make use of the additional information carried by interventional data points to rule out its observational competitors. On the other hand, the dominance of observational data is probably one of the reasons for the fact that GIES does not outperform the observational methods more clearly but has an overall performance which is comparable with that of its competitors. Another reason could be the fact that the underlying networks used for data generation are not acyclic as assumed by GIES. Interestingly, the winning margin of GIES in network 5 was not smaller than in other networks although the corresponding data set has the smallest intervention levels (in absolute values; see Figure 14).

29 teams participated in the DREAM4 challenge. Their AUROC values are available from the DREAM4 website;[1] adding our values gives a data set of 30 evaluations. Among those, our results had overall rank 10, and ranks 8, 4, 21, 10 and 3, respectively, for networks 1 to 5. Except for network 3, we could keep up with the best third of the participating teams despite the beforementioned model misspecification given by the assumption of acyclicity, and despite the fact that we ignored the time series structure and half of the time series data.

6. Conclusion

We gave a definition and a graph theoretic criterion for the Markov equivalence of DAGs under multiple interventions. We characterized corresponding equivalence classes by their *essential graph*,

1. DREAM4 can be found at `http://wiki.c2b2.columbia.edu/dream/index.php/D4c2`.

defined as the union of all DAGs in an equivalence class in analogy to the observational case. Using those essential graphs as a basis for the algorithmic representation of interventional Markov equivalence classes, we presented a new greedy algorithm (including a new turning phase), GIES, for learning causal structures from data arising from multiple interventions.

In a simulation study, we showed that the number of non-orientable edges in causal structures drops quickly even with a small number of interventions; our description of interventional essential graphs makes it possible to *quantify* the gain in identifiability. For a fixed sample size n, GIES estimates got closer to the true causal structure as the number of intervention vertices grew. For DAGs with $p \leq 20$ vertices, the GIES algorithm could keep up with a consistent, exponential-time DP approach maximizing the BIC score. It clearly beat GDS, a simple greedy search on the space of DAGs, as well as GES which cannot cope with interventional data. Our novel turning phase proved to be an improvement of GES even on observational data, as it was already conjectured by Chickering (2002b). Applying GIES to synthetic data sets from the DREAM4 challenge (Marbach et al., 2010), we got better predictions of gene expression levels of knockout or knockdown experiments than with observational estimation methods.

The accurate structure learning performance of GIES in the limit of large data sets raises the question whether GIES is consistent. Chickering (2002b) proved the consistency of GES on observational data. However, the generalization of his proof for GIES operating on interventional data is not obvious since such data are in general not identically distributed.

Acknowledgments

We are grateful to Markus Kalisch for very carefully reading the proofs and for his helpful comments on the text. Furthermore, we would like to thank the anonymous reviewers for constructive comments.

Appendix A. Graphs

In this appendix, we shortly summarize our notation (mostly following Andersson et al., 1997) and basic facts concerning graphs. All statements about perfect elimination orderings that are used in Sections 3 and 4 are listed or proven in Section A.2.

A.1 Definitions and Notation

A **graph** is a pair $G = (V,E)$, where V is a finite set of vertices and $E \subset E^*(V) := (V \times V) \setminus \{(a,a)|a \in V\}$ is a set of edges. We use graphs to denote causal relationships between random variables $X_1,\ldots,X_p$. To keep notation simple, we always assume $V = \{1,2,\ldots,p\} =: [p]$, in order to represent each random variable by its index in the graph.

An edge $(a,b) \in E$ with $(b,a) \in E$ is called **undirected** (or a **line**), whereas an edge $(a,b) \in E$ with $(b,a) \notin E$ is called **directed** (or an **arrow**). Consequently, a graph G is called directed (or undirected, resp.) if all its edges are directed (or undirected, resp.); a directed graph is also called

digraph for short. We use the short-hand notation

$$\begin{aligned} a \longrightarrow b \in G &:\Leftrightarrow (a,b) \in E \wedge (b,a) \notin E, \\ a \text{---} b \in G &:\Leftrightarrow (a,b) \in E \wedge (b,a) \in E, \\ a \cdots\!\cdots b \in G &:\Leftrightarrow (a,b) \in E \vee (b,a) \in E. \end{aligned}$$

A **subgraph** of some graph G is a graph $G' = (V', E')$ with the property $V' \subset V$, $E' \subset E$, denoted by $G' \subset G$. For a subset $A \subset V$ of the vertices of G, the **induced subgraph** on A is $G[A] := (A, E[A])$, where $E[A] := E \cap (A \times A)$. A **v-structure** (also called *immorality* by, for example, Lauritzen, 1996) is an induced subgraph of G of the form $a \longrightarrow b \longleftarrow c$. The **skeleton** of a graph G is the undirected graph $G^u := (V, E^u)$, $E^u := \{(a,b) \in V \times V \mid a \cdots\!\cdots b \in G\}$. For two graphs $G_1 = (V, E_1)$ and $G_2 = (V, E_2)$ on the same vertex set, we define the union and the intersection as $G_1 \cup G_2 := (V, E_1 \cup E_2)$ and $G_1 \cap G_2 := (V, E_1 \cap E_2)$, respectively. For a graph $G = (V, E)$ and $(a,b) \in E^*(V)$, we use the shorthand notation $G - (a,b) := (V, E \setminus \{(a,b)\})$ and $G + (a,b) := (V, E \cup \{(a,b)\})$.

The following sets describe the local environment of a vertex a in a graph G:

$$\begin{aligned} \mathrm{pa}_G(a) &:= \{b \in V \mid b \longrightarrow a \in G\}, \text{ the } \textbf{parents} \text{ of } a, \\ \mathrm{ch}_G(a) &:= \{b \in V \mid a \longrightarrow b \in G\}, \text{ the } \textbf{children} \text{ of } a, \\ \mathrm{ne}_G(a) &:= \{b \in V \mid a \text{---} b \in G\}, \text{ the } \textbf{neighbors} \text{ of } a, \\ \mathrm{ad}_G(a) &:= \{b \in V \mid a \cdots\!\cdots b \in G\}, \text{ the vertices } \textbf{adjacent} \text{ to } a. \end{aligned}$$

The subscripts "G" in the above definitions are omitted when it is clear which graph is meant. For a set $A \subset V$ of vertices, we generalize those definitions as follows:

$$\mathrm{pa}_G(A) := \bigcup_{a \in A} \mathrm{pa}_G(a) \setminus A, \quad \mathrm{ne}_G(A) := \bigcup_{a \in A} \mathrm{ne}_G(a) \setminus A, \text{ etc.}$$

The **degree** of a vertex $a \in V$ is defined as $\deg_G(a) := |\mathrm{ad}_G(a)|$.

For two distinct vertices a and $b \in V$, a **chain** of length k from a to b is a sequence of distinct vertices $\gamma = (a \equiv a_0, a_1, \ldots, a_k \equiv b)$ such that for each $i = 1, \ldots, k$, either $a_{i-1} \longrightarrow a_i \in G$ or $a_{i-1} \longleftarrow a_i \in G$; if for all i, $(a_{i-1}, a_i) \in E$ (that is, $a_{i-1} \longrightarrow a_i \in G$ or $a_{i-1} \text{---} a_i \in G$), the sequence γ is called a **path**. If at least one edge $a_{i-1} \longrightarrow a_i$ is directed in a path, the path is called *directed*, otherwise *undirected*. A **(directed) cycle** is defined as a (directed) path with the difference that $a_0 = a_n$. Paths define a preorder on the vertices of a graph: $a \preceq_G b :\Leftrightarrow \exists$ a path γ from a to b in G. Furthermore, $a \approx_G b :\Leftrightarrow (a \preceq_G b) \wedge (b \preceq_G a)$ is an equivalence relation on the set of vertices.

An undirected graph $G = (V, E)$ is **complete** if all pairs of vertices are adjacent. A **clique** is a subset of vertices $C \subset V$ such that $G[C]$ is complete; a vertex $a \in V$ is called **simplicial** if $\mathrm{ne}(a)$ is a clique. An undirected graph G is called **chordal** if every cycle of length $k \geq 4$ contains a **chord**, that means two nonconsecutive adjacent vertices. For pairwise disjoint subsets $A, B, S \subset V$ with $A \neq \emptyset$ and $B \neq \emptyset$, A and B are **separated** by S in G if every path from a vertex in A to a vertex in B contains a vertex in S.

A **directed acyclic graph**, or **DAG** for short, is a digraph that contains no cycle. In the paper, we mostly use the symbol D for DAGs, whereas arbitrary graphs are, as in this appendix, mostly named G. Chain graphs can be viewed as something between undirected graphs and DAGs: a graph $G = (V, E)$ is a **chain graph** if it contains no directed cycle; undirected graphs and DAGs are

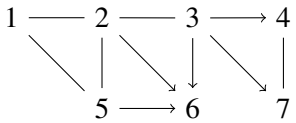


Figure 16: A chain graph G with three chain components $A = T_G(1) = [1]_{\approx_G} = \{1,2,3,5\}$, $B = T_G(6) = \{6\}$ and $C = T_G(4) = \{4,7\}$. The arrows induce the partial order $A \preceq_G B$, $A \preceq_G C$. The graph is no chain graph anymore when we replace the arrow $3 \longrightarrow 4$ by a line since this would create a directed cycle: $(3,7,4,3)$.

special cases of chain graphs. The equivalence classes in V w.r.t. the equivalence relation $\approx_G$ are the connected components of G after removing all directed edges. We denote the quotient set of V by $\mathbf{T}(G) := V/\approx_G$, and its members $T \in \mathbf{T}(G)$ are called **chain components** of G. For a vertex $a \in V$, $T_G(a)$ stands for $[a]_{\approx_G}$. The preorder $\preceq_G$ on V induces in a canonical way a *partial order* on $\mathbf{T}(G)$ which we also denote by $\preceq_G$: $T_G(a) \preceq_G T_G(b) :\Leftrightarrow a \preceq_G b$. An illustration is shown in Figure 16.

An **ordering** of a graph is a bijection $[p] \to V$, hence, since we assume $V = [p]$ here, a permutation $\sigma \in S_p$. An ordering σ canonically induces a total order on V by the definition $a \leq_\sigma b :\Leftrightarrow \sigma^{-1}(a) \leq \sigma^{-1}(b)$. An ordering $\sigma = (v_1, \ldots, v_p)$ is called a **perfect elimination ordering** if for all i, v_i is simplicial in $G^u[\{v_1, \ldots, v_i\}]$. A graph $G = (V,E)$ is a DAG if and only if the previously defined preorder $\preceq_G$ is a partial order; such a partial order can be extended to a total order (Szpilrajn, 1930). Thus every DAG has at least one **topological ordering**, that is an ordering σ whose total order $\leq_\sigma$ extends $\preceq_G$: $a \preceq_G b \Rightarrow a \leq_\sigma b$. For $\sigma \in S_p$, a DAG $D = ([p], E)$ is said to be **oriented according to** σ if σ is a topological ordering of D. In a DAG D with topological ordering σ, the arrows point from vertices with low to vertices with high ordered indices. The vertex $\sigma(1)$ is a **source**, that means all arrows point away from it.

A.2 Perfect Elimination Orderings

Perfect elimination orderings play an important role in the characterization of interventional Markov equivalence classes of DAGs as well as in the implementation of the Greedy Interventional Equivalence Search (GIES). In this section, we provide all results for this topic that are used as auxiliary tools in the proofs of Sections 3 and 4.

Lemma 37 *Let $D = (V,E)$ be a DAG. D has no v-structures if and only if any topological ordering of D is a perfect elimination ordering.*

The proof of this lemma follows easily from the definitions of a v-structure and a perfect elimination ordering. Moreover, if *any* topological ordering of a DAG is a perfect elimination ordering, this is automatically the case for *every* topological ordering.

Proposition 38 (Rose, 1970) *Let $G = (V,E)$ be an undirected graph. Then G is chordal if and only if it has a perfect elimination ordering.*

```
   Input  : An undirected graph G = (V, E)
   Output: An ordering σ of the vertices V, called a LexBFS-ordering
   Σ ← (V); // Initialize sequence Σ of vertex sets to contain the single set V in
      the beginning
   σ ← (); // Initialize output sequence of vertices
3  while Σ ≠ ∅ do
4      Remove a vertex a from the first set in the sequence Σ;
       if first set of Σ is empty then remove first set from Σ;
       Append a to σ;
       Mark all sets of Σ as not visited;
       foreach b ∈ ne_G(a) s.t. b ∈ S for some S ∈ Σ do
           if S not visited then
               Insert empty set T into Σ in front of S;
               Mark S as visited;
           else let T be the set preceding S in Σ;
13         Move b from S to T;
14         if S = ∅ then remove S from Σ;
```

Algorithm 6: LexBFS(V,E). Lexicographic breadth-first search in the so-called "partitioning paradigm" (Rose et al., 1976; Corneil, 2004)

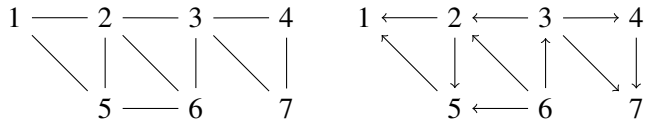
Perfect elimination orderings of chordal graphs can be produced by a variant of the breadth-first search algorithm, the so-called lexicographic breadth-first search (LexBFS; see Algorithm 6). The term "lexicographic" reflects the fact that the algorithm visits edges in lexicographic order w.r.t. the produced ordering σ.

Proposition 39 (Rose et al., 1976) *Let $G = (V,E)$ be an undirected chordal graph with a* LexBFS*-ordering σ. Then σ is also a perfect elimination ordering on G.*

Corollary 40 *Let G be an undirected chordal graph with a* LexBFS*-ordering σ. A DAG $D \subset G$ with $D^{\mu} = G$ that is oriented according to σ has no v-structures.*

Corollary 40 is a consequence of Lemma 37 and Proposition 39. According to this corollary, LexBFS-orderings can be used for constructing representatives of essential graphs (see Proposition 16). Corollary 40 as well as Algorithm 6 are therefore of great importance for the proofs and algorithms of Sections 3 and 4.

Figure 17 shows an undirected chordal graph G and a DAG D that has the skeleton G and is oriented according to a LexBFS-ordering σ of G. The functioning of Algorithm 6 when producing a LexBFS-ordering on G is illustrated in Table 1. Note that the "sets" in Σ are written as tuples. We use this notation to ensure that we can always remove the first (leftmost) vertex from the first "set" of Σ (line 3 in Algorithm 6), and that we keep the relative order of vertices when moving them from one set S to the preceding one, T, in Σ (line 12 in Algorithm 6). Throughout the text, we always assume an implementation of Algorithm 6 in which the data structure used to represent the "sets" in the sequence Σ guarantees this "first in, first out" (FIFO) behavior. In particular, the start sequence $(v_1, v_2, \ldots, v_p)$ of the vertices in V provided to the algorithm determines the vertex the LexBFS-ordering $\sigma := \text{LexBFS}((v_1, \ldots, v_p), E)$ starts with: $\sigma(1) = v_1$. It is often sufficient



(a) G (b) D

Figure 17: An undirected, chordal graph $G = ([7], E)$ and the DAG D we get by orienting all edges of G according to the ordering $\sigma := \text{LexBFS}((6,3,1,2,4,5,7), E)$.

i	Σ	σ
0	$((6,3,1,2,4,5,7))$	$()$
1	$((3,2,5),(1,4,7))$	(6)
2	$((2),(5),(4,7),(1))$	$(6,3)$
3	$((5),(4,7),(1))$	$(6,3,2)$
4	$((4,7),(1))$	$(6,3,2,5)$
5	$((7),(1))$	$(6,3,2,5,4)$
6	$((1))$	$(6,3,2,5,4,7)$
7	$()$	$(6,3,2,5,4,7,1)$

Table 1: State of the sequences Σ and σ after the i^{th} run ($i = 0,\dots,7$) of the while loop (lines 2 to 13) of Algorithm 6 applied to the graph G of Figure 17 with start order $(6,3,1,2,4,5,7)$.

to specify the start order of LexBFS up to arbitrary orderings of some subsets of vertices. For a set $A = \{a_1,\dots,a_k\} \subset V$ and an additional vertex $v \in V \setminus A$, for example, we use the notation

$$\text{LexBFS}((A, v, V \setminus (A \cup \{v\})), E), \quad \text{or even} \quad \text{LexBFS}((A, v, \dots), E)$$

to denote a LexBFS-ordering produced from a start order of the form $(a_1,\dots,a_k,v,\dots)$, without specifying the orderings of A and $V \setminus (A \cup \{v\})$.

By using appropriate data structures (for example, doubly linked lists for the representation of Σ and its sets, and a pointer at each vertex pointing to the set in Σ in which it is contained), Algorithm 6 has complexity $O(|E| + |V|)$ (Corneil, 2004).

For the rest of this section, we state further consequences of Lemma 37 and Proposition 39 which are relevant for the proofs of Sections 3 and 4.

Corollary 41 *Let $G = (V,E)$ be an undirected chordal graph, and let $a - b \in G$. There exist DAGs D_1 and D_2 with $D_1, D_2 \subset G$ and $D_1^u = D_2^u = G$ without v-structures such that $a \longrightarrow b \in D_1$ and $a \longleftarrow b \in D_2$.*

Proof Set $\sigma_1 := \text{LexBFS}((a, V \setminus \{a\}), E)$ and $\sigma_2 := \text{LexBFS}((b, V \setminus \{b\}), E)$, and let D_1 and D_2 be two DAGs with skeleton G and oriented according to σ_1 and σ_2, resp. Then, by Corollary 40, D_1 and D_2 have the requested properties; in particular, all edges point away from a in D_1, whereas all edges point away from b in D_2. ■

Corollary 42 (Andersson et al., 1997) *Let $G = (V,E)$ be an undirected chordal graph, $a \in V$ and $C \subset \mathrm{ne}(a)$. Then there is a DAG $D \subset G$ with $D^u = G$ and $\{b \in \mathrm{ne}(a) \mid b \longrightarrow a \in D\} = C$ that has no v-structures if and only if C is a clique.*

Proof *"⇒":* Assume that there are non-adjacent vertices $b, c \in C$. Then, $b - a - c$ is an induced subgraph of G, and by construction, the same vertices occur in configuration $b \longrightarrow a \longleftarrow c$ in D, which means that D has a v-structure, a contradiction.

"⇐": Let $(c_1, \ldots, c_k)$ be an arbitrary ordering of C. Run LexBFS on a start order of the form $(c_1, \ldots, c_k, a, \ldots)$. After the first run of the while loop (lines 2 to 13 of Algorithm 6), $\sigma = (c_1)$, and the first set in the sequence Σ contains $(C \cup \{a\}) \setminus \{c_1\}$ as a subset (all vertices in this set are adjacent to c_1), in an unchanged order $c_2, \ldots, c_k, a$ due to our FIFO convention. After the second run of the while loop, $\sigma = (c_1, c_2)$, and the first set in Σ contains $(C \cup \{a\}) \setminus \{c_1, c_2\}$, and so on. In the end, we get a LexBFS-ordering of the form $\sigma = (c_1, \ldots, c_k, a, \ldots)$. Orienting the edges of G according to σ yields a DAG with the requested properties by Corollary 40. ■

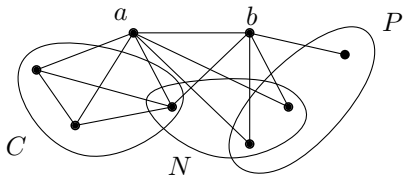


Figure 18: Configuration of vertices in Proposition 43.

Proposition 43 *Let $G = (V,E)$ be an undirected, chordal graph, $a - b \in G$, and $C \subset \mathrm{ne}_G(a) \setminus \{b\}$ a clique. Let $N := \mathrm{ne}_G(a) \cap \mathrm{ne}_G(b)$, and assume that $C \cap N$ separates $C \setminus N$ and $N \setminus C$ in $G[\mathrm{ne}_G(a)]$ (see Figure 18). Then there exists a DAG $D \subset G$ with $D^u = G$ such that*

- *(i) D has no v-structures;*
- *(ii) all edges in $D[C \cup \{a\}]$ point towards a;*
- *(iii) all other edges of D point away from vertices in $C \cup \{a\}$ (in particular, $a \longrightarrow b \in D$);*
- *(iv) $b \longrightarrow d \in D$ for all $d \in P := \mathrm{ne}_G(b) \setminus (C \cup \{a\})$.*

Proof Set $\sigma := \mathrm{LexBFS}((C, a, b, \ldots), E)$, and let D be the DAG that we get by orienting the edges of G according to σ. As in Corollary 42, properties (i) to (iii) are met.

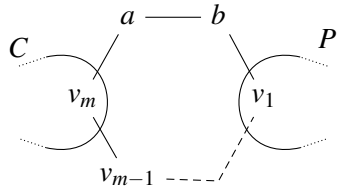
It remains to show that b occurs before any $d \in P$ in σ (that means $b <_\sigma d \; \forall \, d \in P$) in order that D obeys property (iv). W.l.o.g., we can assume $C = \{1, 2, \ldots, k\}$, $a = k+1$ and $b = k+2$. The start order of the vertices for LexBFS is then $(1, 2, \ldots, p)$. Due to the FIFO convention for the sets of the sequence Σ in Algorithm 6, b always precedes any $d \in P$ whenever they appear in the same set; hence we only must show that the set containing b is never preceded by a set containing some $d \in P$ in Σ.

Suppose, for the sake of contradiction, that this is the case for some $d \in P$; name $v_1 := d$. At the beginning, b is in the same set as v_1 in the sequence Σ; there is some vertex v_2 that forces LexBFS to move v_1 into the set preceding the one containing b. A careful inspection of Algorithm 6 shows that v_2 is the vertex which is minimal w.r.t. $\leq_\sigma$ in

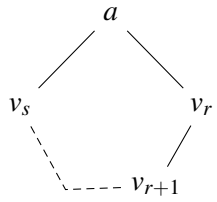
$$S(v_1) := \{v \in V \mid v \in \mathrm{ne}_G(v_1) \setminus \mathrm{ne}_G(b), v <_\sigma b\}.$$

If $v_2 > b$ (that is, if $v_2 \notin C \cup \{a\}$ due to our convention), v_2, as v_1, always follows b whenever they are in the same set in Σ. Therefore, $v_2 <_\sigma b$ implies that there is some vertex v_3 that moves v_2 in the set preceding the one of b in Σ during the execution of LEXBFS; as before, we see that this is the vertex which is minimal w.r.t. $\leq_\sigma$ in $S(v_2)$.

We can now continue to construct this sequence $v_{i+1} := \min S(v_i)$ (always taking the minimum w.r.t. $\leq_\sigma$) until we find some vertex $v_m < b$; this is a vertex in $C \cup \{a\}$. Even more, $v_m \in C \setminus N$, since, by definition of $S(v_{m-1})$, we only consider vertices that are not adjacent to b. We now have constructed a path $\gamma = (v_1, \ldots, v_m)$ of length $m \geq 2$ in G such that $v_1 \in P$, $v_i \notin \mathrm{ne}_G(b)\ \forall\ i > 1$, $v_i > b\ \forall\ i < m$ and $v_m \in C \setminus N$; furthermore, we have $v_m <_\sigma \ldots <_\sigma v_1 <_\sigma b$. The path γ can be elongated to a cycle $(a, v_0 := b, v_1, v_2, \ldots, v_m, a)$:



We now claim that $v_i - a \in G$ for all $0 \leq i \leq m$. This is clearly the case for $i = 0$ and $i = m$ by construction. Assume, for the sake of contradiction, that there is some i, $0 < i < m$, that is not adjacent to a. Let r be the *largest* index smaller than i such that $v_r - a \in G$ and s be the *smallest* index larger than i such that $v_s - a \in G$. Then the following is an *induced* subgraph of G:



Note that a chord between different v_l's, say, a chord of the form $v_l - v_{l+h}$ with $h \geq 2$, would violate the minimality of v_{l+1} in the set $S(v_l)$. This means that G contains an induced cycle of length 4 or more, contradicting the chordality.

This proves the claim that $v_i - a \in G$ for all $0 \leq i \leq m$, or, in other words, $v_i \in \mathrm{ne}_G(a)$ for all $0 \leq i \leq m$. Hence $v_1 \in N \setminus C$, and γ is a path from $N \setminus C$ to $C \setminus N$ in $G[\mathrm{ne}_G(a)]$ that has no vertex in $C \cap N$, in contradiction with the assumption. ■

Proposition 44 *Let $G = (V, E)$ be a chain graph with chordal chain components that does not contain $a \longrightarrow b - c$ as an induced subgraph, and let $D \subset G$ be a digraph with $D^u = G^u$. D is acyclic and has the same v-structures as G if and only if $D[T]$ is oriented according to a perfect elimination ordering for each chain component $T \in \mathbf{T}(G)$.*

Proof "$\Rightarrow$": let $T \in \mathbf{T}(G)$. $G[T]$ obviously does not have any v-structures, hence $D[T]$ has no v-structures, either. It follows from Lemma 37 that $D[T]$ must be oriented according to a perfect elimination ordering.

"⇐": for each $T \in \mathbf{T}(G)$, $D[T]$ is acyclic by construction. Assume that D has some directed cycle γ; this cycle must reach different chain components of G, so it contains at least one edge $a \longrightarrow b$ that is also present in G. Because of $D \subset G$ and $D^u = G^u$, γ is also a cycle in G; and since $a \longrightarrow b \in G$, it is even a *directed* cycle in G, a contradiction. So D is acyclic.

By construction, every v-structure in G is also present in D. Suppose that D has some v-structure $a \longrightarrow b \longleftarrow c$ that G has not. a, b and c cannot belong to the same chain component of G according to Lemma 37. So, w.l.o.g., $a \longrightarrow b - c$ must be an induced subgraph of G, contradicting the assumption. Hence D and G have the same v-structures. ■

Appendix B. Proofs

In this appendix, the technically interested reader finds all proofs that were left out in Sections 2 to 4 for better readability.

B.1 Proofs for Section 2

We start with the proof of Lemma 8 which motivates Definition 7 by showing that, for some DAG D and some (conservative) family of targets $\mathcal{I}$, the elements of $\mathcal{M}_{\mathcal{I}}(D)$ are exactly the density tuples that can be realized as interventional densities of a causal model with structure D. Note that we use the conservativeness of $\mathcal{I}$ only in the proof of point (ii); it can even be proven without assuming conservativeness, although the proof becomes harder.

Proof of Lemma 8

(i) $f(x|\,\mathrm{do}(X_I = U_I))$ obeys the Markov property of $D^{(I)}$ (Section 2.1). Furthermore, for $I, J \in \mathcal{I}$ and $a \notin I \cup J$, we have

$$f(x_a \mid x_{\mathrm{pa}_D(a)}; \mathrm{do}(X_I = U_I)) = f(x_a \mid x_{\mathrm{pa}_D(a)}) = f(x_a \mid x_{\mathrm{pa}_D(a)}; \mathrm{do}(X_J = U_J))$$

by the truncated factorization of Equation (1).

(ii) Let $a \in [p]$. Since $\mathcal{I}$ is conservative, there is some $I \in \mathcal{I}$ such that $a \notin I$. Define $h_a(x_a, x_{\mathrm{pa}_D(a)}) := f^{(I)}(x_a|x_{\mathrm{pa}_D(a)})$. Note that, due to Definition 7, the function h_a does *not* depend on the choice of I.
Let $f(x) := \prod_{a=1}^{p} h_a(x_a, x_{\mathrm{pa}_D(a)})$; this is a positive density on $\mathcal{X}$ with $f(x_a|x_{\mathrm{pa}_D(a)}) = h_a(x_a, x_{\mathrm{pa}_D(a)})$, hence $f \in \mathcal{M}(D)$ and (D, f) is a causal model.
By defining level densities $\tilde{f}_I(x_I) := \prod_{i \in I} f^{(I)}(x_i)$, we can construct an intervention setting $\mathcal{S} := \{(I, \tilde{f}_I)\}_{I \in \mathcal{I}}$ with the requested properties. ■

The proof of the main result of Section 2, the graph theoretic criterion for two DAGs being interventionally Markov equivalent (Theorem 10), requires additional lemmas.

Lemma 45 *Let D be a DAG, $\mathcal{I}$ a family of targets and $I \in \mathcal{I}$ a target in this family. Define*

$$\mathcal{M}^{(I)}(D) := \{f^{(I)} \mid (f^{(J)})_{J \in \mathcal{I}} \in \mathcal{M}_{\mathcal{I}}(D)\}\,,$$

the projection of $\mathcal{M}_{\mathcal{I}}(D)$ to the density component associated with the intervention target I. Then, $\mathcal{M}^{(I)}(D) = \mathcal{M}(D^{(I)})$.

Proof The inclusion "$\subset$" is immediately clear from Definition 7. It remains to show "$\supset$".

Let $f \in \mathcal{M}(D^{(I)})$. Since $D^{(I)} \subset D$, f also obeys the Markov property of D; this means $f \in \mathcal{M}(D)$. Set $\tilde{f}_I(x_I) := f(x_I)$; since $f \in \mathcal{M}(D^{(I)})$, the components of $\tilde{f}_I$ are independent. For $J \in \mathcal{I}$, $J \neq I$, let $\tilde{f}_J$ be an arbitrary level density on $\mathcal{X}_J$. By Lemma 8(i), we know that, for intervention variables $U_J \sim \tilde{f}_J$ ($J \in \mathcal{I}$),

$$\big(f(\cdot \mid \mathrm{do}_D(X_J = U_J))\big)_{J\in\mathcal{I}} \in \mathcal{M}_{\mathcal{I}}(D) \,,$$

hence $f(\cdot \mid \mathrm{do}_D(X_I = U_I)) \in \mathcal{M}^{(I)}(D)$ by definition of $\mathcal{M}^{(I)}(D)$. Moreover, by construction of $\tilde{f}_I$, we have $f(x \mid \mathrm{do}_D(X_I = U_I)) = f(x)$ and hence $f \in \mathcal{M}^{(I)}(D)$. ■

Lemma 46 *Let D be a DAG, $f \in \mathcal{M}(D)$, and $A \subset [p]$. Then,*

$$\prod_{a\in A} f(x_a \mid x_{\mathrm{pa}(a)}) = f(x_A \mid x_{\mathrm{pa}(A)}).$$

Proof Let $\sigma \in S_p$ be a topological ordering of D. Then, for $a \in A$,

$$\mathrm{pa}(a) \subset \mathrm{pa}(A) \cup \big[A \cap \sigma^{-1}(\{1,\ldots,a-1\})\big] \tag{8}$$

holds: every $b \in \mathrm{pa}(a)$ either lies in A^c and hence in $\mathrm{pa}(A)$ by the definition given in Appendix A.1, or in $A \cap \sigma^{-1}(\{1,\ldots,a-1\})$ by the definition of a topological ordering.

Hence we conclude

$$f(x_A \mid x_{\mathrm{pa}(A)}) = \prod_{a\in A} f(x_a \mid x_{A\cap\sigma^{-1}(\{1,\ldots,a-1\})}, x_{\mathrm{pa}(A)}) = \prod_{a\in A} f(x_a \mid x_{\mathrm{pa}(a)});$$

the first equality is the usual factorization of a density, the second equality follows from the Markov properties of f and Equation (8). ■

Lemma 47 *Let $\mathcal{I}$ be a family of targets. Assume D_1 and D_2 are DAGs with the same skeleton and the same v-structures such that $D_1^{(I)}$ and $D_2^{(I)}$ have the same skeleton for all $I \in \mathcal{I}$. Moreover, let $a \longrightarrow b \in D_1$. If there is some $I \in \mathcal{I}$ such that $|I \cap \{a,b\}| = 1$, then the arrow is also present in D_2: $a \longrightarrow b \in D_2$.*

Proof Since D_1 and D_2 have the same skeleton, we have at least $a \cdots\!\cdots b \in D_2$. Suppose $a \longleftarrow b \in D_2$. If $a \in I$, $b \notin I$, a and b are adjacent in $D_1^{(I)}$, but not in $D_2^{(I)}$, hence $D_1^{(I)}$ and $D_2^{(I)}$ have a different skeleton, a contradiction. On the other hand, if $a \notin I$ but $b \in I$, a and b are not adjacent in $D_1^{(I)}$, but in $D_2^{(I)}$, a contradiction, too. ■

Proof of Theorem 10 *(i) $\Rightarrow$ (ii):* Let $I \in \mathcal{I}$, and let $\mathcal{M}^{(I)}(D_1)$ and $\mathcal{M}^{(I)}(D_2)$ be defined as in Lemma 45. By Definition 9 of interventional Markov equivalence, it follows that $\mathcal{M}^{(I)}(D_1) = \mathcal{M}^{(I)}(D_2)$; hence $\mathcal{M}(D_1^{(I)}) = \mathcal{M}(D_2^{(I)})$ by Lemma 45.

(ii) $\Rightarrow$ (iii): this implication follows from Theorem 3.

(iii) ⇒ (iv): Let $a \longrightarrow b \in D_1$ be an arrow. Since $\mathcal{I}$ is conservative, there is some $I \in \mathcal{I}$ such that $b \notin I$. For this I, $a \longrightarrow b \in D_1^{(I)}$, so $a \cdots\!\cdots b \in D_2^{(I)}$ by assumption and hence $a \cdots\!\cdots b \in D_2$ because of $D_2^{(I)} \subset D_2$. Similarly, we can show the implication $a \longrightarrow b \in D_2 \;\Rightarrow\; a \cdots\!\cdots b \in D_1$, what proves that D_1 and D_2 have the same skeleton.

It remains to show that D_1 and D_2 also have the same v-structures. Let $a \longrightarrow b \longleftarrow c$ be a v-structure of D_1. There is some $I \in \mathcal{I}$ that does not contain b; $a \longrightarrow b \longleftarrow c$ is then an induced subgraph of $D_1^{(I)}$ and hence by assumption also of $D_2^{(I)}$. By consequence, $a \longrightarrow b \longleftarrow c$ is also an induced subgraph of D_2 since D_2 has the same skeleton as D_1. The argument is of course symmetric w.r.t. exchanging D_1 and D_2.

(iv) ⇒ (i): Let $(f^{(I)})_{I\in\mathcal{I}} \in \mathcal{M}_{\mathcal{I}}(D_1)$. By Lemma 8(ii), there is some density $f \in \mathcal{M}(D_1)$ and some intervention setting $\mathcal{S} = \{(I, \tilde{f}_I)\}_{I\in\mathcal{I}}$ such that $f^{(I)}(\cdot) = f(\cdot \,|\, \mathrm{do}_{D_1}(X_I = U_I))$ for random variables $U_I \sim \tilde{f}_I$, $I \in \mathcal{I}$.

The truncated factorization in Equation (1) tells us

$$\begin{aligned} f(x \mid \mathrm{do}_{D_1}(X_I = U_I)) &= \prod_{a\notin I} f(x_a \mid x_{\mathrm{pa}_{D_1}(a)}) \prod_{a\in I} \tilde{f}_I(x_a) = f(x) \prod_{a\in I} \frac{\tilde{f}_I(x_a)}{f(x_a \mid x_{\mathrm{pa}_{D_1}(a)})} \\ &= f(x) \frac{\tilde{f}_I(x_I)}{f(x_I \mid x_{\mathrm{pa}_{D_1}(I)})}. \end{aligned} \tag{9}$$

The last step uses Lemma 46.

We now claim that $\mathrm{pa}_{D_1}(I) = \mathrm{pa}_{D_2}(I)$. Indeed, if $b \in I$ and $a \in \mathrm{pa}_{D_1}(b) \setminus I$, $a \longrightarrow b$ is an arrow in D_1 with $|I \cap \{a,b\}| = 1$, hence $a \longrightarrow b \in D_2$ by Lemma 47 and therefore $a \in \mathrm{pa}_{D_2}(I)$; the argument is symmetric w.r.t. exchanging D_1 and D_2. It follows that $f(x_I|x_{\mathrm{pa}_{D_1}(I)}) = f(x_I|x_{\mathrm{pa}_{D_2}(I)})$, and by repeating the calculation in (9) for D_2 instead of D_1, we find $f(x|\,\mathrm{do}_{D_1}(X_I = U_I)) = f(x|\,\mathrm{do}_{D_2}(X_I = U_I))$.

Since this equality is true for all $I \in \mathcal{I}$, we have $f^{(I)}(\cdot) = f(\cdot|\,\mathrm{do}_{D_2}(X_I = U_I))$ for all $I \in \mathcal{I}$, so $(f^{(I)})_{I\in\mathcal{I}} \in \mathcal{M}_{\mathcal{I}}(D_2)$ by Lemma 8(i), which proves $\mathcal{M}_{\mathcal{I}}(D_1) \subset \mathcal{M}_{\mathcal{I}}(D_2)$. The other direction is completely analogous. ■

Points (i) to (iii) are even equivalent under non-conservative families of targets. The proof is more difficult in this case though.

B.2 Proofs for Section 3

All statements of Section 3.2 are similar to analogous statements for the observational case developed by Andersson et al. (1997). Some of the proofs given there are even literally valid also for our interventional setting; in such cases, we will not repeat them here, but just refer to the original ones. However, in most cases, the generalization from the observational to the interventional case is not obvious and requires adapted techniques presented in this section. Here, $\mathcal{I}$ always stands for a conservative family of targets.

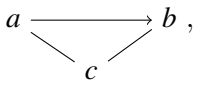
First, we show that for some DAG D, $\mathcal{E}_{\mathcal{I}}(D)$ is a chain graph (Proposition 15). For that purpose, we define $\mathcal{E}_{\mathcal{I}}(D)^*$ as the smallest chain graph containing $\mathcal{E}_{\mathcal{I}}(D)$. $\mathcal{E}_{\mathcal{I}}(D)^*$ is obtained from $\mathcal{E}_{\mathcal{I}}(D)$ by converting all arrows that are part of a directed cycle in $\mathcal{E}_{\mathcal{I}}(D)$ into lines (Andersson et al.,

1997). We first state a couple of properties of $\mathcal{E}_{\mathcal{I}}(D)$ and $\mathcal{E}_{\mathcal{I}}(D)^*$ (Lemma 48), and then show that $\mathcal{E}_{\mathcal{I}}(D)^* = \mathcal{E}_{\mathcal{I}}(D)$ (Proposition 15).

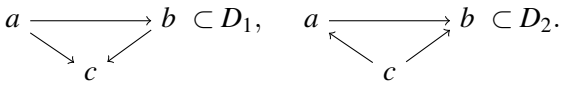
Lemma 48 (adapted from Andersson et al., 1997) *Let D be a DAG. Then:*

(i) *$\mathcal{E}_{\mathcal{I}}(D)$ has no induced subgraph of the form $a \longrightarrow b - c$.*

(ii) *If $\mathcal{E}_{\mathcal{I}}(D)$ has an induced subgraph of the form*



then there exist $D_1, D_2 \in [D]_{\mathcal{I}}$ such that



(iii) *$\mathcal{E}_{\mathcal{I}}(D)^*$ has the same v-structures as D (and hence as $\mathcal{E}_{\mathcal{I}}(D)$).*

(iv) *$\mathcal{E}_{\mathcal{I}}(D)$ and $\mathcal{E}_{\mathcal{I}}(D)^*$ do not have any undirected chordless k-cycle of length $k \geq 4$.*

(v) *$\mathcal{E}_{\mathcal{I}}(D)^*$ has no induced subgraph of the form $a \longrightarrow b - c$.*

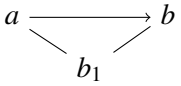
(vi) *If two vertices a and b are adjacent in $\mathcal{E}_{\mathcal{I}}(D)^*$ and there is some $I \in \mathcal{I}$ such that $|I \cap \{a,b\}| = 1$, then the edge between a and b is directed in $\mathcal{E}_{\mathcal{I}}(D)$ and $\mathcal{E}_{\mathcal{I}}(D)^*$.*

Proof Points (i) to (v) correspond to Facts 1 to 5 of Andersson et al. (1997) where these properties were proven for observational essential graphs. A thorough inspection of the proofs given there reveals that they only make use of the fact that two Markov equivalent DAGs have the same skeleton and the same v-structures, which is also true in the interventional case by Theorem 10. Thanks to this, the proofs of Andersson et al. (1997) can be literally used here. (Note that the inverse implication also holds in the observational case, but not in the interventional one; see the discussion after Theorem 10.)

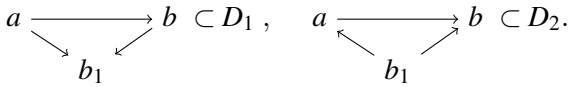
It remains to prove point (vi). The edge between a and b in $\mathcal{E}_{\mathcal{I}}(D)$ is directed since the arrow between a and b is $\mathcal{I}$-essential in D by Corollary 13. It remains to show that the edge is also directed in $\mathcal{E}_{\mathcal{I}}(D)^*$, that is, to show that it is *not* part of a directed cycle in $\mathcal{E}_{\mathcal{I}}(D)$.

Let's suppose, for the sake of contradiction, that the edge between a and b is part of a directed cycle $\gamma = (a, b \equiv b_0, b_1, \ldots, b_k \equiv a)$ in $\mathcal{E}_{\mathcal{I}}(D)$. W.l.o.g., we can assume that $a \longrightarrow b \in \mathcal{E}_{\mathcal{I}}(D)$, and that γ is the *shortest* such cycle containing a directed edge with one end point in I and the other one outside I.

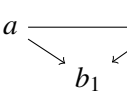
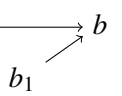
Case 1: $k = 2$. Then γ is of the form

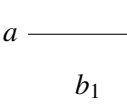
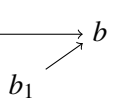


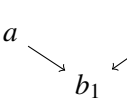
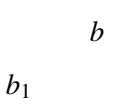
since two or three directed edges would imply the existence of a digraph with a cycle in the equivalence class of D. By point (ii), there are DAGs D_1 and D_2 in $[D]_{\mathcal{I}}$ such that

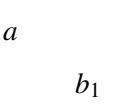
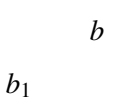


The condition $|I \cap \{a,b\}| = 1$ leaves four possibilities:

a) $a \in I; b, b_1 \notin I$: then,  $\subset D_1^{(I)}$,  $\subset D_2^{(I)}$;

b) $a, b_1 \in I; b \notin I$: then,  $\subset D_1^{(I)}$,  $\subset D_2^{(I)}$;

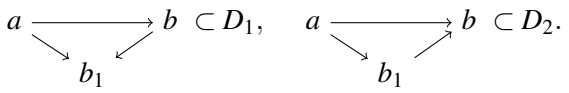
c) $b \in I; a, b_1 \notin I$: then,  $\subset D_1^{(I)}$,  $\subset D_2^{(I)}$;

d) $b, b_1 \in I; a \notin I$: then,  $\subset D_1^{(I)}$,  $\subset D_2^{(I)}$.

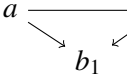
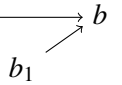
In all four cases, $(D_1^{(I)})^u \neq (D_2^{(I)})^u$, hence $D_1 \not\sim_{\mathcal{I}} D_2$, a contradiction.

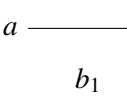
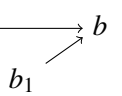
Case 2: $k \geq 3$. Let i be the smallest index such that $b_i - b_{i+1} \in \mathcal{E}_{\mathcal{I}}(D)$ (there must be such an index, otherwise γ would be a directed cycle in D).

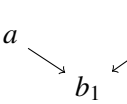
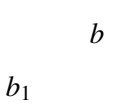
Case 2.1: $i = 0$. Since $a \longrightarrow b - b_1$ cannot be an induced subgraph of $\mathcal{E}_{\mathcal{I}}(D)$ by point (i), we must have $a \cdots b_1 \in \mathcal{E}_{\mathcal{I}}(D)$. More precisely, we must have $a \longrightarrow b_1 \in \mathcal{E}_{\mathcal{I}}(D)$, otherwise (a, b, b_1, a) would form a shorter directed cycle than γ, in contradiction to the assumption. This means that there exist DAGs $D_1, D_2 \in [D]_{\mathcal{I}}$ such that

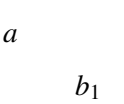
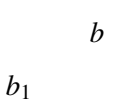


Again, the condition $|I \cap \{a, b\}| = 1$ leaves four possibilities:

a) $a \in I; b, b_1 \notin I$: then,  $\subset D_1^{(I)}$,  $\subset D_2^{(I)}$;

b) $a, b_1 \in I; b \notin I$: then,  $\subset D_1^{(I)}$,  $\subset D_2^{(I)}$;

c) $b \in I; a, b_1 \notin I$: then,  $\subset D_1^{(I)}$,  $\subset D_2^{(I)}$;

d) $b, b_1 \in I; a \notin I$: then,  $\subset D_1^{(I)}$,  $\subset D_2^{(I)}$.

Cases *b)* and *c)* are not compatible with the condition $(D_1^{(I)})^u = (D_2^{(I)})^u$. In cases *a)* and *d)*, the arrow $a \longrightarrow b_1$ is part of a directed cycle $(a, b_1, b_2, \ldots, b_k \equiv a)$, furthermore $|I \cap \{a, b_1\}| = 1$; this contradicts the assumption of minimality of the larger cycle γ.

Case 2.2: $i \geq 1$. Since $b_{i-1} \longrightarrow b_i - b_{i+1}$ cannot be an induced subgraph of $\mathcal{E}_{\mathcal{I}}(D)$, we must have $b_{i-1} \cdots\cdots b_{i+1} \in \mathcal{E}_{\mathcal{I}}(D)$. Either $b_{i-1} \longleftarrow b_{i+1} \in \mathcal{E}_{\mathcal{I}}(D)$, that is

$$b_{i-1} \longrightarrow b_i \ \subset \mathcal{E}_{\mathcal{I}}(D) \ , \quad \nwarrow \ \swarrow \ b_{i+1}$$

which would imply the existence of a digraph with a directed 3-cycle in the equivalence class of D, a contradiction. The other cases are $b_{i-1} \longrightarrow b_{i+1} \in \mathcal{E}_{\mathcal{I}}(D)$ or $b_{i-1} - b_{i+1} \in \mathcal{E}_{\mathcal{I}}(D)$ which would mean that $a \longrightarrow b$ would be part of a shorter directed cycle $(a, b \equiv b_0, \ldots, b_{i-1}, b_{i+1}, \ldots, b_k \equiv a)$, contradicting the assumption of minimality of the cycle γ. ■

Proof of Proposition 15 We only prove the first point; the second one is an immediate consequence of Lemma 48(iv). We have to show that $\mathcal{E}_{\mathcal{I}}(D) = \mathcal{E}_{\mathcal{I}}(D)^*$, that means that

$$a - b \in \mathcal{E}_{\mathcal{I}}(D)^* \ \Rightarrow \ a - b \in \mathcal{E}_{\mathcal{I}}(D) \ .$$

By Lemma 48(iv), all chain components of $\mathcal{E}_{\mathcal{I}}(D)^*$ are chordal. Let D_1 and D_2 be two DAGs that are obtained by orienting all chain components of $\mathcal{E}_{\mathcal{I}}(D)^*$ according to some perfect elimination ordering, such that $a \longrightarrow b \in D_1$ and $a \longleftarrow b \in D_2$; such orientations exist by Proposition 44 and Corollary 41.

We now claim that $D_1 \sim_{\mathcal{I}} D_2$ by verifying the criteria of Theorem 10(iv); it then follows that $a - b \in \mathcal{E}_{\mathcal{I}}(D)$ because of $D_1 \cup D_2 \subset \mathcal{E}_{\mathcal{I}}(D)$:

- By Proposition 44, D_1 and D_2 have the same skeleton and the same v-structures.
- $D_1^{(I)}$ and $D_2^{(I)}$ have the same skeleton for all $I \in \mathcal{I}$: suppose, for the sake of contradiction, that $(D_1^{(I)})^u$ has some edge $c - d$ that $(D_2^{(I)})^u$ has not. W.l.o.g., we then have $c \longrightarrow d \in D_1$, $c \longleftarrow d \in D_2$, $c \in I$, $d \notin I$. But then c and d are adjacent in $\mathcal{E}_{\mathcal{I}}(D)^*$ with $|I \cap \{c,d\}| = 1$, hence the edge between c and d must be oriented in $\mathcal{E}_{\mathcal{I}}(D)^*$ by point (vi) of Lemma 48, and hence it is not possible that this edge has two different orientations in D_1 and D_2 by their construction. ■

Proof of Proposition 16 *"⇐":* By the construction of $\mathcal{E}_{\mathcal{I}}(D)$, we know that $D \subset \mathcal{E}_{\mathcal{I}}(D)$ and $D^u = \mathcal{E}_{\mathcal{I}}(D)^u$. Furthermore, D has the same v-structures as $\mathcal{E}_{\mathcal{I}}(D)$. Let D' be another digraph that is obtained by orienting all chain components of $\mathcal{E}_{\mathcal{I}}(D)$ according to a perfect elimination ordering; by Proposition 44, D' is acyclic and has the same v-structures as $\mathcal{E}_{\mathcal{I}}(D)$ and hence as D. It remains to show that $D^{(I)}$ and $D'^{(I)}$ have the same skeleton for all $I \in \mathcal{I}$; this can be done similarly to the proof of Proposition 15.

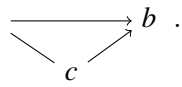
"⇒": let D' be a DAG with $D' \sim_{\mathcal{I}} D$. In particular, D' and D have the same skeleton and the same v-structures, so D' also has the same skeleton and the same v-structures as $\mathcal{E}_{\mathcal{I}}(D)$. It follows, with Proposition 44, that D' is oriented according to a perfect elimination ordering on all chain components of $\mathcal{E}_{\mathcal{I}}(D)$. ■

Lemma 49 *Let D be a DAG and $a \longrightarrow b$ an $\mathcal{I}$-essential arrow in D. Then $a \longrightarrow b$ is strongly $\mathcal{I}$-protected in $\mathcal{E}_{\mathcal{I}}(D)$.*

This lemma is an auxiliary result needed to prove Theorem 18. In its proof, we first show the weaker statement that every $\mathcal{I}$-essential arrow of D is $\mathcal{I}$-protected in $\mathcal{E}_{\mathcal{I}}(D)$.

Definition 50 (Protection) *Let G be a graph. An arrow $a \longrightarrow b \in G$ is $\mathcal{I}$-**protected** in G if there is some intervention target $I \in \mathcal{I}$ such that $|I \cap \{a,b\}| = 1$, or $\mathrm{pa}_G(a) \neq \mathrm{pa}_G(b) \setminus \{a\}$.*

This definition is again a generalization of the notion of protection of Andersson et al. (1997); for $\mathcal{I} = \{\emptyset\}$, we gain back their definition. A *strongly* $\mathcal{I}$-protected arrow (Definition 14) is also $\mathcal{I}$-protected. In a chain graph G, an arrow $a \longrightarrow b$ is $\mathcal{I}$-protected if and only if there is some $I \in \mathcal{I}$ such that $|I \cap \{a,b\}| = 1$, or the arrow $a \longrightarrow b$ occurs in at least one subgraph of the form (a), (b), (c) in the notation of Definition 14, or in a subgraph of the form (d') (Andersson et al., 1997), where

(d'):  .

Proof of Lemma 49 As foreshadowed, we prove this lemma in two steps, corresponding to Facts 6 and 7 of Andersson et al. (1997): in a first step, we show that $a \longrightarrow b$ must be $\mathcal{I}$-protected, in a second step, we strengthen the result by showing that it must even be *strongly* $\mathcal{I}$-protected. For notational convenience, we abbreviate $G := \mathcal{E}_{\mathcal{I}}(D)$. We skip some steps of the proof that can be literally copied from proofs in Andersson et al. (1997).

Suppose, for the sake of contradiction, that $a \longrightarrow b$ is not $\mathcal{I}$-protected. Let D_1 be a digraph that is gained by orienting all chain components of G according to a perfect elimination ordering, where the edges of $T_G(a)$ and $T_G(b)$ are oriented such that all edges point away from a or b, respectively. Then D_1 is acyclic and $\mathcal{I}$-equivalent to D by Proposition 16.

Let D_2 be another digraph, differing from D_1 only by the orientation of the edge between a and b. It can be shown that D_2 is acyclic too (Andersson et al., 1997, proof of Fact 6). We now claim that $D_1 \sim_{\mathcal{I}} D_2$:

- D_1 and D_2 clearly have the same skeleton.
- D_1 and D_2 have the same v-structures. Otherwise, there would be some v-structure $c \longrightarrow a \longleftarrow b$ in D_2, or some v-structure $a \longrightarrow b \longleftarrow c$ in D_1. In both cases, this would imply $\mathrm{pa}_G(a) \neq \mathrm{pa}_G(b) \setminus \{a\}$, contradicting the assumption: in the first case, $c \notin T_G(a)$ by construction (all edges of $T_G(a)$ point away from a in D_2), so $c \in \mathrm{pa}_G(a)$, but $c \notin \mathrm{pa}_G(b)$; in the second case, $c \in \mathrm{pa}_G(b)$, but $c \notin \mathrm{pa}_G(a)$ by analogous arguments.
- $(D_1^{(I)})^u = (D_2^{(I)})^u$ for all $I \in \mathcal{I}$. Otherwise, there would be some $I \in \mathcal{I}$ such that the skeletons of $D_1^{(I)}$ and $D_2^{(I)}$ differ in the edge between a and b. This could only happen if $|I \cap \{a,b\}| = 1$, in contradiction with the assumption.

Hence, since $D_1, D_2 \in [D]_{\mathcal{I}}$, we have $D_1 \cup D_2 \subset G$ and thus $a - b \in G$, a contradiction. This proves that $a \longrightarrow b$ is $\mathcal{I}$-protected in G.

In the second step, we show that $a \longrightarrow b$ is even *strongly* $\mathcal{I}$-protected. If this was not the case, $a \longrightarrow b$ would occur in configuration (d') in G, but *not* in configuration (a), (b), (c) or (d) (see the comment following Definition 50). Define $P_a := \{d \in T_G(a) \mid d \longrightarrow b \in G\}$. It can be shown that P_a is a clique $G[T_G(a)]$ (Andersson et al., 1997, proof of Fact 7).

Let D_1 be the DAG that we get by orienting all chain components of G according to a perfect elimination ordering, such that, additionally,

- all edges of $D_1[T_G(b)]$ point away from b,
- all edges of $D_1[P_a]$ point towards a,
- and all other edges of $D_1[T_G(a)]$ point away from a.

Such an orientation exists by Corollary 42. Let D_2 be the digraph that we get by changing the orientation of the edge $a \longrightarrow b$ in D_1; as in the first part, it can be shown that D_2 is acyclic (Andersson et al., 1997, proof of Fact 7). Again, we claim that $D_1 \sim_{\mathcal{I}} D_2$:

- D_1 and D_2 clearly have the same skeleton.
- D_1 and D_2 have the same v-structures. Otherwise, there would be some v-structure $d \longrightarrow a \longleftarrow b$ in D_2, or a v-structure $a \longrightarrow b \longleftarrow d$ in D_1. In the first case, $d \notin P_a$ (otherwise, $d \longrightarrow b \in G$ by definition of P_a, and hence $d \longrightarrow b \in D_2$ since $D_2 \subset G$), and $d \notin T_G(a) \setminus P_a$ by construction (edges in $T_G(a) \setminus P_a$ point away from a in D_2), hence $d \longrightarrow a \in G$ and $a \longrightarrow b$ is in configuration (a) in G; in the second case, $d \notin T_G(b)$ by construction (all edges of $T_G(b)$ point away from b in D_1), so $a \longrightarrow b$ is in configuration (b) (notation of Definition 14) in G. Both cases contradict the assumption.
- Exactly as in the first part, $(D_1^{(I)})^u = (D_2^{(I)})^u$ for all $I \in \mathcal{I}$.

We can conclude that, since $D_1, D_2 \in [D]_{\mathcal{I}}$, $D_1 \cup D_2 \subset G$, so $a \longrightarrow b \in G$, a contradiction. ■

Proof of Theorem 18 *"⇒":* (i) and (ii) follow from Proposition 15, (iii) from Lemma 48(v), (iv) from Corollary 13 and (v) from Lemma 49.

"⇐": Consider the set $\mathbf{D}(G)$ of all DAGs that can be obtained by orienting the chain components of G according to a perfect elimination ordering; we have $\bigcup \mathbf{D}(G) \subset G$. On the other hand, for each undirected edge $a \longrightarrow b \in G$, there are DAGs D_1 and D_2 in $\mathbf{D}(G)$ such that $a \longrightarrow b \in D_1$, $a \longleftarrow b \in D_2$ (Corollary 41), hence $G \subset \bigcup \mathbf{D}(G)$. Together, we find $G = \bigcup \mathbf{D}(G)$.

We claim that $D_1 \sim_{\mathcal{I}} D_2$ for any two DAGs $D_1, D_2 \in \mathbf{D}(G)$:

- D_1 and D_2 have the same skeleton and the same v-structures by Proposition 44.
- $(D_1^{(I)})^u = (D_2^{(I)})^u$ for all $I \in \mathcal{I}$. Otherwise, there would be arrows $a \longrightarrow b \in D_1$, $a \longleftarrow b \in D_2$, and some $I \in \mathcal{I}$ such that $|I \cap \{a,b\}| = 1$; this would mean that $a \longrightarrow b \in G$ although $|I \cap \{a,b\}| = 1$, contradicting property (iv).

Let $D \in \mathbf{D}(G)$. We have shown that $\mathbf{D}(G) \subset [D]_{\mathcal{I}}$, hence $G = \bigcup \mathbf{D}(G) \subset \mathcal{E}_{\mathcal{I}}(D)$. It remains to show that $G \supset \mathcal{E}_{\mathcal{I}}(D)$.

Assume, for the sake of contradiction, that G has some arrow $a \longrightarrow b$ where $\mathcal{E}_{\mathcal{I}}(D)$ has an undirected edge $a \longrightarrow b$. According to property (v), $a \longrightarrow b$ is strongly $\mathcal{I}$-protected in G. If there was some $I \in \mathcal{I}$ such that $|I \cap \{a,b\}| = 1$, the edge between a and b was also directed in $\mathcal{E}_{\mathcal{I}}(D)$ by Corollary 13, a contradiction. Hence $a \longrightarrow b$ occurs in G in one of the configurations depicted in Definition 14. Exactly as in the proof of Theorem 4.1 of Andersson et al. (1997), we can construct a contradiction for each of the four configurations. Although the proof given there can be used literally, we reproduce it here since since we will use the following steps again in the proof of Lemma 22.

We assume w.l.o.g. that $T_G(a)$ is minimal in

$$A := \{T \in \mathbf{T}(G) | \exists\, a \in T, b \in V(G) : a \longrightarrow b \in G, a - b \in \mathcal{E}_{\mathcal{I}}(D)\}$$

w.r.t. $\preceq_G$, and that $T_G(b)$ is minimal in

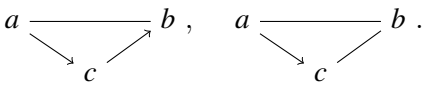
$$B := \{T \in \mathbf{T}(G) | \exists\, a \in T(a), b \in T : a \longrightarrow b \in G, a - b \in \mathcal{E}_{\mathcal{I}}(D)\}.$$

Each configuration (a) to (d) of Definition 14 leads to a contradiction (c, c_1 and c_2 denote the vertices involved in the respective configuration):

(a) Because of the minimality of $T_G(a)$, $c \longrightarrow a$ must be oriented in $\mathcal{E}_{\mathcal{I}}(D)$, hence $c \longrightarrow a - b$ is an induced subgraph of $\mathcal{E}_{\mathcal{I}}(D)$, contradicting Lemma 48(i).

(b) $a \longrightarrow b \longleftarrow c$ is then a v-structure in D, hence it is also a v-structure in $\mathcal{E}_{\mathcal{I}}(D)$, that means $a \longrightarrow b \in \mathcal{E}_{\mathcal{I}}(D)$, a contradiction.

(c) Because of the minimality of $T_G(b)$, the edge between a and c must be oriented in $\mathcal{E}_{\mathcal{I}}(D)$, so the vertices a, b and c are in one of the following configurations in $\mathcal{E}_{\mathcal{I}}(D)$:



Both possibilities violate Proposition 15(i).

(d) The v-structure $c_1 \longrightarrow b \longleftarrow c_2$ of D is also a v-structure of $\mathcal{E}_{\mathcal{I}}(D)$, hence $\mathcal{E}_{\mathcal{I}}(D)$ has two directed 3-cycles (c_1, b, a, c_1) and (c_2, b, a, c_2), a contradiction. ■

Proof of Lemma 20

(i) This immediately follows from Theorem 18(iii).

(ii) Let $a \longrightarrow b$ be an arrow in $\mathcal{E}_{\mathcal{I}}(D)$; by Theorem 18(v), it is strongly $\mathcal{I}$-protected in $\mathcal{E}_{\mathcal{I}}(D)$. If there is some $I \in \mathcal{I}$ such that $|I \cap \{a, b\}| = 1$, the arrow is by definition also strongly $\mathcal{I}$-protected in G. Otherwise, $a \longrightarrow b$ occurs in one of the configurations (a) to (d) of Definition 14 in $\mathcal{E}_{\mathcal{I}}(D)$. In configurations (a) to (c), the other arrows involved ($a \longleftarrow c$; $c \longrightarrow b$; or $a \longrightarrow c$ and $c \longrightarrow b$, resp.) are also present in G, hence $a \longrightarrow b$ is strongly $\mathcal{I}$-protected in G by the same configuration as in $\mathcal{E}_{\mathcal{I}}(D)$.
It remains to show that if $a \longrightarrow b$ is in configuration (d) in $\mathcal{E}_{\mathcal{I}}(D)$, it is also strongly $\mathcal{I}$-protected in G. In D, the vertices $\{a, c_1, c_2\}$ as defined in Definition 14 can occur in one of the following configurations:

$$c_1 \longleftarrow a \longleftarrow c_2, \quad c_1 \longleftarrow a \longrightarrow c_2, \quad c_1 \longrightarrow a \longrightarrow c_2.$$

The first and the third case are symmetric w.r.t. exchanging c_1 and c_2, hence we only consider the first two. Table 2 lists all possible configurations for the vertices $\{a, c_1, c_2\}$ in the graph G according to the condition $D \subset G \subset \mathcal{E}_{\mathcal{I}}(D)$. There is only one possibility for the arrow $a \longrightarrow b$ not to occur in one of the configurations (a) to (d) of Definition 14, and hence not being strongly $\mathcal{I}$-protected in G; however, the corresponding subgraph of $\{a, c_1, c_2\}$, $c_1 - a \longleftarrow c_2$, is forbidden by Definition 19.

(iii) According to Theorem 10, we have to check the following properties:

- D_1 and D_2 have the same skeleton, namely $D_1^u = D_2^u = G^u$.
- D_1 and D_2 have the same v-structures: let $a \longrightarrow b \longleftarrow c$ be a v-structure in D_1. This v-structure is then also present in $\mathcal{E}_{\mathcal{I}}(D_1)$. Because of $D_2 \subset G \subset \mathcal{E}_{\mathcal{I}}(D_1)$, we find it also in G and in D_2. The argument is completely symmetric w.r.t. exchanging D_1 and D_2.
- For all $I \in \mathcal{I}$, $D_1^{(I)}$ and $D_2^{(I)}$ have the same skeleton: assume, for the sake of contradiction, that there is some $I \in \mathcal{I}$ and an edge $a - b$ that is present in $(D_1^{(I)})^u$, but not in $(D_2^{(I)})^u$. W.l.o.g., we can assume that $a \longrightarrow b \in D_1$, $a \longleftarrow b \in D_2$, $a \in I$, $b \notin I$. Because of Theorem 18(iv), we then have $a \longrightarrow b \in \mathcal{E}_{\mathcal{I}}(D_1)$ and $a \longleftarrow b \in \mathcal{E}_{\mathcal{I}}(D_2)$; however, this is not compatible with the requirements $G \subset \mathcal{E}_{\mathcal{I}}(D_1)$ and $G \subset \mathcal{E}_{\mathcal{I}}(D_2)$. ■

Proof of Lemma 21 If $a \longrightarrow b \in \mathcal{E}_{\mathcal{I}}(D)$, it would be strongly $\mathcal{I}$-protected by Theorem 18(v), and hence also strongly $\mathcal{I}$-protected in G by Lemma 20(ii), contradicting the assumption. Therefore, $a - b \in \mathcal{E}_{\mathcal{I}}(D)$ and hence $D \subset G' \subset \mathcal{E}_{\mathcal{I}}(D)$.

Suppose that G' contains an induced subgraph of the form $c \longrightarrow d - e$. Since G does not contain such an induced subgraph, it must be of the form $c \longrightarrow a - b$ or $c \longrightarrow b - a$ in G'. In both cases, $a \longrightarrow b$ is then strongly $\mathcal{I}$-protected in G, either by configuration (a) or (b), a contradiction. ■

Proof of Lemma 22 Let $D \subset G \subset \mathcal{E}_{\mathcal{I}}(D)$ be a partial $\mathcal{I}$-essential graph that only has strongly $\mathcal{I}$-protected arrows. We can literally use the second part of the proof of Theorem 18 to show $G \supset \mathcal{E}_{\mathcal{I}}(D)$; there, we only used the fact that every arrow in G is strongly $\mathcal{I}$-protected. ■

B.3 Proofs for Section 4

Proof of Proposition 25 *"⇒":*

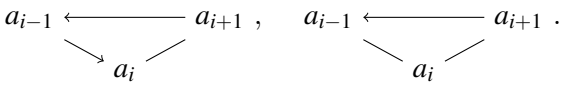
(i) This claim follows from Corollary 42.

Induced subgraph of $\{a, c_1, c_2\}$...		Configuration
... in D	... in G	of $a \longrightarrow b$ in G
$c_1 \longleftarrow a \longleftarrow c_2$	$c_1 \longleftarrow a \longleftarrow c_2$	(c)
	$c_1 - a \longleftarrow c_2$	—
	$c_1 \longleftarrow a - c_2$	(c)
	$c_1 - a - c_2$	(d)
$c_1 \longleftarrow a \longrightarrow c_2$	$c_1 \longleftarrow a \longrightarrow c_2$	(c)
	$c_1 - a \longrightarrow c_2$	(c)
	$c_1 \longleftarrow a - c_2$	(c)
	$c_1 - a - c_2$	(d)

Table 2: Possible configurations for the vertices $\{a, c_1, c_2\}$ in the proof of Lemma 20(ii). The labels in the last column refer to the configurations of Definition 14.

(ii) Suppose that there is some vertex $a \in N \setminus C$, that is a vertex $a \in N$ with $a \longleftarrow v \in D$. D' would have a directed cycle if $u \longleftarrow a \in D$, so $u \longrightarrow a \in D$. But then, $u \longrightarrow a \longleftarrow v$ is a v-structure in D, hence also in G, and consequently $a \notin \mathrm{ne}_G(v)$, a contradiction.

(iii) Assume that $\gamma = (v \equiv a_0, a_1, \ldots, a_k \equiv u)$ is a shortest path from v to u in G that does not intersect with C. We claim that γ is a directed path in D, which means that D' has a directed cycle, a contradiction.
Suppose that the claim is wrong, and let $a_i \longleftarrow a_{i+1} \in D$ be the first edge of (the chain) γ that points away from u in D; $i \geq 1$ holds by the assumption that, in particular, $a_1 \notin C$. $a_{i-1} \longrightarrow a_i \longleftarrow a_{i+1}$ cannot be an induced subgraph of D, otherwise it would also be present in G and hence γ would not be a *path* in G. Hence $a_{i-1} \cdots\cdots a_{i+1} \in G$; more precisely, $a_{i-1} \longleftarrow a_{i+1} \in G$ (and hence also in D), otherwise there would be a shorter path from v to u in G than γ that does not intersect with C. Because γ is a path in G, a_{i-1}, a_i and a_{i+1} can occur in G only in one of the following configurations:



However, both graphs cannot be an induced subgraph of the chain graph G.

"$\Leftarrow$": Since C is a clique in $G[T_G(v)]$, there is a DAG $D \in \mathbf{D}(G)$ with $\{a \in \mathrm{ne}_G(v) \mid a \longrightarrow v \in D\} = C$ by Proposition 44 and Corollary 42. It remains to show that D' is a DAG.

Assume, for the sake of contradiction, that D' has a directed cycle going through $u \longrightarrow v$. The return path from v to u, $\gamma = (v \equiv a_0, a_1, \ldots, a_k \equiv u)$, must come from a path in G and must therefore, by assumption, contain a vertex $a_i \in C$ ($i \geq 2$). Since $a_i \longrightarrow v \in D$ by construction, this means that D has a directed cycle $(a_0, a_1, \ldots, a_i, a_0)$, a contradiction.

Uniqueness of $\mathcal{E}_{\mathcal{I}}(D')$: Let $D_1, D_2 \in \mathbf{D}(G)$ with $\{a \in \mathrm{ne}_G(v) \mid a \longrightarrow v \in D_1\} = \{a \in \mathrm{ne}_G(v) \mid a \longrightarrow v \in D_2\} = C$, and set $D'_i := D_i + (u, v)$, $i = 1, 2$; we assume that $D'_1, D'_2 \in \mathbf{D}^+(G)$. To prove $D'_1 \sim_{\mathcal{I}} D'_2$, we have to check the following three points according to Theorem 10(iv):

- D'_1 and D'_2 obviously have the same skeleton.
- D'_1 and D'_2 have the same v-structures. We already know that D_1 and D_2 have the same v-structures. Let's assume, for the sake of contradiction, that (w.l.o.g.) D'_1 has a v-structure $u \longrightarrow v \longleftarrow a$ that D'_2 has not. In G, we must then have a line $a \multimap v$, hence $a \in \mathrm{ne}_G(v)$. However, the arrow between a and v would then have the same orientation in D_1 and D_2 by construction, a contradiction.
- For all $I \in \mathcal{I}$, $D_1^{\prime(I)}$ and $D_2^{\prime(I)}$ have the same skeleton. If this was not the case, there would be some vertices $a, b \in [p]$ and some $I \in \mathcal{I}$ such that $a \longrightarrow b \in D'_1$, $a \longleftarrow b \in D'_2$ and $|I \cap \{a, b\}| = 1$. The arrow $u \longrightarrow v$ is part of D'_1 and D'_2 by construction, so the arrows between a and b must be present in D_1 and D_2; however, $D_1^{(I)}$ and $D_2^{(I)}$ would then not have the same skeleton, a contradiction. ■

Corollary 26 is an immediate consequence of Proposition 25 and the fact that we assume the score function to be decomposable, so we skip the proof here.

Proof of Lemma 27 Obviously, we have $D' \subset H$. To show $H \subset \mathcal{E}_{\mathcal{I}}(D')$, we look at some edge $a - b \in G$ with $a, b \notin T_G(v)$ and show that $a - b \in \mathcal{E}_{\mathcal{I}}(D')$. W.l.o.g., we can assume that $a \longrightarrow b \in D$. By Corollary 41, there exists a $D_2 \in \mathbf{D}(G)$ that has the same orientation of edges in $T_G(v)$, but an orientation of edges in $T_G(a)$ such that $a \longleftarrow b \in D_2$. By Proposition 25, we know that $D_2' := D_2 + (u,v)$ is $\mathcal{I}$-equivalent to D', so in particular $a - b \in D' \cup D_2' \subset \mathcal{E}_{\mathcal{I}}(D')$.

It remains to show that $a \longrightarrow b - c$ does not occur as an induced subgraph of H. The inserted arrow $u \longrightarrow v$ cannot be part of such a subgraph, since all other edges incident to v are oriented in H by construction. Since G has no such subgraph either (Theorem 18), it could only appear in H through one of the newly oriented edges of $T_G(v)$. This means that if H had an induced subgraph of the form $a \longrightarrow b - c$, the corresponding vertices would be in configuration $a - b - c$ in G; however, $c \in T_G(v)$ then, and so the edge between b and c would be oriented in H, a contradiction. ■

Proof of Proposition 28 *"⇒":*

(i) By Corollary 42, $\{a \in \mathrm{ne}_G(v) \mid a \longrightarrow v \in D\}$ is a clique, hence every subset—in particular, C—is a clique, too.

(ii) Assume that there is some $a \in C \setminus \mathrm{ad}_G(u)$; then $u \in \mathrm{ne}_G(v)$, otherwise $u \longrightarrow v - a$ would be an induced subgraph of G. Nevertheless, $a \in C$ means that $u \longrightarrow v \longleftarrow a$ is a v-structure in D, which should hence also be present in G.

"⇐": We only must prove the existence of the claimed $D \in \mathbf{D}(G)$, see the comment in the beginning of Section 4.2. We distinguish two cases:

- $u \longrightarrow v \in G$. The existence of the DAG $D \in \mathbf{D}(G)$ with the requested properties follows from Corollary 42.

- $u - v \in G$, hence $u - a \in G$ for all $a \in N$ because G is a chain graph. Therefore, $C \cup \{u\}$ is a clique in $G[\mathrm{ne}_G(v)]$, and the existence of the claimed D again follows from Corollary 42.

Uniqueness of $\mathcal{E}_{\mathcal{I}}(D')$: Let $D_1, D_2 \in \mathbf{D}(G)$ with $u \longrightarrow v \in D_1, D_2$ and $\{a \in \mathrm{ne}_G(v) \setminus \{u\} \mid a \longrightarrow v \in D_1\} = \{a \in \mathrm{ne}_G(v) \setminus \{u\} \mid a \longrightarrow v \in D_2\} = C$, and set $D_i' := D_i - (u,v)$, $i = 1,2$. To prove $D_1' \sim_{\mathcal{I}} D_2'$, we have to check the following three points according to Theorem 10(iv):

- D_1' and D_2' have the same skeleton, namely $G^u - (u,v) - (v,u)$.

- D_1' and D_2' have the same v-structures. Otherwise, w.l.o.g., D_1' would have a v-structure $a \longrightarrow b \longleftarrow c$ that D_2' has not. D_1 and D_2 have the same v-structures, so $a \longrightarrow b \longleftarrow c$ is no induced subgraph of D_1; this implies $a = u$, $c = v$. Since D_2' does not have the v-structure $u \longrightarrow b \longleftarrow v$, the vertices u, b and v must occur in configuration $u \longrightarrow b \longrightarrow v$ or $u \longleftarrow b \longrightarrow v$ in D_2' (the configuration $u \longleftarrow b \longleftarrow v$ is not consistent with the acyclicity of D_2). However, all edges incident to v must have the same orientation in D_1' and D_2' by construction, a contradiction.

- Let $I \in \mathcal{I}$. Because of $(D_1^{(I)})^u = (D_2^{(I)})^u$ and $(D_i'^{(I)})^u = (D_i^{(I)})^u - (u,v) - (v,u)$ for $i = 1,2$, we have $(D_1'^{(I)})^u = (D_2'^{(I)})^u$. ■

Corollary 29 follows quickly from Proposition 28, and the proof of Lemma 30 is very similar to that of Lemma 27. Therefore we skip both proofs here and proceed with the proofs of Section 4.3.

Proof of Proposition 31 Note that we can write $N = \mathrm{ne}_G(v) \cap \mathrm{ad}_G(u) = \mathrm{ne}_G(v) \cap \mathrm{ne}_G(u)$ because $u - v \in G$ and G is a chain graph.

"$\Rightarrow$":

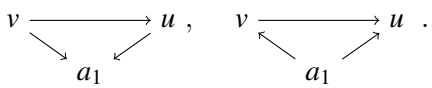
(i) This follows from Corollary 42.

(ii) D and D' have the same skeleton; the same is true for $D^{(I)}$ and $D'^{(I)}$ for all $I \in \mathcal{I}$. To see the latter, assume that for some $I \in \mathcal{I}$, the intervention graphs $D^{(I)}$ and $D'^{(I)}$ have a different skeleton. Since D and D' only differ in the orientation of the arrow between u and v, the skeletons of $D^{(I)}$ and $D'^{(I)}$ can only differ in that u and v are adjacent in one of them and not adjacent in the other one. However, this would imply that $|I \cap \{u,v\}| = 1$, and hence the edge between u and v would be directed in G by Theorem 18(iv), contradicting the assumption of the proposition. Finally, D' has at least all v-structures that D has by construction.
As a consequence $D' \not\sim_{\mathcal{I}} D$ if and only if D' has *more* v-structures than D (Theorem 10). An additional v-structure in D' must be of the form $u \longrightarrow v \longleftarrow a$. The edge between v and a cannot be directed in G, otherwise $u - v \longleftarrow a$ would be an induced subgraph of G, which is forbidden by Theorem 18(iii). Hence $a \in \mathrm{ne}_G(v)$, or, more precisely, $a \in C \setminus N$.

(iii) If $N \setminus C$ is empty, the statement is trivial. Otherwise, assume that there is some shortest path $\gamma = (a_0, a_1, \ldots, a_k)$ from $N \setminus C$ to $C \setminus N$ in $G[\mathrm{ne}_G(v)]$ that has no vertex in $C \cap N$.
By definition of C, $a_k \longrightarrow v \in D$; furthermore, $u \longrightarrow a_0 \in D$ must hold, otherwise (v, a_0, u, v) would be a directed cycle in D'. Therefore, γ must not be a path from a_0 to a_k in D. Let $a_i \longleftarrow a_{i+1}$ be the first arrow in γ that points away from a_k in D. If $i = 0$, $u \longrightarrow a_0 \longleftarrow a_1$ would be a v-structure in D since $a_1 \notin N$: by assumption, $a_1 \notin N \cap C$, and $a_1 \notin N \setminus C$ because of the minimality of γ. Hence $i > 0$ (and $k > 1$) must hold, and $a_{i-1} \cdots\cdots a_{i+1}$ in D and G, otherwise there would be a v-structure in D. However, γ is not the *shortest* path with the requested properties then, a contradiction.

"$\Leftarrow$": From Proposition 43, we see that there exists a DAG D that has the requested properties, and in which, in addition, $\{a \in \mathrm{ne}_G(u) \mid a \longrightarrow u \in D\} = (C \cap N) \cup \{v\}$ (point (iv) of Proposition 43). The fact that $D' := D - (v,u) + (u,v) \not\sim_{\mathcal{I}} D$ can be seen by an argument very similar to the proof of point (ii) above; it remains to show that D has no v-u-path except (v,u). Suppose that $\gamma = (a_0 \equiv v, a_1, \ldots, a_k \equiv u)$, $k \geq 2$, is such a path. In particular, γ is then also a v-u-path in G, hence γ lies completely in $T_G(v)$.

If $k = 2$, then $a_1 \in N$, and so the vertices u, v and a_1 occur in one of the following configurations in D by Proposition 43:



Both configurations contradict the assumption that $\gamma = (v, a_1, u)$ forms a path in D. Thus we conclude $k \geq 3$, and we notice $a_{k-1} \in \mathrm{ne}_G(u) \setminus \{v\}$. If $a_{k-1} \in C$, $a_{k-1} \longrightarrow v \in D$, hence $(a_0, a_1, \ldots, a_{k-1}, a_0)$ would be a cycle in D. On the other hand, if $a_{k-1} \notin C$, we would have $a_{k-1} \longleftarrow u \in D$, so γ would not be a path in D.

Uniqueness of $\mathcal{E}_{\mathcal{I}}(D')$: Let $D_1, D_2 \in \mathbf{D}(G)$ with $u \longleftarrow v \in D_1, D_2$ and $\{a \in \mathrm{ne}_G(v) \mid a \longrightarrow v \in D_1\} = \{a \in \mathrm{ne}_G(v) \mid a \longrightarrow v \in D_2\} = C$, and set $D_i' := D_i - (v,u) + (u,v)$, $i = 1,2$; we assume that $D_1', D_2' \in \mathbf{D}^{\circlearrowleft}(G)$. As in the proofs of Proposition 25, we can check that $D_1' \sim_{\mathcal{I}} D_2'$:

- D_1' and D_2' obviously have the same skeleton.
- D_1 and D_2 have the same v-structures. If this does not hold for D_1' and D_2', (w.l.o.g.) D_1' must have a v-structure $u \longrightarrow v \longleftarrow a$ that D_2' has not. Since $u \longrightarrow v \longleftarrow a$ cannot be an induced subgraph of G, $a \in \mathrm{ne}_G(v)$; however, the edges between v and its neighbors are oriented in the same way in D_1' and D_2' by construction, a contradiction.
- For all $I \in \mathcal{I}$, $D_1'^{(I)}$ and $D_2'^{(I)}$ have the same skeleton: this can be seen by an argument very similar to that in the proof of Proposition 25. ■

Proof of Corollary 32 We have to show $\mathrm{pa}_D(v) = \mathrm{pa}_G(v) \cup C$ and $\mathrm{pa}_D(u) = \mathrm{pa}_G(u) \cup (C \cap N) \cup \{v\}$. The first identity is immediately clear. For the second identity, note that for any vertex $a \in C \cap N$, the arrow between a and u must be oriented as $a \longrightarrow u \in D$ because the other orientation would induce a 3-cycle. On the other hand, we have $a \longleftarrow u \in D$ for $a \in N \setminus C$ because a different orientation would induce a 3-cycle in D'. Finally, we also have $a \longleftarrow u \in D$ for any $a \in \mathrm{ne}_G(u) \setminus (\mathrm{ne}_G(v) \cup \{v\})$ since the other orientation would induce a v-structure $v \longrightarrow u \longleftarrow a$ in D. ■

Lemma 33 can be proven very similarly as Lemma 27. Finally, we finish this proof section with the proof of Proposition 34 characterizing a step of the turning phase of GIES for the case that we turn an $\mathcal{I}$-essential arrow in some representative $D \in \mathbf{D}(G)$. We will omit the proof of Lemma 35 since it can be proven similarly to Lemma 27.

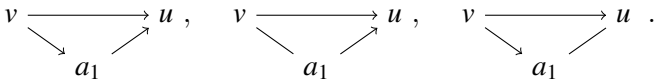
Proof of Proposition 34 When $v \longrightarrow u \in G$ (that is, u and v lie in different chain components), $N = \mathrm{ne}_G(v) \cap \mathrm{ad}_G(u) = \mathrm{ne}_G(v) \cap \mathrm{pa}_G(u)$ holds because G is a chain graph.

"$\Rightarrow$":

(i) This point follows from Corollary 42.

(ii) If this was not true, D' would have a cycle of the form (u,v,a,u) for some $a \in N$ since $N \subset \mathrm{pa}_G(u)$.

(iii) Suppose that the path $\gamma = (a_0 \equiv v, a_1, \ldots, a_k \equiv u)$ is a shortest counterexample of a path without vertex in $C \cup \mathrm{ne}_G(u)$.
Assume that $k = 2$. Since u and v lie in different chain components, the vertices u, v and a_1 can occur in one of the following configurations in G:



The first case implies the existence of a directed cycle in D'; in the second case, $a_1 \in N \subset C$, in the third case, $a_1 \in \mathrm{ne}_G(u)$.
Therefore $k \geq 3$. In complete analogy to the proof of Proposition 25, we can show that γ is also a v-u-path in D, hence D' has a directed cycle, a contradiction.

"⇐": Let $D \in \mathbf{D}(G)$ be a DAG with $\{a \in \mathrm{ne}_G(v) \mid a \longrightarrow v \in D\} = C$ and in which all edges of $D[T_G(u)]$ point away from u; such a DAG exists by Corollary 42 and meets the requirements of Proposition 34. It remains to show that D' is acyclic, that means that D has no v-u-path except (v,u).

Suppose, for the sake of contradiction, that D has such a path $\gamma = (a_0 \equiv v, a_1, \ldots, a_k \equiv u)$. γ is then also a v-u-path in G, hence there is, by assumption, some $a_i \in C \cup P$. If $a_i \in C$, $(a_0, a_1, \ldots, a_i, a_0)$ would be a cycle in D; on the other hand, if $a_i \in P$, $(a_i, a_{i+1}, \ldots, a_k, a_i)$ would be a cycle in D, a contradiction.

Uniqueness of $\mathcal{E}_{\mathcal{I}}(D')$: The proof given for Proposition 31 is also valid here. ■

Proof of Corollary 36 The fact that $\mathrm{pa}_D(v) = \mathrm{pa}_G(v) \cup C$ is clear from Proposition 34; it remains to show that $\mathrm{pa}_D(u) = \mathrm{pa}_G(u)$. Any neighbor a of u must also be a child of v, otherwise G would have a subgraph of the form $v \longrightarrow u - a$, which is forbidden by Theorem 18(iii). Hence $a \longleftarrow u \in D$ for all $a \in \mathrm{ne}_G(u)$ since the other orientation would imply a directed cycle in D'. ■

References

S. A. Andersson, D. Madigan, and M. D. Perlman. A characterization of Markov equivalence classes for acyclic digraphs. *Annals of Statistics*, 25(2):505–541, 1997.

L. E. Brown, I. Tsamardinos, and C. F. Aliferis. A comparison of novel and state-of-the-art polynomial bayesian network learning algorithms. In *Proceedings of the Twentieth National Conference on Artificial Intelligence (AAAI)*, volume 20, page 739, 2005.

R. Castelo and T. Kočka. On inclusion-driven learning of Bayesian networks. *Journal of Machine Learning Research*, 4:527–574, 2003.

D. M. Chickering. Learning equivalence classes of Bayesian-network structures. *Journal of Machine Learning Research*, 2(3):445–498, 2002a.

D. M. Chickering. Optimal structure identification with greedy search. *Journal of Machine Learning Research*, 3(3):507–554, 2002b.

G. F. Cooper and C. Yoo. Causal discovery from a mixture of experimental and observational data. In *Uncertainty in Artificial Intelligence*, pages 116–125, 1999.

D. G. Corneil. Lexicographic breadth first search—a survey. In *Graph-Theoretic Concepts in Computer Science*, volume 3353 of *Lecture Notes in Computational Science*, pages 1–19. Springer, Berlin, 2004.

D. Eaton and K. Murphy. Exact Bayesian structure learning from uncertain interventions. In *Artificial Intelligence and Statistics*, volume 2, pages 107–114, 2007.

F. Eberhardt. Almost optimal intervention sets for causal discovery. In *Uncertainty in Artificial Intelligence*, pages 161–168, 2008.

F. Eberhardt, C. Glymour, and R. Scheines. On the number of experiments sufficient and in the worst case necessary to identify all causal relations among N variables. In *Uncertainty in Artificial Intelligence*, pages 178–184, 2005.

F. Eberhardt, P. O. Hoyer, and R. Scheines. Combining experiments to discover linear cyclic models with latent variables. In *Artificial Intelligence and Statistics*, pages 185–192, 2010.

A. Hauser and P. Bühlmann. Jointly interventional and observational data: estimation of corresponding Markov equivalence classes of directed acyclic graphs. Work in progress, 2012.

Y. He and Z. Geng. Active Learning of Causal Networks with Intervention Experiments and Optimal Designs. *Journal of Machine Learning Research*, 9:2523–2547, 2008.

M. Kalisch and P. Bühlmann. Estimating high-dimensional directed acyclic graphs with the PC-algorithm. *Journal of Machine Learning Research*, 8:636, 2007.

M. Kalisch, M. Mächler, D. Colombo, M. H. Maathuis, and P. Bühlmann. Causal inference using graphical models with the R package pcalg. *Journal of Statistical Software*, 47(11):1–26, 2012.

K. B. Korb, L. R. Hope, A. E. Nicholson, and K. Axnick. Varieties of causal intervention. In *Pacific Rim International Conference on Artificial Intelligence*, pages 322–331, New York, 2004. Springer.

S. L. Lauritzen. *Graphical Models*. Oxford University Press, USA, 1996.

C. Marbach, R. J. Prill, T. Schaffter, C. Mattiussi, D. Floreano, and G. Stolovitzky. Revealing strengths and weaknesses of methods for gene network inference. *Proceedings of the National Academy of Sciences*, 107(14):6286–6291, 2010.

D. Marbach, T. Schaffter, C. Mattiussi, and D. Floreano. Generating realistic in silico gene networks for performance assessment of reverse engineering methods. *Journal of Computational Biology*, 16(2):229–239, 2009.

C. Meek. *Graphical models: selecting causal and statistical models*. PhD thesis, Philosophy Dept. Carnegie Mellon University, Pittsburgh, PA, 1997.

K. Murphy. The Bayes net toolbox for Matlab. *Computing science and statistics*, 33(2):1024–1034, 2001.

J. Pearl. Causal diagrams for empirical research. *Biometrika*, 82(4):669–688, 1995.

R. J. Prill, D. Marbach, J. Saez-Rodriguez, P. K. Sorger, L. G. Alexopoulos, X. Xue, N. D. Clarke, G. Altan-Bonnet, and G. Stolovitzky. Towards a rigorous assessment of systems biology models: the DREAM3 challenges. *PLoS ONE*, 5(2):e9202, 02 2010.

R. W. Robinson. Counting labeled acyclic digraphs. In F. Harary, editor, *New Directions in the Theory of Graphs*, pages 239–273, New York, 1973. Academic Press.

D. J. Rose. Triangulated graphs and the elimination process. *Journal of Mathematical Analysis and Applications*, 32(3):597–609, 1970.

D. J. Rose, R. E. Tarjan, and G. S. Lueker. Algorithmic aspects of vertex elimination on graphs. *SIAM Journal on Computing*, 5(2):266–283, 1976.

G. Schwarz. Estimating the dimension of a model. *Annals of Statistics*, 6(2):461–464, 1978.

T. Silander and P. Myllymäki. A simple approach for finding the globally optimal Bayesian network structure. In *Uncertainty in Artificial Intelligence*, San Francisco, 2006.

P. Spirtes, C. N. Glymour, and R. Scheines. *Causation, Prediction, and Search*. MIT Press, 2000.

E. Szpilrajn. Sur l'extension de l'ordre partiel. *Fundamenta Mathematicae*, 16:386–389, 1930.

J. Tian and J. Pearl. Causal discovery from changes. In *Uncertainty in Artificial Intelligence*, pages 512–521, 2001.

S. Tong and D. Koller. Active learning for structure in Bayesian networks. In *International Joint Conference on Artificial Intelligence*, volume 17, pages 863–869, 2001.

I. Tsamardinos, L. E. Brown, and C. F. Aliferis. The max-min hill-climbing Bayesian network structure learning algorithm. *Machine Learning*, 65(1):31–78, 2006.

T.S. Verma and J. Pearl. Equivalence and synthesis of causal models. In *Uncertainty in Artificial Intelligence*, page 270, 1990.

S. Wright. Correlation and causation. *Journal of Agricultural Research*, 20(7):557–585, 1921.

Journal of Machine Learning Research 13 (2012) 2465-2501 Submitted 3/11; Revised 2/12; Published 8/12

On the Convergence Rate of ℓ_p-Norm Multiple Kernel Learning*

Marius Kloft† KLOFT@TU-BERLIN.DE
Machine Learning Laboratory
Technische Universität Berlin
Franklinstr. 28/29
10587 Berlin, Germany

Gilles Blanchard GILLES.BLANCHARD@MATH.UNI-POTSDAM.DE
Department of Mathematics
University of Potsdam
Am Neuen Palais 10
14469 Potsdam, Germany

Editor: Sören Sonnenburg, Francis Bach, and Cheng Soog Ong

Abstract

We derive an upper bound on the local Rademacher complexity of ℓ_p-norm multiple kernel learning, which yields a tighter excess risk bound than global approaches. Previous local approaches analyzed the case $p = 1$ only while our analysis covers all cases $1 \leq p \leq \infty$, assuming the different feature mappings corresponding to the different kernels to be uncorrelated. We also show a lower bound that shows that the bound is tight, and derive consequences regarding excess loss, namely fast convergence rates of the order $O(n^{-\frac{\alpha}{1+\alpha}})$, where α is the minimum eigenvalue decay rate of the individual kernels.

Keywords: multiple kernel learning, learning kernels, generalization bounds, local Rademacher complexity

1. Introduction

Propelled by the increasing "industrialization" of modern application domains such as bioinformatics or computer vision leading to the accumulation of vast amounts of data, the past decade experienced a rapid professionalization of machine learning methods. Sophisticated machine learning solutions such as the support vector machine can nowadays almost completely be applied out-of-the-box (Bouckaert et al., 2010). Nevertheless, a displeasing stumbling block towards the complete automatization of machine learning remains that of finding the best abstraction or *kernel* (Schölkopf et al., 1998; Müller et al., 2001) for a problem at hand.

In the current state of research, there is little hope that in the near future a machine will be able to automatically engineer the *perfect* kernel for a particular problem (Searle, 1980). However, by restricting to a less general problem, namely to a finite set of base kernels the algorithm can pick

∗. This is a longer version of a short conference paper entitled *The Local Rademacher Complexity of ℓ_p-Norm MKL*, which is appearing in Advances in Neural Information Processing Systems 24 edited by J. Shawe-Taylor and R.S. Zemel and P. Bartlett and F. Pereira and K.Q. Weinberger (2011).

†. Parts of the work were done while MK was at Learning Theory Group, Computer Science Division and Department of Statistics, University of California, Berkeley, CA 94720-1758, USA.

from, one might hope to achieve automatic kernel selection: clearly, cross-validation based model selection (Stone, 1974) can be applied if the number of base kernels is decent. Still, the performance of such an algorithm is limited by the performance of the best kernel in the set.

In the seminal work of Lanckriet et al. (2004) it was shown that it is computationally feasible to simultaneously learn a support vector machine *and* a linear combination of kernels at the same time, if we require the so-formed kernel combinations to be positive definite and trace-norm normalized. Though feasible for small sample sizes, the computational burden of this so-called *multiple kernel learning* (MKL) approach is still high. By further restricting the multi-kernel class to only contain convex combinations of kernels, the efficiency can be considerably improved, so that ten thousands of training points and thousands of kernels can be processed (Sonnenburg et al., 2006).

However, these computational advances come at a price. Empirical evidence has accumulated showing that sparse-MKL optimized kernel combinations rarely help in practice and frequently are to be outperformed by a regular SVM using an unweighted-sum kernel $K = \sum_m K_m$ (Cortes et al., 2008; Gehler and Nowozin, 2009), leading for instance to the provocative question "Can learning kernels help performance?" (Cortes, 2009).

A first step towards a model of learning the kernel that is useful in practice was achieved in Kloft et al. (2008), Cortes et al. (2009), Kloft et al. (2009) and Kloft et al. (2011), where an ℓ_q-norm, $q \geq 1$, rather than an ℓ_1 penalty was imposed on the kernel combination coefficients. The ℓ_q-norm MKL is an empirical minimization algorithm that operates on the multi-kernel class consisting of functions $f : x \mapsto \langle \boldsymbol{w}, \phi_k(x) \rangle$ with $\|\boldsymbol{w}\|_k \leq D$, where ϕ_k is the kernel mapping into the reproducing kernel Hilbert space (RKHS) $\mathcal{H}_k$ with kernel k and norm $\|.\|_k$, while the kernel k itself ranges over the set of possible kernels $\left\{ k = \sum_{m=1}^M \theta_m k_m \;\middle|\; \|\boldsymbol{\theta}\|_q \leq 1,\ \boldsymbol{\theta} \geq 0 \right\}$.

In Figure 1, we reproduce exemplary results taken from Kloft et al. (2009, 2011) (see also references therein for further evidence pointing in the same direction). We first observe that, as expected, ℓ_q-norm MKL enforces strong sparsity in the coefficients θ_m when $q = 1$, and no sparsity at all for $q = \infty$, which corresponds to the SVM with an unweighted-sum kernel, while intermediate values of q enforce different degrees of soft sparsity (understood as the steepness of the decrease of the ordered coefficients θ_m). Crucially, the performance (as measured by the AUC criterion) is not monotonic as a function of q; $q = 1$ (sparse MKL) yields significantly worse performance than $q = \infty$ (regular SVM with sum kernel), but optimal performance is attained for some intermediate value of q. This is an empirical strong motivation to theoretically study the performance of ℓ_q-MKL beyond the limiting cases $q = 1$ or $q = \infty$.

A conceptual milestone going back to the work of Bach et al. (2004) and Micchelli and Pontil (2005) is that the above multi-kernel class can equivalently be represented as a block-norm regularized linear class in the product Hilbert space $\mathcal{H} := \mathcal{H}_1 \times \cdots \times \mathcal{H}_M$, where $\mathcal{H}_m$ denotes the RKHS associated to kernel k_m, $1 \leq m \leq M$. More precisely, denoting by ϕ_m the kernel feature mapping associated to kernel k_m over input space $\mathcal{X}$, and $\phi : x \in \mathcal{X} \mapsto (\phi_1(x), \ldots, \phi_M(x)) \in \mathcal{H}$, the class of functions defined above coincides with

$$H_{p,D,M} = \left\{ f_{\boldsymbol{w}} : x \mapsto \langle \boldsymbol{w}, \phi(x) \rangle \;\middle|\; \boldsymbol{w} = (\boldsymbol{w}^{(1)}, \ldots, \boldsymbol{w}^{(M)}), \|\boldsymbol{w}\|_{2,p} \leq D \right\}, \tag{1}$$

where there is a one-to-one mapping of $q \in [1, \infty]$ to $p \in [1,2]$ given by $p = \frac{2q}{q+1}$ (see Appendix A for a derivation). The $\ell_{2,p}$-norm is defined here as $\left\|\boldsymbol{w}\right\|_{2,p} := \left\| \left(\|\boldsymbol{w}^{(1)}\|_{k_1}, \ldots, \|\boldsymbol{w}^{(M)}\|_{k_M} \right) \right\|_p = \left(\sum_{m=1}^M \left\|\boldsymbol{w}^{(m)}\right\|_{k_m}^p \right)^{1/p}$; for simplicity, we will frequently write $\left\|\boldsymbol{w}^{(m)}\right\|_2 = \left\|\boldsymbol{w}^{(m)}\right\|_{k_m}$.

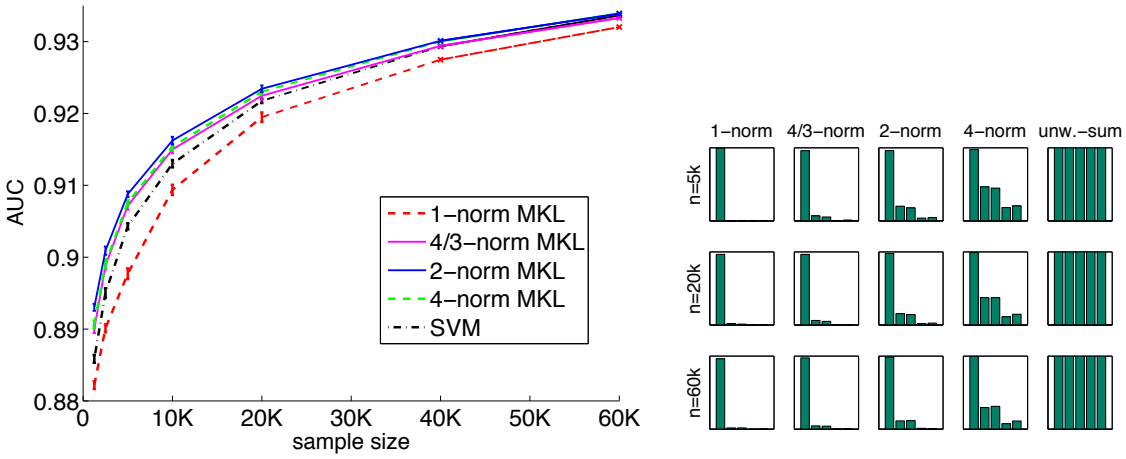


Figure 1: Splice site detection experiment in Kloft et al. (2009, 2011). LEFT: The Area under ROC curve as a function of the training set size is shown. The regular SVM is equivalent to $q = \infty$ (or $p = 2$). RIGHT: The optimal kernel weights θ_m as output by ℓ_q-norm MKL are shown.

Clearly, the complexity of the class (1) will be greater than one that is based on a single kernel only. However, it is unclear whether the increase is decent or considerably high and—since there is a free parameter p—how this relates to the choice of p. To this end the main aim of this paper is to analyze the sample complexity of the above hypothesis class (1). An analysis of this model, based on global Rademacher complexities, was developed by Cortes et al. (2010). In the present work, we base our main analysis on the theory of *local* Rademacher complexities, which allows to derive improved and more precise rates of convergence.

1.1 Outline of the Contributions

This paper makes the following contributions:

- Upper bounds on the local Rademacher complexity of ℓ_p-norm MKL are shown, from which we derive an excess risk bound that achieves a fast convergence rate of the order $O(M^{1+\frac{2}{1+\alpha}\left(\frac{1}{p^*}-1\right)} n^{-\frac{\alpha}{1+\alpha}})$, where α is the minimum eigenvalue decay rate of the individual kernels[1] (previous bounds for ℓ_p-norm MKL only achieved $O(M^{\frac{1}{p^*}} n^{-\frac{1}{2}})$.
- A lower bound is shown that besides absolute constants matches the upper bounds, showing that our results are tight.
- The generalization performance of ℓ_p-norm MKL as guaranteed by the excess risk bound is studied for varying values of p, shedding light on the appropriateness of a small/large p in various learning scenarios.

1. That is, it $\exists d > 0$ and $\alpha > 1$ such that for all $m = 1, \ldots, M$ it holds $\lambda_j^{(m)} \leq d j^{-\alpha}$, where $\lambda_j^{(m)}$ is the jth eigenvalue of the mth kernel (sorted in descending order).

Furthermore, we also present a simple proof of a global Rademacher bound similar to the one shown in Cortes et al. (2010). A comparison of the rates obtained with local and global Rademacher analysis, respectively, can be found in Section 6.1.

1.2 Notation

For notational simplicity we will omit feature maps and directly view $\phi(x)$ and $\phi_m(x)$ as random variables $\boldsymbol{x}$ and $\boldsymbol{x}^{(m)}$ taking values in the Hilbert space $\mathcal{H}$ and $\mathcal{H}_m$, respectively, where $\boldsymbol{x} = (\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(M)})$. Correspondingly, the hypothesis class we are interested in reads $H_{p,D,M} = \left\{ f_{\boldsymbol{w}} : \boldsymbol{x} \mapsto \langle \boldsymbol{w}, \boldsymbol{x} \rangle \;\middle|\; \|\boldsymbol{w}\|_{2,p} \leq D \right\}$. If D or M are clear from the context, we sometimes synonymously denote $H_p = H_{p,D} = H_{p,D,M}$. We will frequently use the notation $(\boldsymbol{u}^{(m)})_{m=1}^M$ for the element $\boldsymbol{u} = (\boldsymbol{u}^{(1)}, \ldots, \boldsymbol{u}^{(M)}) \in \mathcal{H} = \mathcal{H}_1 \times \ldots \times \mathcal{H}_M$.

We denote the kernel matrices corresponding to k and k_m by K and K_m, respectively. Note that we are considering normalized kernel Gram matrices, that is, the ijth entry of K is $\frac{1}{n} k(\boldsymbol{x}_i, \boldsymbol{x}_j)$. We will also work with covariance operators in Hilbert spaces. In a finite dimensional vector space, the (uncentered) covariance operator can be defined in usual vector/matrix notation as $\mathbb{E}\boldsymbol{x}\boldsymbol{x}^\top$. Since we are working with potentially infinite-dimensional vector spaces, we will use instead of $\boldsymbol{x}\boldsymbol{x}^\top$ the tensor notation $\boldsymbol{x} \otimes \boldsymbol{x} \in \mathrm{HS}(\mathcal{H})$, which is a Hilbert-Schmidt operator $\mathcal{H} \mapsto \mathcal{H}$ defined as $(\boldsymbol{x} \otimes \boldsymbol{x})\boldsymbol{u} = \langle \boldsymbol{x}, \boldsymbol{u} \rangle \boldsymbol{x}$. The space $\mathrm{HS}(\mathcal{H})$ of Hilbert-Schmidt operators on $\mathcal{H}$ is itself a Hilbert space, and the expectation $\mathbb{E}\boldsymbol{x} \otimes \boldsymbol{x}$ is well-defined and belongs to $\mathrm{HS}(\mathcal{H})$ as soon as $\mathbb{E}\|\boldsymbol{x}\|^2$ is finite, which will always be assumed (as a matter of fact, we will often assume that $\|\boldsymbol{x}\|$ is bounded a.s.). We denote by $J = \mathbb{E}\boldsymbol{x} \otimes \boldsymbol{x}$, $J_m = \mathbb{E}\boldsymbol{x}^{(m)} \otimes \boldsymbol{x}^{(m)}$ the uncentered covariance operators corresponding to variables $\boldsymbol{x}$, $\boldsymbol{x}^{(m)}$; it holds that $\mathrm{tr}(J) = \mathbb{E}\|\boldsymbol{x}\|_2^2$ and $\mathrm{tr}(J_m) = \mathbb{E}\left\|\boldsymbol{x}^{(m)}\right\|_2^2$.

Finally, for $p \in [1, \infty]$ we use the standard notation p^* to denote the conjugate of p, that is, $p^* \in [1, \infty]$ and $\frac{1}{p} + \frac{1}{p^*} = 1$.

2. Global Rademacher Complexities in Multiple Kernel Learning

We first review global Rademacher complexities (GRC) in MKL. Let $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n$ be an i.i.d. sample drawn from P. The global Rademacher complexity is defined as

$$R(H_p) = \mathbb{E} \sup_{f_{\boldsymbol{w}} \in H_p} \langle \boldsymbol{w}, \frac{1}{n} \sum_{i=1}^n \sigma_i \boldsymbol{x}_i \rangle \tag{2}$$

where $(\sigma_i)_{1 \leq i \leq n}$ is an i.i.d. family (independent of $(\boldsymbol{x}_i)$) of Rademacher variables (random signs). Its empirical counterpart is denoted by $\widehat{R}(H_p) = \mathbb{E}\left[R(H_p) \middle| \boldsymbol{x}_1, \ldots, \boldsymbol{x}_n\right] = \mathbb{E}_{\boldsymbol{\sigma}} \sup_{f_{\boldsymbol{w}} \in H_p} \langle \boldsymbol{w}, \frac{1}{n} \sum_{i=1}^n \sigma_i \boldsymbol{x}_i \rangle$. The interest in the global Rademacher complexity comes from that if known it can be used to bound the generalization error (Koltchinskii, 2001; Bartlett and Mendelson, 2002).

In the recent paper of Cortes et al. (2010) it was shown using a combinatorial argument that the empirical version of the global Rademacher complexity can be bounded as

$$\widehat{R}(H_p) \leq D \sqrt{\frac{cp^*}{2n} \left\| \left(\mathrm{tr}(K_m) \right)_{m=1}^M \right\|_{\frac{p^*}{2}}},$$

where $c = \frac{23}{22}$ and $\mathrm{tr}(K)$ denotes the trace of the kernel matrix K.

We will now show a quite short proof of this result, extending it to the whole range $p \in [1, \infty]$, but at the expense of a slightly worse constant, and then present a novel bound on the population version of the GRC. The proof presented here is based on the Khintchine-Kahane inequality (Kahane, 1985) using the constants taken from Lemma 3.3.1 and Proposition 3.4.1 in Kwapién and Woyczyński (1992).

Lemma 1 (Khintchine-Kahane inequality)**.** *Let be $\boldsymbol{v}_1, \ldots, \boldsymbol{v}_M \in \mathcal{H}$. Then, for any $q \geq 1$, it holds*

$$\mathbb{E}_{\boldsymbol{\sigma}} \Big\| \sum_{i=1}^{n} \sigma_i \boldsymbol{v}_i \Big\|_2^q \leq \Big(c \sum_{i=1}^{n} \big\| \boldsymbol{v}_i \big\|_2^2 \Big)^{\frac{q}{2}},$$

where $c = \max(1, q^* - 1)$. In particular the result holds for $c = q^*$.

Proposition 2 (Global Rademacher complexity, empirical version)**.** *For any $p \geq 1$ the empirical version of global Rademacher complexity of the multi-kernel class H_p can be bounded as*

$$\forall t \geq p: \quad \widehat{R}(H_p) \leq D \sqrt{\frac{t^*}{n} \Big\| \big(\operatorname{tr}(K_m) \big)_{m=1}^{M} \Big\|_{\frac{t^*}{2}}}.$$

Proof First note that it suffices to prove the result for $t = p$ as trivially $\|\boldsymbol{x}\|_{2,t} \leq \|\boldsymbol{x}\|_{2,p}$ holds for all $t \geq p$ and therefore $R(H_p) \leq R(H_t)$.

We can use a block-structured version of Hölder's inequality (cf. Lemma 15) and the Khintchine-Kahane (K.-K.) inequality (cf. Lemma 1) to bound the empirical version of the global Rademacher complexity as follows:

$$\begin{aligned}
\widehat{R}(H_p) &\overset{\text{def.}}{=} \mathbb{E}_{\boldsymbol{\sigma}} \sup_{f_{\boldsymbol{w}} \in H_p} \langle \boldsymbol{w}, \frac{1}{n} \sum_{i=1}^{n} \sigma_i \boldsymbol{x}_i \rangle \\
&\overset{\text{Hölder}}{\leq} D \, \mathbb{E}_{\boldsymbol{\sigma}} \Big\| \frac{1}{n} \sum_{i=1}^{n} \sigma_i \boldsymbol{x}_i \Big\|_{2,p^*} \\
&\overset{\text{Jensen}}{\leq} D \Big(\mathbb{E}_{\boldsymbol{\sigma}} \sum_{m=1}^{M} \Big\| \frac{1}{n} \sum_{i=1}^{n} \sigma_i \boldsymbol{x}_i^{(m)} \Big\|_2^{p^*} \Big)^{\frac{1}{p^*}} \\
&\overset{\text{K.-K.}}{\leq} D \sqrt{\frac{p^*}{n}} \Big(\sum_{m=1}^{M} \Big(\underbrace{\frac{1}{n} \sum_{i=1}^{n} \big\| \boldsymbol{x}_i^{(m)} \big\|_2^2}_{= \operatorname{tr}(K_m)} \Big)^{\frac{p^*}{2}} \Big)^{\frac{1}{p^*}} \\
&= D \sqrt{\frac{p^*}{n} \Big\| \big(\operatorname{tr}(K_m) \big)_{m=1}^{M} \Big\|_{\frac{p^*}{2}}},
\end{aligned}$$

what was to show. ■

Note that there is a very good reason to state the above bound in terms of $t \geq p$ instead of solely in terms of p: the Rademacher complexity $\widehat{R}(H_p)$ is not monotonic in p and thus it is not always the best choice to take $t := p$ in the above bound. This can be readily seen, for example, for the easy case where all kernels have the same trace—in that case the bound translates into $\widehat{R}(H_p) \leq D \sqrt{t^* M^{\frac{2}{t^*}} \frac{\operatorname{tr}(K_1)}{n}}$. Interestingly, the function $x \mapsto x M^{2/x}$ is not monotone and attains its minimum for

$x = 2\log M$, where log denotes the natural logarithm with respect to the base e. This has interesting consequences: for any $p \le (2\log M)^*$ we can take the bound $\widehat{R}(H_p) \le D\sqrt{\frac{e\log(M)\operatorname{tr}(K_1)}{n}}$, which has only a mild dependency on the number of kernels; note that in particular we can take this bound for the ℓ_1-norm class $\widehat{R}(H_1)$ for all $M > 1$.

The above proof is very simple. However, computing the population version of the global Rademacher complexity of MKL is somewhat more involved and to the best of our knowledge has not been addressed yet by the literature. To this end, note that from the previous proof we obtain $R(H_p) \le \mathbb{E}\, D\sqrt{p^*/n}\big(\sum_{m=1}^M \big(\frac{1}{n}\sum_{i=1}^n \big\|\boldsymbol{x}_i^{(m)}\big\|_2^2\big)^{\frac{p^*}{2}}\big)^{\frac{1}{p^*}}$. We thus can use Jensen's inequality to move the expectation operator inside the root,

$$R(H_p) \le D\sqrt{p^*/n}\Big(\sum_{m=1}^M \mathbb{E}\big(\frac{1}{n}\sum_{i=1}^n \big\|\boldsymbol{x}_i^{(m)}\big\|_2^2\big)^{\frac{p^*}{2}}\Big)^{\frac{1}{p^*}}, \tag{3}$$

but now need a handle on the $\frac{p^*}{2}$-th moments. To this aim we use the inequalities of Rosenthal (1970) and Young (e.g., Steele, 2004) to show the following Lemma.

Lemma 3 (Rosenthal + Young)**.** *Let $X_1, \ldots, X_n$ be independent nonnegative random variables satisfying $\forall i : X_i \le B < \infty$ almost surely. Then, denoting $C_q = (2qe)^q$, for any $q \ge \frac{1}{2}$ it holds*

$$\mathbb{E}\left(\frac{1}{n}\sum_{i=1}^n X_i\right)^q \le C_q\left(\left(\frac{B}{n}\right)^q + \left(\frac{1}{n}\sum_{i=1}^n \mathbb{E}X_i\right)^q\right).$$

The proof is defered to Appendix B. It is now easy to show:

Corollary 4 (Global Rademacher complexity, population version)**.** *Assume the kernels are uniformly bounded, that is, $\|k\|_\infty \le B < \infty$, almost surely. Then for any $p \ge 1$ the population version of global Rademacher complexity of the multi-kernel class H_p can be bounded as*

$$\forall t \ge p: \quad R(H_{p,D,M}) \le D\, t^* \sqrt{\frac{e}{n}\Big\|\big(\operatorname{tr}(J_m)\big)_{m=1}^M\Big\|_{\frac{t^*}{2}}} + \frac{\sqrt{Be}DM^{\frac{1}{t^*}}t^*}{n}.$$

For $t \ge 2$ the right-hand term can be discarded and the result also holds for unbounded kernels.

Proof As above in the previous proof it suffices to prove the result for $t = p$. From (3) we conclude by the previous Lemma

$$\begin{aligned} R(H_p) &\le D\sqrt{\frac{p^*}{n}}\left(\sum_{m=1}^M (ep^*)^{\frac{p^*}{2}}\left(\left(\frac{B}{n}\right)^{\frac{p^*}{2}} + \Big(\underbrace{\mathbb{E}\frac{1}{n}\sum_{i=1}^n \big\|\boldsymbol{x}_i^{(m)}\big\|_2^2}_{=\operatorname{tr}(J_m)}\Big)^{\frac{p^*}{2}}\right)\right)^{\frac{1}{p^*}} \\ &\le Dp^*\sqrt{\frac{e}{n}\Big\|\big(\operatorname{tr}(J_m)\big)_{m=1}^M\Big\|_{\frac{p^*}{2}}} + \frac{\sqrt{Be}DM^{\frac{1}{p^*}}p^*}{n}, \end{aligned}$$

where for the last inequality we use the subadditivity of the root function. Note that for $p \ge 2$ it is $p^*/2 \le 1$ and thus it suffices to employ Jensen's inequality instead of the previous lemma so that we come along without the last term on the right-hand side. ■

For example, when the traces of the kernels are bounded, the above bound is essentially determined by $O\Big(\frac{p^* M^{\frac{1}{p^*}}}{\sqrt{n}}\Big)$. We can also remark that by setting $t = (\log(M))^*$ we obtain the bound $R(H_1) = O\Big(\frac{\log M}{\sqrt{n}}\Big)$.

2.1 Relation to Other Work

As discussed by Cortes et al. (2010), the above results lead to a generalization bound that improves on a previous result based on covering numbers by Srebro and Ben-David (2006). Another recently proposed approach to theoretically study MKL uses the Rademacher chaos complexity (RCC) (Ying and Campbell, 2009). The RCC is actually itself an upper bound on the usual Rademacher complexity. In their discussion, Cortes et al. (2010) observe that in the case $p = 1$ (traditional MKL), the bound of Proposition 2 grows logarithmically in the number of kernels M, and claim that the RCC approach would lead to a bound which is multiplicative in M. However, a closer look at the work of Ying and Campbell (2009) shows that this is not correct; in fact the RCC also leads to a logarithmic dependence in M when $p = 1$. This is because the RCC of a kernel class is the same as the RCC of its convex hull, and the RCC of the base class containing only the M individual kernels is logarithmic in M. This convex hull argument, however, only works for $p = 1$; we are unaware of any existing work trying to estimate the RCC or comparing it to the above approach in the case $p > 1$.

3. The Local Rademacher Complexity of Multiple Kernel Learning

We first give a gentle introduction to local Rademacher complexities in general and then present the main result of this paper: a lower and an upper bound on the local Rademacher complexity of ℓ_p-norm multiple kernel learning.

3.1 Local Rademacher Complexities in a Nutshell

Let $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n$ be an i.i.d. sample drawn from P; denote by $\mathbb{E}$ the expectation operator corresponding to P; let $\mathcal{F}$ be a class of functions mapping $\boldsymbol{x}_i$ to $\mathbb{R}$. Then the local Rademacher complexity is defined as

$$R_r(\mathcal{F}) = \mathbb{E} \sup_{f \in \mathcal{F} : Pf^2 \leq r} \frac{1}{n} \sum_{i=1}^{n} \sigma_i f(\boldsymbol{x}_i)\,, \tag{4}$$

where $Pf^2 := \mathbb{E}(f(\boldsymbol{x}))^2$. In a nutshell, when comparing the global and local Rademacher complexities, that is, (2) and (4), we observe that the local one involves the additional constraint $Pf^2 \leq r$ on the (uncentered) "variance" of functions. It allows us to sort the functions according to their variances and discard the ones with suboptimal high variance. We can do so by, instead of McDiarmid's inequality, using more powerful concentration inequalities such as Talagrand's inequality (Talagrand, 1995). Roughly speaking, the local Rademacher complexity allows us to consider the problem at various scales simultaneously, leading to refined bounds. We will discuss this argument in more detail now. Our presentation is based on Koltchinskii (2006).

First, note that the classical (global) Rademacher theory of Bartlett and Mendelson (2002) and Koltchinskii (2001) gives an excess risk bound of the following form: $\exists C > 0$ so that with probability larger then $1 - \exp(-t)$ it holds

$$\left| P\hat{f} - Pf^* \right| \leq C\left(R(\mathcal{F}) + \sqrt{\frac{t}{n}} \right) =: \delta\,, \tag{5}$$

where $\hat{f} := \operatorname{argmin}_{f \in \mathcal{F}} \frac{1}{n} \sum_{i=1}^{n} f(x_i)$, $f^* := \operatorname{argmin}_{f \in \mathcal{F}} Pf$, and $Pf := \mathbb{E} f(x)$. We denote the bound's value by δ and observe that, remarkably, if we consider the restricted class

$\mathcal{F}_\delta := \{f \in \mathcal{F} : |Pf - Pf^*| \leq \delta\}$, we have by (5) that $\hat{f} \in \mathcal{F}_\delta$ (and trivially $f^* \in \mathcal{F}_\delta$). This is remarkable and of significance because we can now state: with probability larger than $1 - \exp(-2t)$ it holds

$$\left|P\hat{f} - Pf^*\right| \leq C\left(R(\mathcal{F}_\delta) + \sqrt{\frac{t}{n}}\right). \tag{6}$$

The striking fact about the above inequality is that it depends on the complexity of the restricted class—no longer on the one of the original class; usually the complexity of the restricted class will be smaller than the one of the original class. Moreover, we can again denote the right-hand side of (6) by δ^{new} and repeat the argumentation. This way, we can step by step decrease the bound's value. If the bound (seen as a function in δ) defines a contraction, the limit of this iterative procedure is given by the fixed point of the bound.

This method has a serious limitation: although we can step by step decrease the Rademacher complexity occurring in the bound, the term $\sqrt{t/n}$ stays as it is and thus will hinder us from attaining a rate faster than $O(\sqrt{1/n})$. It would be desirable to have the term shrinking when passing to a smaller class $\mathcal{F}_\delta$. Can we replace the undesirable term by a more favorable one? And what properties would such a term need to have?

One of the basic foundations of learning theory are concentration inequalities (e.g., Bousquet et al., 2004). Even the most modern proof technique such as the fixed-point argument presented above can fail if it is built upon an insufficiently precise concentration inequality. As mentioned above, the stumbling block is the presence of the term $\sqrt{t/n}$ in the bound (5). The latter is a byproduct from the application of McDiarmid's inequality (McDiarmid, 1989)—a uniform version of Höffding's inequality—,which is used in Bartlett and Mendelson (2002) and Koltchinskii (2001) to relate the global Rademacher complexity with the excess risk.

The core idea now is that we can, instead of McDiarmid's inequality, use Talagrand's inequality (Talagrand, 1995), which is a uniform version of Bernstein's inequality. This gives

$$\left|P\hat{f} - Pf^*\right| \leq C\left(R(\mathcal{F}) + \sigma(\mathcal{F})\sqrt{\frac{t}{n}} + \frac{t}{n}\right) =: \delta. \tag{7}$$

Hereby $\sigma^2(\mathcal{F}) := \sup_{f\in\mathcal{F}} \mathbb{E}f^2$ is a bound on the (uncentered) "variance" of the functions considered. Now, denoting the right-hand side of (7) by δ, we obtain the following bound for the restricted class: $\exists C > 0$ so that with probability larger then $1 - \exp(-2t)$ it holds

$$\left|P\hat{f} - Pf^*\right| \leq C\left(R(\mathcal{F}_\delta) + \sigma(\mathcal{F}_\delta)\sqrt{\frac{t}{n}} + \frac{t}{n}\right). \tag{8}$$

As above, we denote the right-hand side of (8) by δ^{new} and repeat the argumentation. In general, we can expect the variance $\sigma^2(\mathcal{F}_\delta)$ to decrease step by step and if, seen as a function of δ, the bound defines a contraction, the limit is given by the fixed point of the bound.

It turns out that by this technique we can obtain fast convergence rates of the excess risk in the number of training examples n, which would be impossible by using global techniques such as the global Rademacher complexity or the Rademacher chaos complexity (Ying and Campbell, 2009), which—we recall—is in itself an upper bound on the global Rademacher complexity.

3.2 The Local Rademacher Complexity of MKL

In the context of ℓ_p-norm multiple kernel learning, we consider the hypothesis class H_p as defined in (1). Thus, given an i.i.d. sample $\boldsymbol{x}_1,\ldots,\boldsymbol{x}_n$ drawn from P, the local Rademacher complexity is given by $R_r(H_p) = \mathbb{E}\sup_{f_{\boldsymbol{w}}\in H_p: Pf_{\boldsymbol{w}}^2\le r}\langle \boldsymbol{w}, \frac{1}{n}\sum_{i=1}^n \sigma_i \boldsymbol{x}_i\rangle$, where $Pf_{\boldsymbol{w}}^2 := \mathbb{E}(f_{\boldsymbol{w}}(\boldsymbol{x}))^2$.

We will need the following assumption for the case $1 \le p \le 2$:

Assumption (A) (low-correlation). *There exists a $c_\delta \in (0,1]$ such that, for any $m \neq m'$ and $\boldsymbol{w}_m \in \mathcal{H}_m, \boldsymbol{w}_{m'} \in \mathcal{H}_{m'}$, the Hilbert-space-valued variables $\boldsymbol{x}^{(1)},\ldots,\boldsymbol{x}^{(M)}$ satisfy*

$$c_\delta \sum_{m=1}^M \mathbb{E}\left\langle \boldsymbol{w}_m, \boldsymbol{x}^{(m)}\right\rangle^2 \le \mathbb{E}\Big(\sum_{m=1}^M \left\langle \boldsymbol{w}_m, \boldsymbol{x}^{(m)}\right\rangle\Big)^2.$$

Since $\mathcal{H}_m, \mathcal{H}_{m'}$ are RKHSs with kernels $k_m, k_{m'}$, if we go back to the input random variable in the original space $X \in \mathcal{X}$, the above property means that for any fixed $t, t' \in \mathcal{X}$, the variables $k_m(X,t)$ and $k_{m'}(X,t')$ have a *low correlation*. In the most extreme case, $c_\delta = 1$, the variables are completely uncorrelated. This is the case, for example, if the original input space $\mathcal{X}$ is $\mathbb{R}^M$, the original input variable $X \in \mathcal{X}$ has independent coordinates, and the kernels $k_1,\ldots,k_M$ each act on a different coordinate. Such a setting was considered in particular by Raskutti et al. (2010) in the setting of ℓ_1-penalized MKL. We discuss this assumption in more detail in Section 6.3.

Note that, as self-adjoint, positive Hilbert-Schmidt operators, covariance operators enjoy discrete eigenvalue-eigenvector decompositions $J = \mathbb{E}\boldsymbol{x}\otimes\boldsymbol{x} = \sum_{j=1}^\infty \lambda_j \boldsymbol{u}_j \otimes \boldsymbol{u}_j$ and $J_m = \mathbb{E}\boldsymbol{x}^{(m)}\otimes\boldsymbol{x}^{(m)} = \sum_{j=1}^\infty \lambda_j^{(m)} \boldsymbol{u}_j^{(m)} \otimes \boldsymbol{u}_j^{(m)}$, where $(\boldsymbol{u}_j)_{j\ge1}$ and $(\boldsymbol{u}_j^{(m)})_{j\ge1}$ form orthonormal bases of $\mathcal{H}$ and $\mathcal{H}_m$, respectively.

We are now equipped to state our main results:

Theorem 5 (Local Rademacher complexity, $p \in [1,2]$)**.** *Assume that the kernels are uniformly bounded ($\|k\|_\infty \le B < \infty$) and that Assumption* **(A)** *holds. The local Rademacher complexity of the multi-kernel class H_p can be bounded for any $1 \le p \le 2$ as*

$$\forall t \in [p,2]: \quad R_r(H_p) \le \sqrt{\frac{16}{n}\left\|\left(\sum_{j=1}^\infty \min\left(rM^{1-\frac{2}{t^*}}, ceD^2t^{*2}\lambda_j^{(m)}\right)\right)_{m=1}^M\right\|_{\frac{t^*}{2}}} + \frac{\sqrt{Be}DM^{\frac{1}{t^*}}t^*}{n}.$$

Theorem 6 (Local Rademacher complexity, $p \ge 2$)**.** *The local Rademacher complexity of the multi-kernel class H_p can be bounded for any $p \ge 2$ as*

$$R_r(H_p) \le \sqrt{\frac{2}{n}\sum_{j=1}^\infty \min(r, D^2M^{\frac{2}{p^*}-1}\lambda_j)}.$$

Remark 7. *Note that for the case $p = 1$, by using $t = (\log(M))^*$ in Theorem 5, we obtain the bound*

$$R_r(H_1) \le \sqrt{\frac{16}{n}\left\|\left(\sum_{j=1}^\infty \min\left(rM, e^3D^2(\log M)^2\lambda_j^{(m)}\right)\right)_{m=1}^M\right\|_\infty} + \frac{\sqrt{B}e^{\frac{3}{2}}D\log(M)}{n},$$

for all $M \ge e^2$ (see below after the proof of Theorem 5 for a detailed justification).

Remark 8. *The result of Theorem 6 for $p \geq 2$ can be proved using considerably simpler techniques and without imposing assumptions on boundedness nor on uncorrelation of the kernels. If in addition the variables $(\boldsymbol{x}^{(m)})$ are centered and uncorrelated, then the spectra are related as follows :* $\operatorname{spec}(J) = \bigcup_{m=1}^{M} \operatorname{spec}(J_m)$*; that is,* $\{\lambda_i, i \geq 1\} = \bigcup_{m=1}^{M} \left\{\lambda_i^{(m)}, i \geq 1\right\}$*. Then one can write equivalently the bound of Theorem 6 as* $R_r(H_p) \leq \sqrt{\frac{2}{n}\sum_{m=1}^{M}\sum_{j=1}^{\infty}\min(r, D^2 M^{\frac{2}{p^*}-1}\lambda_j^{(m)})} = \sqrt{\frac{2}{n}\left\|\left(\sum_{j=1}^{\infty}\min(r, D^2 M^{\frac{2}{p^*}-1}\lambda_j^{(m)})\right)_{m=1}^{M}\right\|_1}$*. However, the main intended focus of this paper is on the more challenging case* $1 \leq p \leq 2$ *which is usually studied in multiple kernel learning and relevant in practice.*

Remark 9. *It is interesting to compare the above bounds for the special case $p = 2$ with the ones of Bartlett et al. (2005). The main term of the bound of Theorem 6 (taking $t = p = 2$) is then essentially determined by* $O\left(\sqrt{\frac{1}{n}\sum_{m=1}^{M}\sum_{j=1}^{\infty}\min\left(r, \lambda_j^{(m)}\right)}\right)$*. If the variables $(\boldsymbol{x}^{(m)})$ are centered and uncorrelated, by the relation between the spectra stated in Remark 8, this is equivalently of order* $O\left(\sqrt{\frac{1}{n}\sum_{j=1}^{\infty}\min\left(r, \lambda_j\right)}\right)$*, which is also what we obtain through Theorem 6, and coincides with the rate shown in Bartlett et al. (2005).*

Proof of Theorem 5 The proof is based on first relating the complexity of the class H_p with its centered counterpart, that is, where all functions $f_{\boldsymbol{w}} \in H_p$ are centered around their expected value. Then we compute the complexity of the centered class by decomposing the complexity into blocks, applying the no-correlation assumption, and using the inequalities of Hölder and Rosenthal. Then we relate it back to the original class, which we in the final step relate to a bound involving the truncation of the particular spectra of the kernels. Note that it suffices to prove the result for $t = p$ as trivially $R(H_p) \leq R(H_t)$ for all $p \leq t$.

STEP 1: RELATING THE ORIGINAL CLASS WITH THE CENTERED CLASS. In order to exploit the no-correlation assumption, we will work in large parts of the proof with the centered class $\tilde{H}_p = \left\{\tilde{f}_{\boldsymbol{w}} \;\middle|\; \|\boldsymbol{w}\|_{2,p} \leq D\right\}$, wherein $\tilde{f}_{\boldsymbol{w}} : \boldsymbol{x} \mapsto \langle \boldsymbol{w}, \tilde{\boldsymbol{x}}\rangle$, and $\tilde{\boldsymbol{x}} := \boldsymbol{x} - \mathbb{E}\boldsymbol{x}$. We start the proof by noting that $\tilde{f}_{\boldsymbol{w}}(\boldsymbol{x}) = f_{\boldsymbol{w}}(\boldsymbol{x}) - \langle \boldsymbol{w}, \mathbb{E}\boldsymbol{x}\rangle = f_{\boldsymbol{w}}(\boldsymbol{x}) - \mathbb{E}\langle \boldsymbol{w}, \boldsymbol{x}\rangle = f_{\boldsymbol{w}}(\boldsymbol{x}) - \mathbb{E}f_{\boldsymbol{w}}(\boldsymbol{x})$, so that, by the bias-variance decomposition, it holds that

$$Pf_{\boldsymbol{w}}^2 = \mathbb{E}f_{\boldsymbol{w}}(\boldsymbol{x})^2 = \mathbb{E}\left(f_{\boldsymbol{w}}(\boldsymbol{x}) - \mathbb{E}f_w(\boldsymbol{x})\right)^2 + \left(\mathbb{E}f_w(\boldsymbol{x})\right)^2 = P\tilde{f}_{\boldsymbol{w}}^2 \;+\; \left(Pf_{\boldsymbol{w}}\right)^2. \tag{9}$$

Furthermore we note that by Jensen's inequality

$$\begin{aligned}\left\|\mathbb{E}\boldsymbol{x}\right\|_{2,p^*} &= \left(\sum_{m=1}^{M}\left\|\mathbb{E}\boldsymbol{x}^{(m)}\right\|_2^{p^*}\right)^{\frac{1}{p^*}} = \left(\sum_{m=1}^{M}\left\langle \mathbb{E}\boldsymbol{x}^{(m)}, \mathbb{E}\boldsymbol{x}^{(m)}\right\rangle^{\frac{p^*}{2}}\right)^{\frac{1}{p^*}} \\ &\overset{\text{Jensen}}{\leq} \left(\sum_{m=1}^{M}\mathbb{E}\left\langle \boldsymbol{x}^{(m)}, \boldsymbol{x}^{(m)}\right\rangle^{\frac{p^*}{2}}\right)^{\frac{1}{p^*}} \;=\; \sqrt{\left\|\left(\operatorname{tr}(J_m)\right)_{m=1}^{M}\right\|_{\frac{p^*}{2}}}\end{aligned} \tag{10}$$

so that we can express the complexity of the centered class in terms of the uncentered one as follows:

$$\begin{aligned} R_r(H_p) &= \mathbb{E} \sup_{\substack{f_{\boldsymbol{w}} \in H_p, \\ Pf_{\boldsymbol{w}}^2 \leq r}} \left\langle \boldsymbol{w}, \frac{1}{n} \sum_{i=1}^n \sigma_i \boldsymbol{x}_i \right\rangle \\ &\leq \mathbb{E} \sup_{\substack{f_{\boldsymbol{w}} \in H_p, \\ Pf_{\boldsymbol{w}}^2 \leq r}} \left\langle \boldsymbol{w}, \frac{1}{n} \sum_{i=1}^n \sigma_i \tilde{\boldsymbol{x}}_i \right\rangle + \mathbb{E} \sup_{\substack{f_{\boldsymbol{w}} \in H_p, \\ Pf_{\boldsymbol{w}}^2 \leq r}} \left\langle \boldsymbol{w}, \frac{1}{n} \sum_{i=1}^n \sigma_i \mathbb{E}\boldsymbol{x} \right\rangle \end{aligned}$$

Concerning the first term of the above upper bound, using (9) we have $P\tilde{f}_{\boldsymbol{w}}^2 \leq Pf_{\boldsymbol{w}}^2$, and thus

$$\mathbb{E} \sup_{\substack{f_{\boldsymbol{w}} \in H_p, \\ Pf_{\boldsymbol{w}}^2 \leq r}} \left\langle \boldsymbol{w}, \frac{1}{n} \sum_{i=1}^n \sigma_i \tilde{\boldsymbol{x}}_i \right\rangle \leq \mathbb{E} \sup_{\substack{f_{\boldsymbol{w}} \in H_p, \\ P\tilde{f}_{\boldsymbol{w}}^2 \leq r}} \left\langle \boldsymbol{w}, \frac{1}{n} \sum_{i=1}^n \sigma_i \tilde{\boldsymbol{x}}_i \right\rangle = R_r(\tilde{H}_p).$$

Now to bound the second term, we write

$$\begin{aligned} \mathbb{E} \sup_{\substack{f_{\boldsymbol{w}} \in H_p, \\ Pf_{\boldsymbol{w}}^2 \leq r}} \left\langle \boldsymbol{w}, \frac{1}{n} \sum_{i=1}^n \sigma_i \mathbb{E}\boldsymbol{x} \right\rangle &= \mathbb{E} \left| \frac{1}{n} \sum_{i=1}^n \sigma_i \right| \sup_{\substack{f_{\boldsymbol{w}} \in H_p, \\ Pf_{\boldsymbol{w}}^2 \leq r}} \langle \boldsymbol{w}, \mathbb{E}\boldsymbol{x} \rangle \\ &\leq \sup_{\substack{f_{\boldsymbol{w}} \in H_p, \\ Pf_{\boldsymbol{w}}^2 \leq r}} \langle \boldsymbol{w}, \mathbb{E}\boldsymbol{x} \rangle \left(\mathbb{E} \left(\frac{1}{n} \sum_{i=1}^n \sigma_i \right)^2 \right)^{\frac{1}{2}} \\ &= \sqrt{n} \sup_{\substack{f_{\boldsymbol{w}} \in H_p, \\ Pf_{\boldsymbol{w}}^2 \leq r}} \langle \boldsymbol{w}, \mathbb{E}\boldsymbol{x} \rangle . \end{aligned}$$

Now observe finally that we have

$$\langle \boldsymbol{w}, \mathbb{E}\boldsymbol{x} \rangle \overset{\text{Hölder}}{\leq} \|\boldsymbol{w}\|_{2,p} \|\mathbb{E}\boldsymbol{x}\|_{2,p^*} \overset{(10)}{\leq} \|\boldsymbol{w}\|_{2,p} \sqrt{\left\| \left(\operatorname{tr}(J_m) \right)_{m=1}^M \right\|_{\frac{p^*}{2}}}$$

as well as

$$\langle \boldsymbol{w}, \mathbb{E}\boldsymbol{x} \rangle = \mathbb{E} f_{\boldsymbol{w}}(\boldsymbol{x}) \leq \sqrt{Pf_{\boldsymbol{w}}^2}.$$

We finally obtain, putting together the steps above,

$$R_r(H_p) \leq R_r(\tilde{H}_p) + n^{-\frac{1}{2}} \min \left(\sqrt{r}, D \sqrt{\left\| \left(\operatorname{tr}(J_m) \right)_{m=1}^M \right\|_{\frac{p^*}{2}}} \right) \tag{11}$$

This shows that we at the expense of the additional summand on the right hand side we can work with the centered class instead of the uncentered one.

STEP 2: BOUNDING THE COMPLEXITY OF THE CENTERED CLASS. Since the (centered) covariance operator $\mathbb{E}\tilde{\boldsymbol{x}}^{(m)} \otimes \tilde{\boldsymbol{x}}^{(m)}$ is also a self-adjoint Hilbert-Schmidt operator on $\mathcal{H}_m$, there exists an eigendecomposition

$$\mathbb{E}\tilde{\boldsymbol{x}}^{(m)} \otimes \tilde{\boldsymbol{x}}^{(m)} = \sum_{j=1}^{\infty} \tilde{\lambda}_j^{(m)} \tilde{\boldsymbol{u}}_j^{(m)} \otimes \tilde{\boldsymbol{u}}_j^{(m)}, \tag{12}$$

wherein $(\tilde{\boldsymbol{u}}_j^{(m)})_{j\geq 1}$ is an orthogonal basis of $\mathcal{H}_m$. Furthermore, the no-correlation assumption **(A)** entails $\mathbb{E}\tilde{\boldsymbol{x}}^{(l)}\otimes\tilde{\boldsymbol{x}}^{(m)}=\mathbf{0}$ for all $l\neq m$. As a consequence,

$$\begin{aligned} P\tilde{f}_{\boldsymbol{w}}^2 &= \mathbb{E}(f_{\boldsymbol{w}}(\tilde{\boldsymbol{x}}))^2 = \mathbb{E}\Big(\sum_{m=1}^M \big\langle \boldsymbol{w}_m, \tilde{\boldsymbol{x}}^{(m)}\big\rangle\Big)^2 = \sum_{l,m=1}^M \Big\langle \boldsymbol{w}_l, \big(\mathbb{E}\tilde{\boldsymbol{x}}^{(l)}\otimes\tilde{\boldsymbol{x}}^{(m)}\big)\boldsymbol{w}_m\Big\rangle \\ &\overset{\text{(A)}}{\geq} c_\delta \sum_{m=1}^M \Big\langle \boldsymbol{w}_m, \big(\mathbb{E}\tilde{\boldsymbol{x}}^{(m)}\otimes\tilde{\boldsymbol{x}}^{(m)}\big)\boldsymbol{w}_m\Big\rangle = \sum_{m=1}^M\sum_{j=1}^\infty \tilde{\lambda}_j^{(m)}\Big\langle \boldsymbol{w}_m, \tilde{\boldsymbol{u}}_j^{(m)}\Big\rangle^2 \end{aligned} \tag{13}$$

and, for all j and m,

$$\begin{aligned} \mathbb{E}\Big\langle \frac{1}{n}\sum_{i=1}^n \sigma_i\tilde{\boldsymbol{x}}_i^{(m)}, \tilde{\boldsymbol{u}}_j^{(m)}\Big\rangle^2 &= \mathbb{E}\frac{1}{n^2}\sum_{i,l=1}^n \sigma_i\sigma_l \Big\langle \tilde{\boldsymbol{x}}_i^{(m)}, \tilde{\boldsymbol{u}}_j^{(m)}\Big\rangle\Big\langle \tilde{\boldsymbol{x}}_l^{(m)}, \tilde{\boldsymbol{u}}_j^{(m)}\Big\rangle \overset{\sigma \text{ i.i.d.}}{=} \mathbb{E}\frac{1}{n^2}\sum_{i=1}^n \Big\langle \tilde{\boldsymbol{x}}_i^{(m)}, \tilde{\boldsymbol{u}}_j^{(m)}\Big\rangle^2 \\ &= \frac{1}{n}\Big\langle \tilde{\boldsymbol{u}}_j^{(m)}, \underbrace{\Big(\frac{1}{n}\sum_{i=1}^n \mathbb{E}\tilde{\boldsymbol{x}}_i^{(m)}\otimes\tilde{\boldsymbol{x}}_i^{(m)}\Big)}_{=\mathbb{E}\tilde{\boldsymbol{x}}^{(m)}\otimes\tilde{\boldsymbol{x}}^{(m)}}\tilde{\boldsymbol{u}}_j^{(m)}\Big\rangle = \frac{\tilde{\lambda}_j^{(m)}}{n}. \end{aligned} \tag{14}$$

Now, let $h_1,\ldots,h_M$ be arbitrary nonnegative integers. We can express the local Rademacher complexity in terms of the eigendecomposition (12) as follows

$$\begin{aligned} R_r(\tilde{H}_p) &= \mathbb{E}\sup_{f_{\boldsymbol{w}}\in\tilde{H}_p: P\tilde{f}_{\boldsymbol{w}}^2\leq r}\Big\langle \boldsymbol{w}, \frac{1}{n}\sum_{i=1}^n \sigma_i\tilde{\boldsymbol{x}}_i\Big\rangle \\ &= \mathbb{E}\sup_{f_{\boldsymbol{w}}\in\tilde{H}_p: P\tilde{f}_{\boldsymbol{w}}^2\leq r}\Big\langle \big(\boldsymbol{w}^{(m)}\big)_{m=1}^M, \big(\frac{1}{n}\sum_{i=1}^n \sigma_i\tilde{\boldsymbol{x}}_i^{(m)}\big)_{m=1}^M\Big\rangle \\ &= \mathbb{E}\sup_{f_{\boldsymbol{w}}\in\tilde{H}_p: P\tilde{f}_{\boldsymbol{w}}^2\leq r}\Big\langle \boldsymbol{w}, \Big(\sum_{j=1}^\infty \langle\frac{1}{n}\sum_{i=1}^n \sigma_i\tilde{\boldsymbol{x}}_i^{(m)}, \tilde{\boldsymbol{u}}_j^{(m)}\rangle\tilde{\boldsymbol{u}}_j^{(m)}\Big)_{m=1}^M\Big\rangle \\ &\overset{(\star)}{=} \mathbb{E}\sup_{P\tilde{f}_{\boldsymbol{w}}^2\leq r}\Big\langle \Big(\sum_{j=1}^{h_m}\sqrt{\tilde{\lambda}_j^{(m)}}\langle \boldsymbol{w}^{(m)}, \tilde{\boldsymbol{u}}_j^{(m)}\rangle\tilde{\boldsymbol{u}}_j^{(m)}\Big)_{m=1}^M, \\ &\qquad\qquad \Big(\sum_{j=1}^{h_m}\sqrt{\tilde{\lambda}_j^{(m)}}^{-1}\langle\frac{1}{n}\sum_{i=1}^n \sigma_i\tilde{\boldsymbol{x}}_i^{(m)}, \tilde{\boldsymbol{u}}_j^{(m)}\rangle\tilde{\boldsymbol{u}}_j^{(m)}\Big)_{m=1}^M\Big\rangle \\ &\quad + \mathbb{E}\sup_{f_{\boldsymbol{w}}\in\tilde{H}_p}\Big\langle \boldsymbol{w}, \Big(\sum_{j=h_m+1}^\infty \langle\frac{1}{n}\sum_{i=1}^n \sigma_i\tilde{\boldsymbol{x}}_i^{(m)}, \tilde{\boldsymbol{u}}_j^{(m)}\rangle\tilde{\boldsymbol{u}}_j^{(m)}\Big)_{m=1}^M\Big\rangle \\ &\overset{\text{C.-S., Jensen}}{\leq} \sup_{P\tilde{f}_{\boldsymbol{w}}^2\leq r}\Bigg[\Bigg(\sum_{m=1}^M\sum_{j=1}^{h_m}\tilde{\lambda}_j^{(m)}\langle \boldsymbol{w}^{(m)}, \tilde{\boldsymbol{u}}_j^{(m)}\rangle^2\Bigg)^{\frac{1}{2}} \\ &\qquad\qquad \times\Bigg(\sum_{m=1}^M\sum_{j=1}^{h_m}\Big(\tilde{\lambda}_j^{(m)}\Big)^{-1}\mathbb{E}\langle\frac{1}{n}\sum_{i=1}^n \sigma_i\tilde{\boldsymbol{x}}_i^{(m)}, \tilde{\boldsymbol{u}}_j^{(m)}\rangle^2\Bigg)^{\frac{1}{2}}\Bigg] \\ &\quad + \mathbb{E}\sup_{f_{\boldsymbol{w}}\in\tilde{H}_p}\Big\langle \boldsymbol{w}, \Big(\sum_{j=h_m+1}^\infty \langle\frac{1}{n}\sum_{i=1}^n \sigma_i\tilde{\boldsymbol{x}}_i^{(m)}, \tilde{\boldsymbol{u}}_j^{(m)}\rangle\tilde{\boldsymbol{u}}_j^{(m)}\Big)_{m=1}^M\Big\rangle, \end{aligned}$$

where for $(\star)$ we use the linearity of the scalar product, so that (13) and (14) yield

$$R_r(\tilde{H}_p) \overset{(13),\,(14)}{\leq} \sqrt{\frac{rc_\delta^{-1}\sum_{m=1}^M h_m}{n}} + \mathbb{E}\sup_{f_{\boldsymbol{w}}\in\tilde{H}_p}\left\langle \boldsymbol{w}, \Big(\sum_{j=h_m+1}^{\infty}\langle\frac{1}{n}\sum_{i=1}^n \sigma_i\tilde{\boldsymbol{x}}_i^{(m)},\tilde{\boldsymbol{u}}_j^{(m)}\rangle\tilde{\boldsymbol{u}}_j^{(m)}\Big)_{m=1}^M\right\rangle$$
$$\overset{\text{Hölder}}{\leq} \sqrt{\frac{rc_\delta^{-1}\sum_{m=1}^M h_m}{n}} + D\,\mathbb{E}\left\|\Big(\sum_{j=h_m+1}^{\infty}\langle\frac{1}{n}\sum_{i=1}^n \sigma_i\tilde{\boldsymbol{x}}_i^{(m)},\tilde{\boldsymbol{u}}_j^{(m)}\rangle\tilde{\boldsymbol{u}}_j^{(m)}\Big)_{m=1}^M\right\|_{2,p^*}.$$

STEP 3: KHINTCHINE-KAHANE'S AND ROSENTHAL'S INEQUALITIES. We can now use the Khintchine-Kahane (K.-K.) inequality (see Lemma 1 in Appendix B) to further bound the right term in the above expression as follows

$$\mathbb{E}\left\|\Big(\sum_{j=h_m+1}^{\infty}\langle\frac{1}{n}\sum_{i=1}^n \sigma_i\tilde{\boldsymbol{x}}_i^{(m)},\tilde{\boldsymbol{u}}_j^{(m)}\rangle\tilde{\boldsymbol{u}}_j^{(m)}\Big)_{m=1}^M\right\|_{2,p^*}$$
$$\overset{\text{Jensen}}{\leq} \mathbb{E}\Bigg(\sum_{m=1}^M \mathbb{E}_{\boldsymbol{\sigma}}\left\|\sum_{j=h_m+1}^{\infty}\langle\frac{1}{n}\sum_{i=1}^n \sigma_i\tilde{\boldsymbol{x}}_i^{(m)},\tilde{\boldsymbol{u}}_j^{(m)}\rangle\tilde{\boldsymbol{u}}_j^{(m)}\right\|_{\mathcal{H}_m}^{p^*}\Bigg)^{\frac{1}{p^*}}$$
$$\overset{\text{K.-K.}}{\leq} \sqrt{\frac{p^*}{n}}\,\mathbb{E}\Bigg(\sum_{m=1}^M\Big(\sum_{j=h_m+1}^{\infty}\frac{1}{n}\sum_{i=1}^n\langle\tilde{\boldsymbol{x}}_i^{(m)},\tilde{\boldsymbol{u}}_j^{(m)}\rangle^2\Big)^{\frac{p^*}{2}}\Bigg)^{\frac{1}{p^*}}$$
$$\overset{\text{Jensen}}{\leq} \sqrt{\frac{p^*}{n}}\,\Bigg(\sum_{m=1}^M\mathbb{E}\Big(\sum_{j=h_m+1}^{\infty}\frac{1}{n}\sum_{i=1}^n\langle\tilde{\boldsymbol{x}}_i^{(m)},\tilde{\boldsymbol{u}}_j^{(m)}\rangle^2\Big)^{\frac{p^*}{2}}\Bigg)^{\frac{1}{p^*}},$$

Note that for $p\geq 2$ it holds that $p^*/2\leq 1$, and thus it suffices to employ Jensen's inequality once again in order to move the expectation operator inside the inner term. In the general case we need a handle on the $\frac{p^*}{2}$-th moments and to this end employ Lemma 3 (Rosenthal + Young), which yields

$$\Bigg(\sum_{m=1}^M\mathbb{E}\Big(\sum_{j=h_m+1}^{\infty}\frac{1}{n}\sum_{i=1}^n\langle\tilde{\boldsymbol{x}}_i^{(m)},\tilde{\boldsymbol{u}}_j^{(m)}\rangle^2\Big)^{\frac{p^*}{2}}\Bigg)^{\frac{1}{p^*}}$$
$$\overset{\text{R+Y}}{\leq}\Bigg(\sum_{m=1}^M(ep^*)^{\frac{p^*}{2}}\Bigg(\Big(\frac{B}{n}\Big)^{\frac{p^*}{2}}+\Big(\sum_{j=h_m+1}^{\infty}\underbrace{\frac{1}{n}\sum_{i=1}^n\mathbb{E}\langle\tilde{\boldsymbol{x}}_i^{(m)},\tilde{\boldsymbol{u}}_j^{(m)}\rangle^2}_{=\tilde{\lambda}_j^{(m)}}\Big)^{\frac{p^*}{2}}\Bigg)\Bigg)^{\frac{1}{p^*}}$$
$$\overset{(*)}{\leq}\sqrt{ep^*\left(\frac{BM^{\frac{2}{p^*}}}{n}+\Bigg(\sum_{m=1}^M\Big(\sum_{j=h_m+1}^{\infty}\tilde{\lambda}_j^{(m)}\Big)^{\frac{p^*}{2}}\Bigg)^{\frac{2}{p^*}}\right)}$$
$$=\sqrt{ep^*\left(\frac{BM^{\frac{2}{p^*}}}{n}+\left\|\Big(\sum_{j=h_m+1}^{\infty}\tilde{\lambda}_j^{(m)}\Big)_{m=1}^M\right\|_{\frac{p^*}{2}}\right)}$$
$$\leq\sqrt{ep^*\left(\frac{BM^{\frac{2}{p^*}}}{n}+\left\|\Big(\sum_{j=h_m+1}^{\infty}\lambda_j^{(m)}\Big)_{m=1}^M\right\|_{\frac{p^*}{2}}\right)}$$

where for $(*)$ we used the subadditivity of $\sqrt[p^*]{\cdot}$ and in the last step we applied the Lidskii-Mirsky-Wielandt theorem which gives $\forall j,m:\ \tilde{\lambda}_j^{(m)} \leq \lambda_j^{(m)}$. Thus by the subadditivity of the root function

$$\begin{aligned} R_r(\tilde{H}_p) &\leq \sqrt{\frac{rc_\delta^{-1}\sum_{m=1}^M h_m}{n}} + D\sqrt{\frac{ep^{*2}}{n}\left(\frac{BM^{\frac{2}{p^*}}}{n} + \left\|\left(\sum_{j=h_m+1}^{\infty}\lambda_j^{(m)}\right)_{m=1}^M\right\|_{\frac{p^*}{2}}\right)} \\ &= \sqrt{\frac{rc_\delta^{-1}\sum_{m=1}^M h_m}{n}} + \sqrt{\frac{ep^{*2}D^2}{n}\left\|\left(\sum_{j=h_m+1}^{\infty}\lambda_j^{(m)}\right)_{m=1}^M\right\|_{\frac{p^*}{2}}} + \frac{\sqrt{Be}DM^{\frac{1}{p^*}}p^*}{n}. \end{aligned} \tag{15}$$

STEP 4: BOUNDING THE COMPLEXITY OF THE ORIGINAL CLASS. Now note that for all nonnegative integers h_m we either have

$$n^{-\frac{1}{2}}\min\left(\sqrt{rc_\delta^{-1}}, D\sqrt{\left\|\left(\mathrm{tr}(J_m)\right)_{m=1}^M\right\|_{\frac{p^*}{2}}}\right) \leq \sqrt{\frac{ep^{*2}D^2}{n}\left\|\left(\sum_{j=h_m+1}^{\infty}\lambda_j^{(m)}\right)_{m=1}^M\right\|_{\frac{p^*}{2}}}$$

(in case all h_m are zero) or it holds

$$n^{-\frac{1}{2}}\min\left(\sqrt{rc_\delta^{-1}}, D\sqrt{\left\|\left(\mathrm{tr}(J_m)\right)_{m=1}^M\right\|_{\frac{p^*}{2}}}\right) \leq \sqrt{\frac{rc_\delta^{-1}\sum_{m=1}^M h_m}{n}}$$

(in case that at least one h_m is nonzero) so that in any case we get

$$\begin{aligned} &n^{-\frac{1}{2}}\min\left(\sqrt{rc_\delta^{-1}}, D\sqrt{\left\|\left(\mathrm{tr}(J_m)\right)_{m=1}^M\right\|_{\frac{p^*}{2}}}\right) \\ &\qquad\leq \sqrt{\frac{rc_\delta^{-1}\sum_{m=1}^M h_m}{n}} + \sqrt{\frac{ep^{*2}D^2}{n}\left\|\left(\sum_{j=h_m+1}^{\infty}\lambda_j^{(m)}\right)_{m=1}^M\right\|_{\frac{p^*}{2}}}. \end{aligned} \tag{16}$$

Thus the following preliminary bound follows from (11) by (15) and (16):

$$R_r(H_p) \leq \sqrt{\frac{4rc_\delta^{-1}\sum_{m=1}^M h_m}{n}} + \sqrt{\frac{4ep^{*2}D^2}{n}\left\|\left(\sum_{j=h_m+1}^{\infty}\lambda_j^{(m)}\right)_{m=1}^M\right\|_{\frac{p^*}{2}}} + \frac{\sqrt{Be}DM^{\frac{1}{p^*}}p^*}{n}, \tag{17}$$

for all nonnegative integers $h_m \geq 0$. We could stop here as the above bound is already the one that will be used in the subsequent section for the computation of the excess loss bounds. However, we can work a little more on the form of the bound to gain more insight on its properties—we will show that it is related to the truncation of the spectra at the scale r.

STEP 5: RELATING THE BOUND TO THE TRUNCATION OF THE SPECTRA OF THE KERNELS. To this end, notice that for all nonnegative real numbers A_1, A_2 and any $\boldsymbol{a}_1, \boldsymbol{a}_2 \in \mathbb{R}_+^m$ it holds for all $q \geq 1$

$$\sqrt{A_1} + \sqrt{A_2} \leq \sqrt{2(A_1+A_2)} \tag{18}$$

$$\|\boldsymbol{a}_1\|_q + \|\boldsymbol{a}_2\|_q \leq 2^{1-\frac{1}{q}}\|\boldsymbol{a}_1+\boldsymbol{a}_2\|_q \leq 2\|\boldsymbol{a}_1+\boldsymbol{a}_2\|_q \tag{19}$$

(the first statement follows from the concavity of the square root function and the second one is proved in appendix B; see Lemma 17) and thus

$$\begin{aligned}
&R_r(H_p)\\
&\overset{(18)}{\leq} \quad \sqrt{8\left(\frac{rc_\delta^{-1}\sum_{m=1}^M h_m}{n}+\frac{ep^{*2}D^2}{n}\left\|\left(\sum_{j=h_m+1}^{\infty}\lambda_j^{(m)}\right)_{m=1}^M\right\|_{\frac{p^*}{2}}\right)}+\frac{\sqrt{Be}DM^{\frac{1}{p^*}}p^*}{n}\\
&\overset{\ell_1\text{-to-}\ell_{\frac{p^*}{2}}}{\leq} \quad \sqrt{\frac{8}{n}\left(rc_\delta^{-1}M^{1-\frac{2}{p^*}}\left\|\left(h_m\right)_{m=1}^M\right\|_{\frac{p^*}{2}}+ep^{*2}D^2\left\|\left(\sum_{j=h_m+1}^{\infty}\lambda_j^{(m)}\right)_{m=1}^M\right\|_{\frac{p^*}{2}}\right)}+\frac{\sqrt{Be}DM^{\frac{1}{p^*}}p^*}{n}\\
&\overset{(19)}{\leq} \quad \sqrt{\frac{16}{n}\left\|\left(rc_\delta^{-1}M^{1-\frac{2}{p^*}}h_m+ep^{*2}D^2\sum_{j=h_m+1}^{\infty}\lambda_j^{(m)}\right)_{m=1}^M\right\|_{\frac{p^*}{2}}}+\frac{\sqrt{Be}DM^{\frac{1}{p^*}}p^*}{n},
\end{aligned}$$

where to obtain the second inequality we applied that for all non-negative $\boldsymbol{a}\in\mathbb{R}^M$ and $0<q<p\leq\infty$ it holds[2]

$$(\ell_q\text{-to-}\ell_p\text{ conversion})\qquad \|\boldsymbol{a}\|_q=\langle\mathbf{1},\boldsymbol{a}^q\rangle^{\frac{1}{q}}\overset{\text{Hölder}}{\leq}\left(\|\mathbf{1}\|_{(p/q)^*}\|\boldsymbol{a}^q\|_{p/q}\right)^{1/q}=M^{\frac{1}{q}-\frac{1}{p}}\|\boldsymbol{a}\|_p. \tag{20}$$

Since the above holds for all nonnegative integers h_m, it follows

$$\begin{aligned}
R_r(H_p) &\leq \sqrt{\frac{16}{n}\left\|\left(\min_{h_m\geq 0} rc_\delta^{-1}M^{1-\frac{2}{p^*}}h_m+ep^{*2}D^2\sum_{j=h_m+1}^{\infty}\lambda_j^{(m)}\right)_{m=1}^M\right\|_{\frac{p^*}{2}}}+\frac{\sqrt{Be}DM^{\frac{1}{p^*}}p^*}{n}\\
&= \sqrt{\frac{16}{n}\left\|\left(\sum_{j=1}^{\infty}\min\left(rc_\delta^{-1}M^{1-\frac{2}{p^*}},ep^{*2}D^2\lambda_j^{(m)}\right)\right)_{m=1}^M\right\|_{\frac{p^*}{2}}}+\frac{\sqrt{Be}DM^{\frac{1}{p^*}}p^*}{n},
\end{aligned}$$

which completes the proof of the theorem. ■

Proof of Remark 7 To see that Remark 7 holds notice that $R(H_1)\leq R(H_p)$ for all $p\geq 1$ and thus by choosing $p=(\log(M))^*$ the above bound implies

$$\begin{aligned}
R_r(H_1) &\leq \sqrt{\frac{16}{n}\left\|\left(\sum_{j=1}^{\infty}\min\left(rc_\delta^{-1}M^{1-\frac{2}{p^*}},ep^{*2}D^2\lambda_j^{(m)}\right)\right)_{m=1}^M\right\|_{\frac{p^*}{2}}}+\frac{\sqrt{Be}DM^{\frac{1}{p^*}}p^*}{n}\\
&\overset{\ell_{\frac{p^*}{2}}\text{-to-}\ell_\infty}{\leq} \sqrt{\frac{16}{n}\left\|\left(\sum_{j=1}^{\infty}\min\left(rc_\delta^{-1}M,ep^{*2}M^{\frac{2}{p^*}}D^2\lambda_j^{(m)}\right)\right)_{m=1}^M\right\|_{\infty}}+\frac{\sqrt{Be}DM^{\frac{1}{p^*}}p^*}{n}\\
&= \sqrt{\frac{16}{n}\left\|\left(\sum_{j=1}^{\infty}\min\left(rc_\delta^{-1}M,e^3D^2(\log M)^2\lambda_j^{(m)}\right)\right)_{m=1}^M\right\|_{\infty}}+\frac{\sqrt{Be}^{\frac{3}{2}}D(\log M)}{n},
\end{aligned}$$

which completes the proof. ■

2. We denote by $\boldsymbol{a}^q$ the vector with entries a_i^q and by $\mathbf{1}$ the vector with entries all 1.

Proof of Theorem 6.

The eigendecomposition $\mathbb{E}\boldsymbol{x}\otimes\boldsymbol{x}=\sum_{j=1}^{\infty}\lambda_j\boldsymbol{u}_j\otimes\boldsymbol{u}_j$ yields

$$Pf_{\boldsymbol{w}}^2=\mathbb{E}(f_{\boldsymbol{w}}(\boldsymbol{x}))^2=\mathbb{E}\langle\boldsymbol{w},\boldsymbol{x}\rangle^2=\big\langle\boldsymbol{w},(\mathbb{E}\boldsymbol{x}\otimes\boldsymbol{x})\boldsymbol{w}\big\rangle=\sum_{j=1}^{\infty}\lambda_j\big\langle\boldsymbol{w},\boldsymbol{u}_j\big\rangle^2, \tag{21}$$

and, for all j

$$\begin{aligned}\mathbb{E}\Big\langle\frac{1}{n}\sum_{i=1}^{n}\sigma_i\boldsymbol{x}_i,\boldsymbol{u}_j\Big\rangle^2 &= \mathbb{E}\frac{1}{n^2}\sum_{i,l=1}^{n}\sigma_i\sigma_l\big\langle\boldsymbol{x}_i,\boldsymbol{u}_j\big\rangle\big\langle\boldsymbol{x}_l,\boldsymbol{u}_j\big\rangle \overset{\sigma\text{ i.i.d.}}{=}\mathbb{E}\frac{1}{n^2}\sum_{i=1}^{n}\big\langle\boldsymbol{x}_i,\boldsymbol{u}_j\big\rangle^2\\ &= \frac{1}{n}\Big\langle\boldsymbol{u}_j,\Big(\underbrace{\frac{1}{n}\sum_{i=1}^{n}\mathbb{E}\boldsymbol{x}_i\otimes\boldsymbol{x}_i}_{=\mathbb{E}\boldsymbol{x}\otimes\boldsymbol{x}}\Big)\boldsymbol{u}_j\Big\rangle=\frac{\lambda_j}{n}.\end{aligned} \tag{22}$$

Therefore, we can use, for any nonnegative integer h, the Cauchy-Schwarz inequality and a block-structured version of Hölder's inequality (see Lemma 15) to bound the local Rademacher complexity as follows:

$$\begin{aligned}R_r(H_p) &= \mathbb{E}\sup_{f_{\boldsymbol{w}}\in H_p:Pf_{\boldsymbol{w}}^2\le r}\Big\langle\boldsymbol{w},\frac{1}{n}\sum_{i=1}^{n}\sigma_i\boldsymbol{x}_i\Big\rangle\\ &= \mathbb{E}\sup_{f_{\boldsymbol{w}}\in H_p:Pf_{\boldsymbol{w}}^2\le r}\Big\langle\sum_{j=1}^{h}\sqrt{\lambda_j}\langle\boldsymbol{w},\boldsymbol{u}_j\rangle\boldsymbol{u}_j,\sum_{j=1}^{h}\sqrt{\lambda_j}^{-1}\langle\frac{1}{n}\sum_{i=1}^{n}\sigma_i\boldsymbol{x}_i,\boldsymbol{u}_j\rangle\boldsymbol{u}_j\Big\rangle\\ &\qquad+\Big\langle\boldsymbol{w},\sum_{j=h+1}^{\infty}\langle\frac{1}{n}\sum_{i=1}^{n}\sigma_i\boldsymbol{x}_i,\boldsymbol{u}_j\rangle\boldsymbol{u}_j\Big\rangle\\ &\overset{\text{C.-S., (21), (22)}}{\le}\sqrt{\frac{rh}{n}}+\mathbb{E}\sup_{f_{\boldsymbol{w}}\in H_p}\Big\langle\boldsymbol{w},\sum_{j=h+1}^{\infty}\langle\frac{1}{n}\sum_{i=1}^{n}\sigma_i\boldsymbol{x}_i,\boldsymbol{u}_j\rangle\boldsymbol{u}_j\Big\rangle\\ &\overset{\text{Hölder}}{\le}\sqrt{\frac{rh}{n}}+D\mathbb{E}\Big\|\sum_{j=h+1}^{\infty}\langle\frac{1}{n}\sum_{i=1}^{n}\sigma_i\boldsymbol{x}_i,\boldsymbol{u}_j\rangle\boldsymbol{u}_j\Big\|_{2,p^*}\\ &\overset{\ell_{\frac{p^*}{2}}\text{-to-}\ell_2}{\le}\sqrt{\frac{rh}{n}}+DM^{\frac{1}{p^*}-\frac{1}{2}}\mathbb{E}\Big\|\sum_{j=h+1}^{\infty}\langle\frac{1}{n}\sum_{i=1}^{n}\sigma_i\boldsymbol{x}_i,\boldsymbol{u}_j\rangle\boldsymbol{u}_j\Big\|_{\mathcal{H}}\\ &\overset{\text{Jensen}}{\le}\sqrt{\frac{rh}{n}}+DM^{\frac{1}{p^*}-\frac{1}{2}}\Bigg(\sum_{j=h+1}^{\infty}\underbrace{\mathbb{E}\langle\frac{1}{n}\sum_{i=1}^{n}\sigma_i\boldsymbol{x}_i,\boldsymbol{u}_j\rangle^2}_{\overset{(22)}{\le}\frac{\lambda_j}{n}}\Bigg)^{\frac{1}{2}}\\ &\le\sqrt{\frac{rh}{n}}+\sqrt{\frac{D^2M^{\frac{2}{p^*}-1}}{n}\sum_{j=h+1}^{\infty}\lambda_j}.\end{aligned}$$

Since the above holds for all h, the result now follows from $\sqrt{A}+\sqrt{B}\le\sqrt{2(A+B)}$ for all nonnegative real numbers A,B (which holds by the concavity of the square root function):

$$R_r(H_p)\le\sqrt{\frac{2}{n}\min_{0\le h\le n}\Big(rh+D^2M^{\frac{2}{p^*}-1}\sum_{j=h+1}^{\infty}\lambda_j\Big)}=\sqrt{\frac{2}{n}\sum_{j=1}^{\infty}\min(r,D^2M^{\frac{2}{p^*}-1}\lambda_j)}.$$

4. Lower Bound

In this subsection we investigate the tightness of our bound on the local Rademacher complexity of H_p. To derive a lower bound we consider the particular case where variables $\boldsymbol{x}^{(1)},\ldots,\boldsymbol{x}^{(M)}$ are i.i.d. For example, this happens if the original input space $\mathcal{X}$ is $\mathbb{R}^M$, the original input variable $X \in \mathcal{X}$ has i.i.d. coordinates, and the kernels $k_1,\ldots,k_M$ are identical and each act on a different coordinate of X.

Lemma 10. *Assume that the variables $\boldsymbol{x}^{(1)},\ldots,\boldsymbol{x}^{(M)}$ are centered and identically independently distributed. Then, the following lower bound holds for the local Rademacher complexity of H_p for any $p \geq 1$:*

$$R_r(H_{p,D,M}) \;\geq\; R_{rM}(H_{1,DM^{1/p^*},1}).$$

Proof First note that since the $\boldsymbol{x}^{(i)}$ are centered and uncorrelated, that

$$Pf_{\boldsymbol{w}}^2 = \Big(\sum_{m=1}^{M} \big\langle \boldsymbol{w}_m, \boldsymbol{x}^{(m)} \big\rangle\Big)^2 = \sum_{m=1}^{M} \big\langle \boldsymbol{w}_m, \boldsymbol{x}^{(m)} \big\rangle^2.$$

Now it follows

$$\begin{aligned}
R_r(H_{p,D,M}) &= \mathbb{E} \sup_{\substack{\boldsymbol{w}\colon \\ \substack{Pf_{\boldsymbol{w}}^2 \leq r \\ \|\boldsymbol{w}\|_{2,p} \leq D}}} \Big\langle \boldsymbol{w}, \frac{1}{n}\sum_{i=1}^{n} \sigma_i \boldsymbol{x}_i \Big\rangle \\
&= \mathbb{E} \sup_{\substack{\boldsymbol{w}\colon \\ \substack{\sum_{m=1}^{M} \langle \boldsymbol{w}^{(m)}, \boldsymbol{x}^{(m)} \rangle^2 \leq r \\ \|\boldsymbol{w}\|_{2,p} \leq D}}} \Big\langle \boldsymbol{w}, \frac{1}{n}\sum_{i=1}^{n} \sigma_i \boldsymbol{x}_i \Big\rangle \\
&\geq \mathbb{E} \sup_{\substack{\boldsymbol{w}\colon \\ \substack{\forall m\colon \langle \boldsymbol{w}^{(m)}, \boldsymbol{x}^{(m)} \rangle^2 \leq r/M \\ \left\|\boldsymbol{w}^{(m)}\right\|_{2,p} \leq D \\ \left\|\boldsymbol{w}^{(1)}\right\| = \cdots = \left\|\boldsymbol{w}^{(M)}\right\|}}} \Big\langle \boldsymbol{w}, \frac{1}{n}\sum_{i=1}^{n} \sigma_i \boldsymbol{x}_i \Big\rangle \\
&= \mathbb{E} \sup_{\substack{\boldsymbol{w}\colon \\ \substack{\forall m\colon \langle \boldsymbol{w}^{(m)}, \boldsymbol{x}^{(m)} \rangle^2 \leq r/M \\ \forall m\colon \left\|\boldsymbol{w}^{(m)}\right\|_2 \leq DM^{-\frac{1}{p}}}}} \sum_{m=1}^{M} \Big\langle \boldsymbol{w}^{(m)}, \frac{1}{n}\sum_{i=1}^{n} \sigma_i \boldsymbol{x}_i^{(m)} \Big\rangle \\
&= \sum_{m=1}^{M} \mathbb{E} \sup_{\substack{\boldsymbol{w}^{(m)}\colon \\ \substack{\langle \boldsymbol{w}^{(m)}, \boldsymbol{x}^{(m)} \rangle^2 \leq r/M \\ \left\|\boldsymbol{w}^{(m)}\right\|_2 \leq DM^{-\frac{1}{p}}}}} \Big\langle \boldsymbol{w}^{(m)}, \frac{1}{n}\sum_{i=1}^{n} \sigma_i \boldsymbol{x}_i^{(m)} \Big\rangle \;,
\end{aligned}$$

so that we can use the i.i.d. assumption on $\boldsymbol{x}^{(m)}$ to equivalently rewrite the last term as follows:

$$\begin{aligned} R_r(H_{p,D,M}) &\overset{\boldsymbol{x}^{(m)}\text{ i.i.d.}}{\geq} \mathbb{E} \sup_{\boldsymbol{w}^{(1)}:\ \substack{\langle \boldsymbol{w}^{(1)}, \boldsymbol{x}^{(1)}\rangle^2 \leq r/M \\ \left\|\boldsymbol{w}^{(1)}\right\|_2 \leq DM^{-\frac{1}{p}}}} \left\langle M\boldsymbol{w}^{(1)}, \frac{1}{n}\sum_{i=1}^n \sigma_i \boldsymbol{x}_i^{(1)} \right\rangle \\ &= \mathbb{E} \sup_{\boldsymbol{w}^{(1)}:\ \substack{\langle M\boldsymbol{w}^{(1)}, \boldsymbol{x}^{(1)}\rangle^2 \leq rM \\ \left\|M\boldsymbol{w}^{(1)}\right\|_2 \leq DM^{\frac{1}{p^*}}}} \left\langle M\boldsymbol{w}^{(1)}, \frac{1}{n}\sum_{i=1}^n \sigma_i \boldsymbol{x}_i^{(1)} \right\rangle \\ &= \mathbb{E} \sup_{\boldsymbol{w}^{(1)}:\ \substack{\langle \boldsymbol{w}^{(1)}, \boldsymbol{x}^{(1)}\rangle^2 \leq rM \\ \left\|\boldsymbol{w}^{(1)}\right\|_2 \leq DM^{\frac{1}{p^*}}}} \left\langle \boldsymbol{w}^{(1)}, \frac{1}{n}\sum_{i=1}^n \sigma_i \boldsymbol{x}_i^{(1)} \right\rangle \\ &= R_{rM}(H_{1,DM^{1/p^*},1}) \end{aligned}$$

■

In Mendelson (2003) it was shown that there is an absolute constant c so that if $\lambda^{(1)} \geq \frac{1}{n}$ then for all $r \geq \frac{1}{n}$ it holds $R_r(H_{1,1,1}) \geq \sqrt{\frac{c}{n}\sum_{j=1}^\infty \min(r, \lambda_j^{(1)})}$. Closer inspection of the proof reveals that more generally it holds $R_r(H_{1,D,1}) \geq \sqrt{\frac{c}{n}\sum_{j=1}^\infty \min(r, D^2\lambda_j^{(1)})}$ if $\lambda_1^{(m)} \geq \frac{1}{nD^2}$ so that we can use that result together with the previous lemma to obtain:

Theorem 11 (Lower bound)**.** *Assume that the kernels are centered and identically independently distributed. Then, the following lower bound holds for the local Rademacher complexity of H_p. There is an absolute constant c such that if $\lambda^{(1)} \geq \frac{1}{nD^2}$ then for all $r \geq \frac{1}{n}$ and $p \geq 1$,*

$$R_r(H_{p,D,M}) \ \geq\ \sqrt{\frac{c}{n}\sum_{j=1}^\infty \min(rM, D^2M^{2/p^*}\lambda_j^{(1)})}. \tag{23}$$

We would like to compare the above lower bound with the upper bound of Theorem 5. To this end note that for centered identical independent kernels the upper bound reads

$$R_r(H_p) \leq \sqrt{\frac{16}{n}\sum_{j=1}^\infty \min\left(rM, ceD^2p^{*2}M^{\frac{2}{p^*}}\lambda_j^{(1)}\right)} + \frac{\sqrt{Be}DM^{\frac{1}{p^*}}p^*}{n},$$

which is of the order $O\big(\sqrt{\sum_{j=1}^\infty \min\left(rM, D^2M^{\frac{2}{p^*}}\lambda_j^{(1)}\right)}\big)$ and, disregarding the quickly converging term on the right hand side and absolute constants, again matches the upper bounds of the previous section. A similar comparison can be performed for the upper bound of Theorem 6: by Remark 8 the bound reads

$$R_r(H_p) \leq \sqrt{\frac{2}{n}\left\|\Big(\sum_{j=1}^\infty \min(r, D^2M^{\frac{2}{p^*}-1}\lambda_j^{(m)})\Big)_{m=1}^M\right\|_1},$$

which for i.i.d. kernels becomes $\sqrt{2/n\sum_{j=1}^\infty \min\left(rM, D^2M^{\frac{2}{p^*}}\lambda_j^{(1)}\right)}$ and thus, besides absolute constants, matches the lower bound. This shows that the upper bounds of the previous section are tight.

5. Excess Risk Bounds

In this section we show an application of our results to prediction problems, such as classification or regression. To this aim, in addition to the data $\boldsymbol{x}_1,\ldots,\boldsymbol{x}_n$ introduced earlier in this paper, let also a label sequence $y_1,\ldots,y_n \subset [-1,1]$ be given that is i.i.d. generated from a probability distribution. The goal in statistical learning is to find a hypothesis f from a pregiven class $\mathcal{F}$ that minimizes the expected loss $\mathbb{E}\, l(f(\boldsymbol{x}),y)$, where $l : \mathbb{R}^2 \mapsto [-1,1]$ is a predefined loss function that encodes the objective of the given learning/prediction task at hand. For example, the hinge loss $l(t,y) = \max(0,1-yt)$ and the squared loss $l(t,y) = (t-y)^2$ are frequently used in classification and regression problems, respectively.

Since the distribution generating the example/label pairs is unknown, the optimal decision function

$$f^* := \operatorname*{argmin}_{f\in\mathcal{F}} \mathbb{E}\, l(f(\boldsymbol{x}),y)$$

can not be computed directly and a frequently used method consists of instead minimizing the *empirical* loss,

$$\hat{f} := \operatorname*{argmin}_{f\in\mathcal{F}} \frac{1}{n}\sum_{i=1}^{n} l(f(\boldsymbol{x}_i),y_i).$$

In order to evaluate the performance of this so-called *empirical risk minimization* (ERM) algorithm we study the excess loss,

$$P(l_{\hat{f}} - l_{f^*}) \;:=\; \mathbb{E}\, l(\hat{f}(\boldsymbol{x}),y) - \mathbb{E}\, l(f^*(\boldsymbol{x}),y).$$

In Bartlett et al. (2005) and Koltchinskii (2006) it was shown that the rate of convergence of the excess risk is basically determined by the fixed point of the local Rademacher complexity. For example, the following result is a slight modification of Corollary 5.3 in Bartlett et al. (2005) that is well-tailored to the class studied in this paper.[3]

Lemma 12. *Let $\mathcal{F}$ be an absolute convex class ranging in the interval $[a,b]$ and let l be a Lipschitz continuous loss with constant L. Assume there is a positive constant F such that*

$$\forall f \in \mathcal{F} :\; P(f-f^*)^2 \le F\, P(l_f - l_{f^*}). \tag{24}$$

Then, denoting by r^* the fixed point of

$$2FL\, R_{\frac{r}{4L^2}}(\mathcal{F})$$

for all $x > 0$ with probability at least $1-e^{-x}$ the excess loss can be bounded as

$$P(l_{\hat{f}} - l_{f^*}) \le 7\frac{r^*}{F} + \frac{(11L(b-a)+27F)x}{n}.$$

Note that condition (24) on the loss function is fulfilled, for example, when the kernel is uniformly bounded and the loss function is strongly convex and Lipschitz continuous on the domain considered (Bartlett et al., 2006). This includes, for example, the squared loss as defined above, the

3. We exploit the improved constants from Theorem 3.3 in Bartlett et al. (2005) because an absolute convex class is star-shaped. Compared to Corollary 5.3 in Bartlett et al. (2005) we also use a slightly more general function class ranging in $[a,b]$ instead of the interval $[-1,1]$. This is also justified by Theorem 3.3.

logistic loss $l(t,y) = \ln(1+\exp(-yt))$, and the exponential loss $l(t,y) = \exp(-yt)$. The case of the hinge loss (see definition above) is more delicate, since it is not a strongly convex loss function. In general, the hinge loss does not satisfy (24) on an arbitrary convex class $\mathcal{F}$; for this reason, there is no direct, general "fast rate" excess loss analogue to the popular margin-radius bounds obtained through global Rademacher analysis. Nevertheless, local Rademacher complexity analysis can still be put to good use for algorithms based on the hinge loss. In fact, the hinge loss satisfies, under an additional "noise exponent condition" assumption, a restricted version of (24), namely, when f^* is taken equal to the Bayes classifier. This can be used to study theoretically the behavior of penalized ERM methods such as the support vector machine, and more precisely to obtain oracle-type inequalities (this roughly means that the penalized ERM can be shown to pick a correct trade-off of bias and estimation error, leading to fast convergence rates). In this sense, the local Rademacher complexity bound we have presented here can in principle be plugged in into the SVM analysis of Blanchard et al. (2008), directly replacing the local Rademacher analysis for a single kernel studied there under setting (S1); see also Steinwart and Christmann (2008, Chapter 8) for a comparable analysis. This more elaborate analysis does, however, not fall directly into the scope of the comparably simpler result of Lemma 12, which considers simple ERM over a fixed model, so that we refer the reader to the references cited above for more details.

Lemma 12 shows that in order to obtain an excess risk bound on the multi-kernel class H_p it suffices to compute the fixed point of our bound on the local Rademacher complexity presented in Section 3. To this end we show:

Lemma 13. *Assume that $\|k\|_\infty \le B$ almost surely and assumption (A) holds; let $p \in [1,2]$. For the fixed point r^* of the local Rademacher complexity $2FLR_{\frac{r}{4L^2}}(H_p)$ it holds*

$$r^* \le \min_{0\le h_m\le\infty} \frac{4c_\delta^{-1}F^2\sum_{m=1}^M h_m}{n} + 8FL\sqrt{\frac{ep^{*2}D^2}{n}\left\|\left(\sum_{j=h_m+1}^{\infty}\lambda_j^{(m)}\right)_{m=1}^M\right\|_{\frac{p^*}{2}}} + \frac{4\sqrt{Be}DFLM^{\frac{1}{p^*}}p^*}{n}\,.$$

Proof For this proof we make use of the bound (17) on the local Rademacher complexity. Defining

$$a = \frac{4c_\delta^{-1}F^2\sum_{m=1}^M h_m}{n} \quad \text{and} \quad b = 4FL\sqrt{\frac{ep^{*2}D^2}{n}\left\|\left(\sum_{j=h_m+1}^{\infty}\lambda_j^{(m)}\right)_{m=1}^M\right\|_{\frac{p^*}{2}}} + \frac{2\sqrt{Be}DFLM^{\frac{1}{p^*}}p^*}{n}\,,$$

in order to find a fixed point of (17) we need to solve for $r = \sqrt{ar}+b$, which is equivalent to solving $r^2-(a+2b)r+b^2=0$ for a positive root. Denote this solution by r^*. It is then easy to see that $r^* \ge a+2b$. Resubstituting the definitions of a and b yields the result. ■

We now address the issue of computing actual rates of convergence of the fixed point r^* under the assumption of algebraically decreasing eigenvalues of the kernel matrices, this means, we assume $\exists d_m:\ \lambda_j^{(m)} \le d_m j^{-\alpha_m}$ for some $\alpha_m > 1$. This is a common assumption and, for example, met for finite rank kernels and convolution kernels (Williamson et al., 2001). Notice that this implies

$$\begin{aligned}\sum_{j=h_m+1}^{\infty}\lambda_j^{(m)} &\le d_m\sum_{j=h_m+1}^{\infty} j^{-\alpha_m} \le d_m\int_{h_m}^{\infty}x^{-\alpha_m}dx = d_m\left[\frac{1}{1-\alpha_m}x^{1-\alpha_m}\right]_{h_m}^{\infty}\\ &= -\frac{d_m}{1-\alpha_m}h_m^{1-\alpha_m}\,. \qquad (25)\end{aligned}$$

To exploit the above fact, first note that by ℓ_p-to-ℓ_q conversion

$$\frac{4c_\delta^{-1}F^2\sum_{m=1}^M h_m}{n} \;\le\; 4F\sqrt{\frac{c_\delta^{-1}F^2M\sum_{m=1}^M h_m^2}{n^2}} \;\le\; 4F\sqrt{\frac{c_\delta^{-1}F^2M^{2-\frac{2}{p^*}}\left\|\left(h_m^2\right)\right)_{m=1}^M\right\|_{2/p^*}}{n^2}}$$

so that we can translate the result of the previous lemma by (18), (19), and (20) into

$$\begin{aligned} r^* \;\le\; \min_{0\le h_m\le\infty} 8F\sqrt{\frac{1}{n}\left\|\left(\frac{c_\delta^{-1}F^2M^{2-\frac{2}{p^*}}h_m^2}{n}+4ep^{*2}D^2L^2\sum_{j=h_m+1}^{\infty}\lambda_j^{(m)}\right)_{m=1}^M\right\|_{\frac{p^*}{2}}} \\ +\frac{4\sqrt{Be}DFLM^{\frac{1}{p^*}}p^*}{n}\,. \end{aligned} \tag{26}$$

Inserting the result of (25) into the above bound and setting the derivative with respect to h_m to zero we find the optimal h_m as

$$h_m = \left(4c_\delta d_m e p^{*2}D^2F^{-2}L^2M^{\frac{2}{p^*}-2}n\right)^{\frac{1}{1+\alpha_m}}.$$

Resubstituting the above into (26) we note that

$$r^* = O\left(\sqrt{\left\|\left(n^{-\frac{2\alpha_m}{1+\alpha_m}}\right)_{m=1}^M\right\|_{\frac{p^*}{2}}}\right)$$

so that we observe that the asymptotic rate of convergence in n is determined by the kernel with the smallest decreasing spectrum (i.e., smallest α_m). Denoting $d_{\max} := \max_{m=1,\ldots,M} d_m$, $\alpha_{\min} := \min_{m=1,\ldots,M}\alpha_m$, and $h_{\max} := \left(4c_\delta d_{\max} e p^{*2}D^2F^{-2}L^2M^{\frac{2}{p^*}-2}n\right)^{\frac{1}{1+\alpha_{\min}}}$ we can upper-bound (26) by

$$\begin{aligned} r^* &\le 8F\sqrt{\frac{3-\alpha_{\min}}{1-\alpha_{\min}}c_\delta^{-1}F^2M^2h_{\max}^2n^{-2}}+\frac{4\sqrt{Be}DFLM^{\frac{1}{p^*}}p^*}{n} \\ &\le 8\sqrt{\frac{3-\alpha_{\min}}{1-\alpha_{\min}}}c_\delta^{-1}F^2Mh_{\max}n^{-1}+\frac{4\sqrt{Be}DFLM^{\frac{1}{p^*}}p^*}{n} \\ &\le 16\sqrt{e\frac{3-\alpha_{\min}}{1-\alpha_{\min}}}c_\delta^{-1}(d_{\max}D^2L^2p^{*2})^{\frac{1}{1+\alpha_{\min}}}F^{\frac{2\alpha_{\min}}{1+\alpha_{\min}}}M^{1+\frac{2}{1+\alpha_{\min}}\left(\frac{1}{p^*}-1\right)}n^{-\frac{\alpha_{\min}}{1+\alpha_{\min}}} \\ &\quad+\frac{4\sqrt{Be}DFLM^{\frac{1}{p^*}}p^*}{n}\,. \end{aligned} \tag{27}$$

We have thus proved the following theorem, which follows by the above inequality, Lemma 12, and the fact that our class H_p ranges in $BDM^{\frac{1}{p^*}}$.

Theorem 14. *Assume that $\|k\|_\infty \le B$, assumption (A) holds, and it $\exists d_{\max} > 0$ and $\alpha := \alpha_{\min} > 1$ such that for all $m = 1,\ldots,M$ it holds $\lambda_j^{(m)} \le d_{\max} j^{-\alpha}$. Let l be a Lipschitz continuous loss with constant L and assume there is a positive constant F such that $\forall f \in \mathcal{F}:\ P(f-f^*)^2 \le F\,P(l_f - l_{f^*})$.*

Then for all $x > 0$ with probability at least $1 - e^{-x}$ the excess loss of the multi-kernel class H_p can be bounded for $p \in [1, \ldots, 2]$ as

$$P(l_{\hat{f}} - l_{f^*}) \leq \min_{t \in [p,2]} 186 \sqrt{\frac{3-\alpha}{1-\alpha}} c_\delta^{\frac{1-\alpha}{1+\alpha}} \left(d_{\max} D^2 L^2 t^{*2}\right)^{\frac{1}{1+\alpha}} F^{\frac{\alpha-1}{\alpha+1}} M^{1+\frac{2}{1+\alpha}\left(\frac{1}{t^*}-1\right)} n^{-\frac{\alpha}{1+\alpha}} + \frac{47\sqrt{B}DLM^{\frac{1}{t^*}} t^*}{n} + \frac{(22BDLM^{\frac{1}{t^*}} + 27F)x}{n}$$

We see from the above bound that convergence can be almost as slow as $O\left(p^* M^{\frac{1}{p^*}} n^{-\frac{1}{2}}\right)$ (if at least one $\alpha_m \approx 1$ is small and thus $\alpha_{\min}$ is small) and almost as fast as $O\left(n^{-1}\right)$ (if α_m is large for all m and thus $\alpha_{\min}$ is large). For example, the latter is the case if all kernels have finite rank and also the convolution kernel is an example of this type.

Notice that we of course could repeat the above discussion to obtain excess risk bounds for the case $p \geq 2$ as well, but since it is very questionable that this will lead to new insights, it is omitted for simplicity.

6. Discussion

In this section we compare the obtained local Rademacher bound with the global one, discuss related work as well as the assumption **(A)**, and give a practical application of the bounds by studying the appropriateness of small/large p in various learning scenarios.

6.1 Global vs. Local Rademacher Bounds

In this section, we discuss the rates obtained from the bound in Theorem 14 for the excess risk and compare them to the rates obtained using the global Rademacher complexity bound of Corollary 4. To simplify somewhat the discussion, we assume that the eigenvalues satisfy $\lambda_j^{(m)} \leq d j^{-\alpha}$ (with $\alpha > 1$) for all m and concentrate on the rates obtained as a function of the parameters n, α, M, D and p, while considering other parameters fixed and hiding them in a big-O notation. Using this simplification, the bound of Theorem 14 reads

$$\forall t \in [p,2]: \quad P(l_{\hat{f}} - l_{f^*}) = O\left(\left(t^* D\right)^{\frac{2}{1+\alpha}} M^{1+\frac{2}{1+\alpha}\left(\frac{1}{t^*}-1\right)} n^{-\frac{\alpha}{1+\alpha}}\right) \tag{28}$$

$\left(\text{and } P(l_{\hat{f}} - l_{f^*}) = O\left(\left(D \log M\right)^{\frac{2}{1+\alpha}} M^{\frac{\alpha-1}{\alpha+1}}\right) \text{ for } p = 1\right)$. On the other hand, the global Rademacher complexity directly leads to a bound on the supremum of the centered empirical process indexed by $\mathcal{F}$ and thus also provides a bound on the excess risk (see, e.g., Bousquet et al., 2004). Therefore, using Corollary 4, wherein we upper bound the trace of each J_m by the constant B (and subsume it under the O-notation), we have a second bound on the excess risk of the form

$$\forall t \in [p,2]: \quad P(l_{\hat{f}} - l_{f^*}) = O\left(t^* D M^{\frac{1}{t^*}} n^{-\frac{1}{2}}\right). \tag{29}$$

First consider the case where $p \geq (\log M)^*$, that is, the best choice in (28) and (29) is $t = p$. Clearly, if we hold all other parameters fixed and let n grow to infinity, the rate obtained through the local Rademacher analysis is better since $\alpha > 1$. However, it is also of interest to consider what happens when the number of kernels M and the ℓ_p ball radius D can grow with n. In general, we have a bound

on the excess risk given by the minimum of (28) and (29); a straightforward calculation shows that the local Rademacher analysis improves over the global one whenever

$$\frac{M^{\frac{1}{p}}}{D} = O(\sqrt{n}).$$

Interestingly, we note that this "phase transition" does not depend on α (i.e., the "complexity" of the individual kernels), but only on p.

If $p \leq (\log M)^*$, the best choice in (28) and (29) is $t = (\log M)^*$. In this case taking the minimum of the two bounds reads

$$\forall p \leq (\log M)^* : \quad P(l_{\hat{f}} - l_{f^*}) \leq O\Big(\min(D(\log M)n^{-\frac{1}{2}}, \big(D\log M\big)^{\frac{2}{1+\alpha}} M^{\frac{\alpha-1}{1+\alpha}} n^{-\frac{\alpha}{1+\alpha}}) \Big), \tag{30}$$

and the phase transition when the local Rademacher bound improves over the global one occurs for

$$\frac{M}{D\log M} = O(\sqrt{n}).$$

Finally, it is also interesting to observe the behavior of (28) and (29) as $\alpha \to \infty$. In this case, it means that only one eigenvalue is nonzero for each kernel, that is, each kernel space is one-dimensional. In other words, in this case we are in the case of "classical" aggregation of M basis functions, and the minimum of the two bounds reads

$$\forall t \in [p,2] : \quad P(l_{\hat{f}} - l_{f^*}) \leq O\Big(\min(Mn^{-1}, t^* D M^{\frac{1}{t^*}} n^{-\frac{1}{2}} \Big). \tag{31}$$

In this configuration, observe that the local Rademacher bound is $O(M/n)$ and does not depend on D, nor p, any longer; in fact, it is the same bound that one would obtain for the empirical risk minimization over the space of all linear combinations of the M base functions, without any restriction on the norm of the coefficients—the ℓ_p-norm constraint becomes void. The global Rademacher bound on the other hand, still depends crucially on the ℓ_p norm constraint. This situation is to be compared to the sharp analysis of the optimal convergence rate of convex aggregation of M functions obtained by Tsybakov (2003) in the framework of squared error loss regression, which are shown to be

$$O\left(\min\left(\frac{M}{n}, \sqrt{\frac{1}{n}\log\left(\frac{M}{\sqrt{n}}\right)} \right) \right).$$

This corresponds to the setting studied here with $D = 1, p = 1$ and $\alpha \to \infty$, and we see that the bound (30) recovers (up to log factors) in this case this sharp bound and the related phase transition phenomenon.

6.2 Discussion of Related Work

We recently learned about independent, closely related work by Suzuki (2011), which has been developed in parallel to ours. The setup considered there somewhat differs from ours: first of all, it is required that the Bayes hypothesis is contained in the class $\boldsymbol{w}^* \in H$ (which is not required in the present work); second, the conditional distribution is assumed to be expressible in terms of the Bayes hypothesis. Similar assumptions are also required in Bach (2008) in the context of sparse recovery. Finally, the analysis there is carried out for the squared loss only, while ours holds more

generally for, for example, strongly convex Lipschitz losses. However, a similarity to our setup is that an algebraic decay of the eigenvalues of the kernel matrices is assumed for the computation of the excess risk bounds and that a so-called incoherence assumption is imposed on the kernels, which is similar to our Assumption (A). Also, we do not spell out the whole analysis for inhomogeneous eigenvalue decays as Suzuki (2011) does—nevertheless, our analysis can be easily adapted to this case at the expense of longer, less-readable bounds.

We now compare the excess risk bounds of Suzuki (2011) for the case of homogeneous eigenvalue decays, that is,

$$P(l_{\hat{f}} - l_{f^*}) = O\Big((D)^{\frac{2}{1+\alpha}} M^{1+\frac{2}{1+\alpha}\left(\frac{1}{p^*}-1\right)} n^{-\frac{\alpha}{1+\alpha}} \Big),$$

to the ones shown in this paper, that is, (28)—we thereby disregard constants and the $O(n^{-1})$ terms. Roughly speaking, the proof idea in Suzuki (2011) is to exploit existing bounds on the LRC of single-kernel learning (Steinwart and Christmann, 2008) by combining Talagrand's inequality (Talagrand, 1995) and the peeling technique (van de Geer, 2000). This way the Khintchine-Kahane, which introduces a factor of $(p^*)^{\frac{2}{1+\alpha}}$ into our bounds, is avoided.

We observe that, importantly, both bounds have the same dependency in D, M, and n, although being derived by a completely different technique. Regarding the dependency in p, we observe that our bound involves a factor of $(t^*)^{\frac{2}{1+\alpha}}$ (for some $t \in [p,2]$ that is not present in the bound of Suzuki (2011). However, it can be easily shown that this factor is never of higher order than $\log(M)$ and thus can be neglected:

1. If $p \leq (\log(M))^*$, then $t = \log(M)$ is optimal in our bound so that the term $(t^*)^{\frac{2}{1+\alpha}}$ becomes $(\log(M))^{\frac{2}{1+\alpha}}$.

2. If $p \geq (\log(M))^*$, then $p^* \leq \log(M)$ so that the term $(t^*)^{\frac{2}{1+\alpha}}$ is smaller equal than $(\log(M))^{\frac{2}{1+\alpha}}$.

We can thus conclude that, besides a logarithmic factor in M as well as constants and $O(n^{-1})$ terms, our bound coincides with the rate shown in Suzuki (2011).

6.3 Discussion of Assumption (A)

Assumption **(A)** is arguably quite a strong hypothesis for the validity of our results (needed for $1 \leq p \leq 2$), which was not required for the global Rademacher bound. A similar assumption is also made in the recent works of Suzuki (2011) and Koltchinskii and Yuan (2010). In the latter paper, a related MKL algorithm using a mixture of an ℓ_1-type penalty and an empirical ℓ_2 penalty is studied (this should not be confused with $\ell_{p=1}$-norm MKL, which does not involve an empirical penalty and which, for $p = 1$, is contained in the ℓ_p-norm MKL methodology studied in this paper). Koltchinskii and Yuan (2010) derive bounds that depend on the "sparsity pattern" of the Bayes function, that is, how many coefficients w_m^* are non-zero, using an Restricted Isometry Property (RIP) assumption. If the kernel spaces are one-dimensional, in which case ℓ_1-penalized MKL reduces qualitatively to standard lasso-type methods, this assumption is known to be necessary to grant the validity of bounds taking into account the sparsity pattern of the Bayes function.[4]

4. We also mention another work by Raskutti et al. (2010), investigating the same algorithm as Koltchinskii and Yuan (2010), but employing a somewhat more restrictive assumption on the uncorrelatedness of the kernels, which corresponds to taking $c_\delta = 1$ in assumption **(A).**

In the present work, our analysis stays deliberately "agnostic" (or worst-case) with respect to the true sparsity pattern (in part because experimental evidence seems to point towards the fact that the Bayes function is not strongly sparse); correspondingly it could legitimately be hoped that the RIP condition, or Assumption **(A)**, could be substantially relaxed. Considering again the special case of one-dimensional kernel spaces and the discussion about the qualitatively equivalent case $\alpha \to \infty$ in the previous section, it can be seen that Assumption **(A)** is indeed unnecessary for bound (31) to hold, and more specifically for the rate of M/n obtained through local Rademacher analysis in this case. However, as we discussed, what happens in this specific case is that the local Rademacher analysis becomes oblivious to the ℓ_p-norm constraint, and we are left with the standard parametric convergence rate in dimension M. In other words, with one-dimensional kernel spaces, the two constraints (on the $L^2(P)$-norm of the function and on the ℓ_p block-norm of the coefficients) appearing in the definition of local Rademacher complexity are essentially not active simultaneously. Unfortunately, it is clear that this property is not true anymore for kernels of higher complexity (i.e., with a non-trivial decay rate of the eigenvalues). This is a specificity of the kernel setting as compared to combinations of a dictionary of M simple functions, and Assumption **(A)** was in effect used to "align" the two constraints. To sum up, Assumption **(A)** is used here for a different purpose from that of the RIP in sparsity analyses of ℓ_1 regularization methods; it is not clear to us at this point if this assumption is necessary or if uncorrelated variables $x^{(m)}$ constitutes a "worst case" for our analysis. We did not suceed so far in relinquishing this assumption for $p \leq 2$, and this question remains open.

Besides the work of Suzuki (2011), there is, up to our knowledge, no previous existing analysis of the ℓ_p-MKL setting for $p > 1$; the recent works of Raskutti et al. (2010) and Koltchinskii and Yuan (2010) focus on the case $p = 1$ and on the sparsity pattern of the Bayes function. A refined analysis of ℓ_p-regularized methods in the case of combination of M basis functions was laid out by Koltchinskii (2009), also taking into account the possible soft sparsity pattern of the Bayes function. Extending the ideas underlying the latter analysis into the kernel setting is likely to open interesting developments.

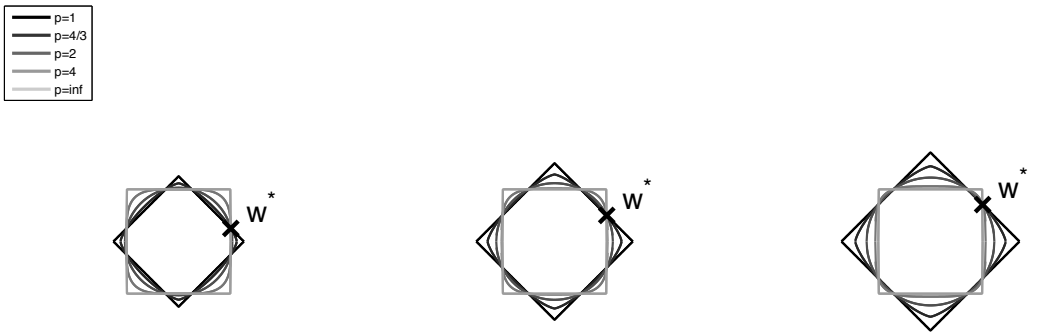
6.4 Analysis of the Impact of the Norm Parameter p on the Accuracy of ℓ_p-norm MKL

As outlined in the introduction, there is empirical evidence that the performance of ℓ_p-norm MKL crucially depends on the choice of the norm parameter p (cf. Figure 1 in the introduction). The aim of this section is to relate the theoretical analysis presented here to this empirically observed phenomenon. We believe that this phenomenon can be (at least partly) explained on base of our excess risk bound obtained in the last section. To this end we will analyze the dependency of the excess risk bounds on the chosen norm parameter p. We will show that the optimal p depends on the geometrical properties of the learning problem and that in general—depending on the true geometry—any p can be optimal. Since our excess risk bound is only formulated for $p \leq 2$, we will limit the analysis to the range $p \in [1,2]$.

To start with, first note that the choice of p only affects the excess risk bound in the factor (cf. Theorem 14 and Equation (28))

$$\nu_t := \min_{t \in [p,2]} \left(D_p t^*\right)^{\frac{2}{1+\alpha}} M^{1+\frac{2}{1+\alpha}\left(\frac{1}{t^*}-1\right)}.$$

So we write the excess risk as $P(l_{\hat{f}} - l_{f^*}) = O(\nu_t)$ and hide all variables and constants in the O-notation for the whole section (in particular the sample size n is considered a constant for the pur-



(a) $\beta = 2$ (b) $\beta = 1$ (c) $\beta = 0.5$

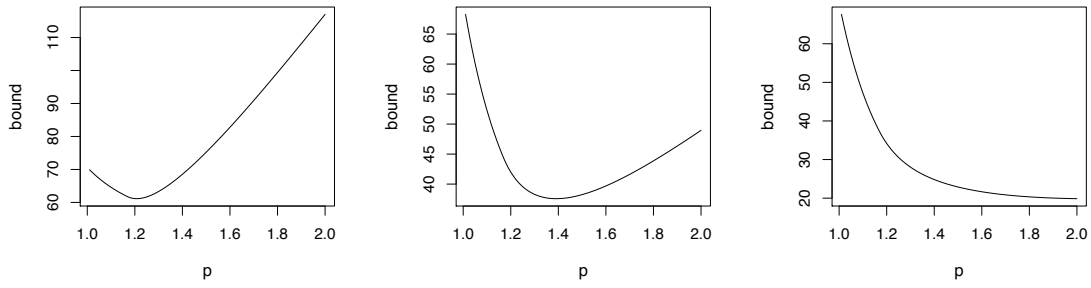
Figure 2: 2D-Illustration of the three learning scenarios analyzed in this section: LEFT: A soft sparse $\boldsymbol{w}^*$; CENTER: an intermediate non-sparse $\boldsymbol{w}^*$; RIGHT: an almost-uniformly non-sparse $\boldsymbol{w}^*$. Each scenario has a Bayes hypothesis $\boldsymbol{w}^*$ with a different soft sparsity (parametrized by β). The colored lines show the smallest ℓ_p-ball containing the Bayes hypothesis. We observe that the radii of the hypothesis classes depend on the sparsity of $\boldsymbol{w}^*$ and the parameter p.

poses of the present discussion). It might surprise the reader that we consider the term in D in the bound although it seems from the bound that it does not depend on p. This stems from a subtle reason that we have ignored in this analysis so far: D is related to the approximation properties of the class, that is, its ability to attain the Bayes hypothesis. For a "fair" analysis we should take the approximation properties of the class into account.

To illustrate this, let us assume that the Bayes hypothesis belongs to the space $\mathcal{H}$ and can be represented by $\boldsymbol{w}^*$; assume further that the block components satisfy $\|\boldsymbol{w}_m^*\|_2 = m^{-\beta}$, $m = 1,\ldots,M$, where $\beta \geq 0$ is a parameter parameterizing the "soft sparsity" of the components. For example, the cases $\beta \in \{0.5, 1, 2\}$ are shown in Figure 2 for $M = 2$ and assuming that each kernel has rank 1 (thus being isomorphic to $\mathbb{R}$). If n is large, the best bias-complexity tradeoff for a fixed p will correspond to a vanishing bias, so that the best choice of D will be close to the minimal value such that $\boldsymbol{w}^* \in H_{p,D}$, that is, $D_p = ||\boldsymbol{w}^*||_p$. Plugging in this value for D_p, the bound factor ν_p becomes

$$\nu_p := \|\boldsymbol{w}^*\|_p^{\frac{2}{1+\alpha}} \min_{t \in [p,2]} t^{* \frac{2}{1+\alpha}} M^{1+\frac{2}{1+\alpha}\left(\frac{1}{t^*}-1\right)} . \tag{32}$$

We can now plot the value ν_p as a function of p for special choices of α, M, and β. We realized this simulation for $\alpha = 2$, $M = 1000$, and $\beta \in \{0.5, 1, 2\}$, which means we generated three learning scenarios with different levels of soft sparsity parametrized by β. The results are shown in Figure 3. Note that the soft sparsity of $\boldsymbol{w}^*$ is increased from the left hand to the right hand side. We observe that in the "soft sparsest" scenario ($\beta = 2$, shown on the left-hand side) the minimum is attained for a quite small $p = 1.2$, while for the intermediate case ($\beta = 1$, shown at the center) $p = 1.4$ is optimal, and finally in the uniformly non-sparse scenario ($\beta = 2$, shown on the right-hand side) the choice of $p = 2$ is optimal (although even a higher p could be optimal, but our bound is only valid for $p \in [1,2]$).



(a) $\beta = 2$ (b) $\beta = 1$ (c) $\beta = 0.5$

Figure 3: Results of the simulation for the three analyzed learning scenarios (which were illustrated in Figure 2). The value of the bound factor ν_t is plotted as a function of p. The minimum is attained depending on the true soft sparsity of the Bayes hypothesis $\boldsymbol{w}^*$ (parametrized by β).

This means that if the true Bayes hypothesis has an intermediately dense representation, our bound gives the strongest generalization guarantees to ℓ_p-norm MKL using an intermediate choice of p. This is also intuitive: if the truth exhibits some soft sparsity but is not strongly sparse, we expect non-sparse MKL to perform better than strongly sparse MKL or the unweighted-sum kernel SVM.

6.5 An Experiment on Synthetic Data

We now present a toy experiment that is meant to check the validity of the theory presented in the previous sections. To this end, we construct learning scenarios where we know the underlying ground truth (more precisely, the ℓ_p-norm of the Bayes hypothesis) and check whether the parameter p that minimizes our bound coincides with the optimal p observed empirically, that is, when applying ℓ_p-norm MKL to the training data. Our analysis is based on the proven synthetic data described in Kloft et al. (2011) and being available from `http://mldata.org/repository/data/viewslug/mkl-toy/`. For completeness, we summarize the experimental description and the empirical results here. Note that we have extended the analysis to the whole range $p \in [1,\infty]$ (only $p \in [1,2]$ was studied in Kloft et al., 2011).

6.5.1 EXPERIMENTAL SETUP AND EMPIRICAL RESULTS

We construct six artificial data sets as described in Kloft et al. (2011), in which we vary the degree of sparsity of the true Bayes hypothesis $\boldsymbol{w}$. For each data set, we generate an $n = 50$-element, balanced sample $\mathcal{D} = \{(\boldsymbol{x}_i, y_i)\}_{i=1}^n$ from two $d = 50$-dimensional isotropic Gaussian distributions with equal covariance matrices $C = I_{d\times d}$ and equal, but opposite, means $\mu_+ = \frac{\rho}{\|\boldsymbol{w}\|_2}\boldsymbol{w}$ and $\mu_- = -\mu_+$. Figure 4 shows bar plots of the $\boldsymbol{w}$ of the various scenarios considered. The components w_i are binary valued; hence, the fraction of zero components, which we define by $\mathit{sparsity}(\boldsymbol{w}) := 1 - \frac{1}{d}\sum_{i=1}^d w_i$, is a measure for the feature sparsity of the learning problem.

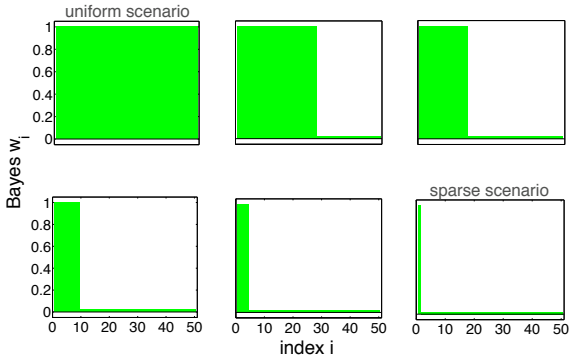


Figure 4: Toy experiment: illustration of the experimental design. We study six scenarios differing the sparsity of the Bayes hypothesis considered.

For each of the w we generate $m = 250$ data sets $\mathcal{D}_1, \ldots, \mathcal{D}_m$ fixing $\rho = 1.75$. Then, each feature is input into a linear kernel and the resulting kernel matrices are multiplicatively normalized as described in Kloft et al. (2011). Next, classification models are computed by training ℓ_p-norm MKL for $p = 1, 4/3, 2, 4, \infty$ on each $\mathcal{D}_i$. Soft margin parameters C are tuned on independent $1,000$-elemental validation sets by grid search over $C \in \left\{ 10^i \,\middle|\, i = -4, -3.5, \ldots, 0 \right\}$ (optimal Cs are attained in the interior of the grid). The relative duality gaps were optimized up to a precision of 10^{-3}. The simulation is realized for $n = 50$. We report on test errors evaluated on $1,000$-elemental independent test sets.

The results in terms of test errors are shown in Figure 5 (top). As expected, ℓ_1-norm MKL performs best and reaches the Bayes error in the sparsest scenario. In contrast, the vanilla SVM using a uniform kernel combination performs best when all kernels are equally informative. The non-sparse $\ell_{4/3}$-norm MKL variants perform best in the balanced scenarios, that is, when the noise level is ranging in the interval 64%-92%. Intuitively, the non-sparse $\ell_{4/3}$-norm MKL is the most robust MKL variant, achieving test errors of less than 12% in all scenarios. Tuning the sparsity parameter p for each experiment, ℓ_p-norm MKL achieves low test error across all scenarios.

6.5.2 BOUND

We evaluate the theoretical bound factor (32) (simply setting $\alpha = 1$) for the six learning scenarios considered. To furthermore analyze whether the p that are minimizing the bound are reflected in the empirical results, we compute the test errors of the various MKL variants again, using the setup above except that we employ a local search for finding the optimal p. The results are shown in Figure 5 (bottom). We observe a striking coincidence of the optimal p as predicted by the bound and the p that worked best empirically: In the sparsest scenario (shown on the lower right-hand side), the bound predicts $p \in [1, 1.14]$ to be optimal and indeed, in the experiments, all $p \in [1, 1.15]$ performed best (and equally well) while $p = 1.19$, already has a slightly (but significantly) worse test error—in striking match with our bounds. In the second sparsest scenario, the bound predicts $p = 1.25$ and we empirically found $p = 1.26$. In the non-sparse scenarios, intermediate values of $p \in [1, 2]$ are optimal (see Figure for details)—again we can observe a good accordance of the

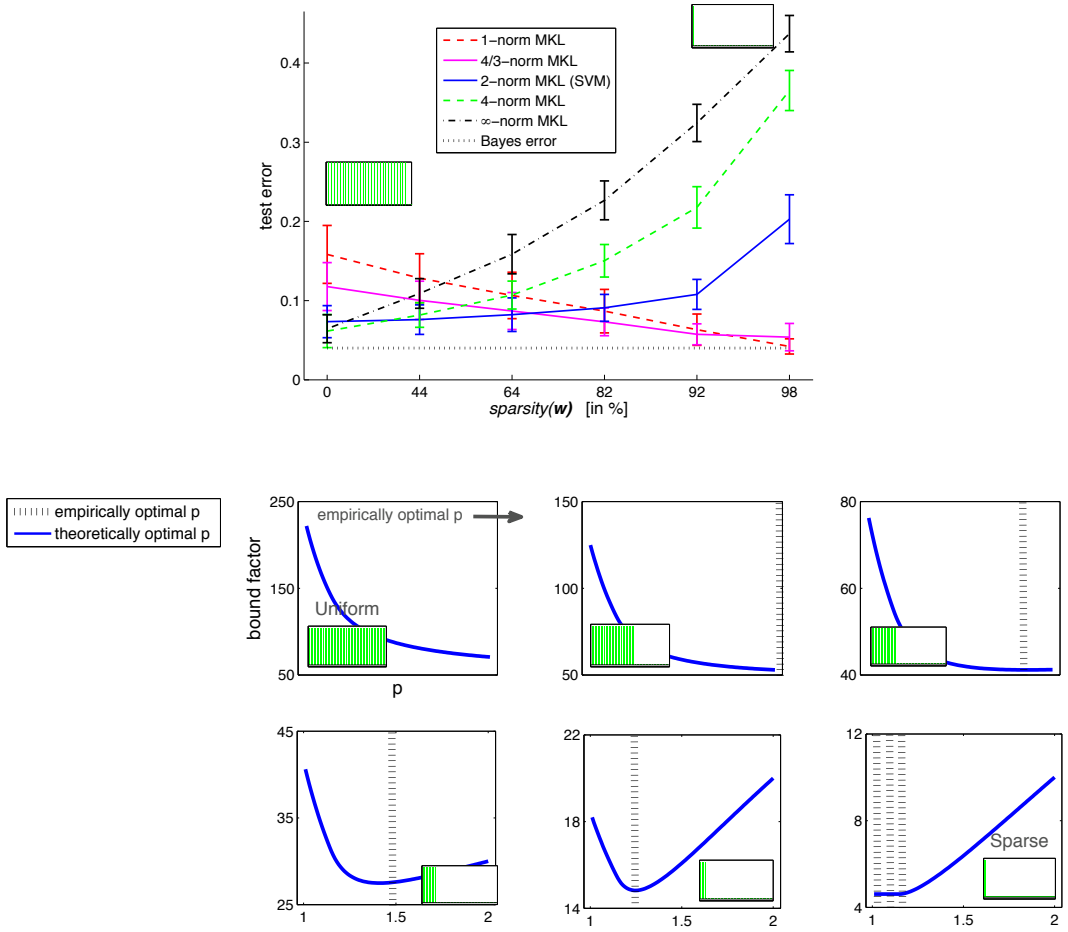


Figure 5: Toy experiment: empirical results (top) and theoretical bounds (bottom).

empirical and theoretical values. In the extreme case, that is, the uniform scenario, the bound indicates a p that lies well beyond the valid interval of the bound (i.e., $p > 2$) and this is also what we observe empirically: $p \in [4, \infty]$ worked best in our experiments.

6.5.3 SUMMARY AND DISCUSSION

We can conclude that the empirical results indicate the validity of our theory: the theoretical bounds reflect the empirically observed optimal p in the idealized setup where we know the underlying ground true, that is, the ℓ_p-norm of the Bayes hypothesis. We also observed that the optimality of a particular p strongly depends on the geometry of the learning task: the sparsity of the underlying Bayes hypothesis $\boldsymbol{w}$. This raises the question into which scenarios practical applications fall. For example, do we rather encounter a "sparse" or non-sparse scenario in bioinformatics? However, this investigation is beyond the scope of this paper (see Chapter 5 in Kloft (2011) for an analysis aiming in that direction).

The results of our analysis are especially surprising, when recalling the result of Suzuki (2011) discussed in Section 6.2. For the setup of homogeneous eigenvalue decay of the kernels as considered in the toy experiment setup here, their bound is optimal for $p = 1$, regardless of the sparsity of the Bayes hypothesis. This is counter-intuitive and in strong contrast to our empirical analysis on synthetic data, where the optimality of a certain value of the norm parameter p crucially depends on the sparsity of the Bayes hypothesis. At this point we have no explanation for this behavior and this leaves an open issue for relating theory to empirical results. The analysis carried out in this paper may serve as a starting point for subsequent analyses aiming in that direction.

7. Conclusion

We derived a sharp upper bound on the local Rademacher complexity of ℓ_p-norm multiple kernel learning under the assumption of uncorrelated kernels. We also proved a lower bound that matches the upper one and shows that our result is tight. Using the local Rademacher complexity bound, we derived an excess risk bound that attains the fast rate of $O(n^{-\frac{\alpha}{1+\alpha}})$, where α is the minimum eigenvalue decay rate of the individual kernels.

In a practical case study, we found that the optimal value of that bound depends on the true Bayes-optimal kernel weights. If the true weights exhibit soft sparsity but are not strongly sparse, then the generalization bound is minimized for an intermediate p. This is not only intuitive but also supports empirical studies showing that sparse MKL ($p = 1$) rarely works in practice, while some intermediate choice of p can improve performance.

Of course, this connection is only valid if the optimal kernel weights are likely to be non-sparse in practice. Indeed, related research points in that direction. For example, already weak connectivity in a causal graphical model may be sufficient for all variables to be required for optimal predictions, and even the prevalence of sparsity in causal flows is being questioned (e.g., for the social sciences Gelman, 2010, argues that "There are (almost) no true zeros").

Finally, we note that there seems to be a certain preference for sparse models in the scientific community. However, previous MKL research has shown that non-sparse models may improve quite impressively over sparse ones in practical applications. The present analysis supports this by showing that the reason for this might be traced back to non-sparse MKL attaining better generalization bounds in non-sparse learning scenarios. We remark that this point of view is also supported by related analyses.

For example, it was shown by Leeb and Pötscher (2008) in a fixed design setup that any sparse estimator (i.e., satisfying the oracle property of correctly predicting the zero values of the true target $\boldsymbol{w}^*$) has a maximal scaled mean squared error (MSMSE) that diverges to ∞. This is somewhat suboptimal since, for example, least-squares regression has a converging MSMSE. Although this is an asymptotic result, it might also be one of the reasons for finding excellent (non-asymptotic) results in non-sparse MKL. In another, recent study of Xu et al. (2008), it was shown that no sparse algorithm can be algorithmically stable. This is noticeable because algorithmic stability is connected with generalization error (Bousquet and Elisseeff, 2002).

Acknowledgments

We greatly thank Peter L. Bartlett, Klaus-Robert Müller, and Taiji Suzuki for their helpful comments on the manuscript.

MK was supported by the German Science Foundation (DFG MU 987/6-1, RA 1894/1-1) and by the World Class University Program through the National Research Foundation of Korea funded by the Korean Ministry of Education, Science, and Technology, under Grant R31-10008. GB was supported by the European Community under the grant agreement 247022 (MASH Project). Both authors were supported by the European Community's 7th Framework Programme under the PASCAL2 Network of Excellence (ICT-216886).

Appendix A. Relation of MKL to Block-Norm Formulation

For completeness, we show in this appendix the relation of kernel weights formulation of MKL to block-norm formulation.

A.1 The Case $p \in [1,2]$

We show that denoting $\boldsymbol{w} = (\boldsymbol{w}^{(1)}, \ldots, \boldsymbol{w}^{(M)})$, for any $q \in [1,\infty]$, the hypothesis class

$$\left\{ f : x \mapsto \sum_{m=1}^{M} \left\langle \boldsymbol{w}_m, \sqrt{\theta_m}\phi_m(x) \right\rangle \;\middle|\; \|\boldsymbol{w}\|_2 \le D,\ \|\boldsymbol{\theta}\|_q \le 1 \right\}, \tag{33}$$

is identical to the block norm class

$$H_{p,D,M} = \left\{ f : x \mapsto \langle \boldsymbol{w}, \phi(x) \rangle \;\middle|\; \|\boldsymbol{w}\|_{2,p} \le D \right\} \tag{34}$$

where $p := \frac{2q}{q+1}$. This is known since Micchelli and Pontil (2005). To this end, first we rewrite (33) as

$$H_{p,D,M} = \left\{ f : x \mapsto \langle \boldsymbol{w}, \phi(x) \rangle \;\middle|\; \sum_{m=1}^{M} \frac{\|\boldsymbol{w}_m\|_2^2}{\theta_m} \le D^2,\ \|\boldsymbol{\theta}\|_q \le 1 \right\}. \tag{35}$$

However, solving

$$\inf_{\boldsymbol{\theta}} \quad \frac{1}{2} \sum_{m=1}^{M} \frac{\|\boldsymbol{w}_m\|_2^2}{\theta_m}, \quad \text{s.t.} \quad \|\boldsymbol{\theta}\|_q \le 1$$

for fixed $\boldsymbol{w} > \boldsymbol{0}$, the optimal $\boldsymbol{\theta}$ is attained at

$$\theta_m = \frac{\|\boldsymbol{w}_m\|_2^{\frac{2}{q+1}}}{\left(\sum_{m'=1}^{M} \|\boldsymbol{w}_{m'}\|_2^{\frac{2q}{q+1}} \right)^{1/q}}, \qquad \forall m = 1, \ldots, M.$$

Plugging the latter into (35), we obtain (34) with $p = \frac{2q}{q+1}$, which was to show.

A.2 The Case $p \in]2, \infty]$

Even if $p > 2$, we can obtain an alternative formulation of the block norm MKL problem, as shown in Aflalo et al. (2011), by the definition of the dual norm $\|\cdot\|_*$ of a norm $\|\cdot\|$, that is, $\|x\|_* = \sup_{\boldsymbol{y}} \langle \boldsymbol{x}, \boldsymbol{y} \rangle - \|y\|$, it holds

$$\|\boldsymbol{w}\|_{2,p}^2 \;=\; \Big\| \Big(\|\boldsymbol{w}_m\|_2^2 \Big)_{m=1}^M \Big\|_{p/2} = \sup_{\boldsymbol{\theta}: \|\boldsymbol{\theta}\|_{(p/2)^*} \leq 1} \; \sum_{m=1}^M \theta_m \|\boldsymbol{w}_m\|_2^2 \,.$$

Thus defining $q := (p/2)^*$ we can obtain a learning-the-kernel MKL formulation from the above equation. A difference to the case $p < 2$ lies in the kernel weights θ_m appearing in the nominator instead of the denominator.

Appendix B. Lemmata and Proofs

The following result gives a block-structured version of Hölder's inequality (e.g., Steele, 2004).

Lemma 15 (Block-structured Hölder inequality)**.** *Let $\boldsymbol{x} = (\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(m)})$, $\boldsymbol{y} = (\boldsymbol{y}^{(1)}, \ldots, \boldsymbol{y}^{(m)}) \in \mathcal{H} = \mathcal{H}_1 \times \cdots \times \mathcal{H}_M$. Then, for any $p \geq 1$, it holds*

$$\langle \boldsymbol{x}, \boldsymbol{y} \rangle \leq \|\boldsymbol{x}\|_{2,p} \|\boldsymbol{y}\|_{2,p^*} \,.$$

Proof By the Cauchy-Schwarz inequality (C.-S.), we have for all $\boldsymbol{x}, \boldsymbol{y} \in \mathcal{H}$:

$$\begin{aligned} \langle \boldsymbol{x}, \boldsymbol{y} \rangle \quad &= \quad \sum_{m=1}^M \langle \boldsymbol{x}^{(m)}, \boldsymbol{y}^{(m)} \rangle \overset{\text{C.-S.}}{\leq} \sum_{m=1}^M \|\boldsymbol{x}\|_2 \|\boldsymbol{y}\|_2 \\ &= \quad \big\langle (\|\boldsymbol{x}^{(1)}\|_2, \ldots, \|\boldsymbol{x}^{(M)}\|_2), (\|\boldsymbol{y}^{(1)}\|_2, \ldots, \|\boldsymbol{y}^{(M)}\|_2) \big\rangle . \\ &\overset{\text{Hölder}}{\leq} \quad \|\boldsymbol{x}\|_{2,p} \|\boldsymbol{y}\|_{2,p^*} \end{aligned}$$

■

Proof of Lemma 3 (Rosenthal + Young) It is clear that the result trivially holds for $\frac{1}{2} \leq p \leq 1$ with $C_q = 1$ by Jensen's inequality . In the case $p \geq 1$, we apply Rosenthal's inequality (Rosenthal, 1970) to the sequence $X_1, \ldots, X_n$ thereby using the optimal constants computed in Ibragimov and Sharakhmetov (2001), that are, $C_q = 2$ ($q \leq 2$) and $C_q = \mathbb{E}Z^q$ ($q \geq 2$), respectively, where Z is a random variable distributed according to a Poisson law with parameter $\lambda = 1$. This yields

$$\mathbb{E}\left(\frac{1}{n} \sum_{i=1}^n X_i \right)^q \leq C_q \max\left(\frac{1}{n^q} \sum_{i=1}^n \mathbb{E} X_i^q, \left(\frac{1}{n} \sum_{i=1}^n X_i \right)^q \right). \tag{36}$$

By using that $X_i \leq B$ holds almost surely, we could readily obtain a bound of the form $\frac{B^q}{n^{q-1}}$ on the first term. However, this is loose and for $q = 1$ does not converge to zero when $n \to \infty$. Therefore,

we follow a different approach based on Young's inequality (e.g., Steele, 2004):

$$\begin{aligned}\frac{1}{n^q}\sum_{i=1}^n \mathbb{E}X_i^q &\leq \left(\frac{B}{n}\right)^{q-1}\frac{1}{n}\sum_{i=1}^n \mathbb{E}X_i \\ &\overset{\text{Young}}{\leq} \frac{1}{q^*}\left(\frac{B}{n}\right)^{q^*(q-1)} + \frac{1}{q}\left(\frac{1}{n}\sum_{i=1}^n \mathbb{E}X_i\right)^q \\ &= \frac{1}{q^*}\left(\frac{B}{n}\right)^q + \frac{1}{q}\left(\frac{1}{n}\sum_{i=1}^n \mathbb{E}X_i\right)^q.\end{aligned}$$

It thus follows from (36) that for all $q \geq \frac{1}{2}$

$$\mathbb{E}\left(\frac{1}{n}\sum_{i=1}^n X_i\right)^q \leq C_q\left(\left(\frac{B}{n}\right)^q + \left(\frac{1}{n}\sum_{i=1}^n \mathbb{E}X_i\right)^q\right),$$

where C_q can be taken as 2 ($q \leq 2$) and $\mathbb{E}Z^q$ ($q \geq 2$), respectively, where Z is Poisson-distributed. In the subsequent Lemma 16 we show $\mathbb{E}Z^q \leq (q+e)^q$. Clearly, for $q \geq \frac{1}{2}$ it holds $q+e \leq qe+eq = 2eq$ so that in any case $C_q \leq \max(2, 2eq) \leq 2eq$, which concludes the result. ■

We use the following Lemma gives a handle on the q-th moment of a Poisson-distributed random variable and is used in the previous Lemma.

Lemma 16. *For the q-moment of a random variable Z distributed according to a Poisson law with parameter $\lambda = 1$, the following inequality holds for all $q \geq 1$:*

$$\mathbb{E}Z^q \overset{def.}{=} \frac{1}{e}\sum_{k=0}^\infty \frac{k^q}{k!} \leq (q+e)^q.$$

Proof We start by decomposing $\mathbb{E}Z^q$ as follows:

$$\begin{aligned}\mathbb{E}^q &= \frac{1}{e}\left(0 + \sum_{k=1}^q \frac{k^q}{k!} + \sum_{k=q+1}^\infty \frac{k^q}{k!}\right) \\ &= \frac{1}{e}\left(\sum_{k=1}^q \frac{k^{q-1}}{(k-1)!} + \sum_{k=q+1}^\infty \frac{k^q}{k!}\right) \\ &\leq \frac{1}{e}\left(q^q + \sum_{k=q+1}^\infty \frac{k^q}{k!}\right) \qquad (37)\end{aligned}$$

(38)

Note that by Stirling's approximation it holds $k! = \sqrt{2\pi}e^{\tau_k}k\left(\frac{k}{e}\right)^q$ with $\frac{1}{12k+1} < \tau_k < \frac{1}{12k}$ for all q. Thus

$$\begin{aligned}
\sum_{k=q+1}^{\infty}\frac{k^q}{k!} &= \sum_{k=q+1}^{\infty}\frac{1}{\sqrt{2\pi}e^{\tau_k}k}e^k k^{-(k-q)} \\
&= \sum_{k=1}^{\infty}\frac{1}{\sqrt{2\pi}e^{\tau_{k+q}}(k+q)}e^{k+q}k^{-k} \\
&= e^q\sum_{k=1}^{\infty}\frac{1}{\sqrt{2\pi}e^{\tau_{k+q}}(k+q)}\left(\frac{e}{k}\right)^k \\
&\overset{(*)}{\leq} e^q\sum_{k=1}^{\infty}\frac{1}{\sqrt{2\pi}e^{\tau_k}k}\left(\frac{e}{k}\right)^k \\
&\overset{\text{Stirling}}{=} e^q\sum_{k=1}^{\infty}\frac{1}{k!} \\
&= e^{q+1}
\end{aligned}$$

where for $(*)$ note that $e^{\tau_k}k \leq e^{\tau_{k+q}}(k+q)$ can be shown by some algebra using $\frac{1}{12k+1} < \tau_k < \frac{1}{12k}$. Now by (37)

$$\mathbb{E}Z^q = \frac{1}{e}\left(q^q + e^{q+1}\right) \leq q^q + e^q \leq (q+e)^q,$$

which was to show. ■

Lemma 17. *For any $\boldsymbol{a}, \boldsymbol{b} \in \mathbb{R}^m_+$ it holds for all $q \geq 1$*

$$\|\boldsymbol{a}\|_q + \|\boldsymbol{b}\|_q \leq 2^{1-\frac{1}{q}}\|\boldsymbol{a}+\boldsymbol{b}\|_q \leq 2\|\boldsymbol{a}+\boldsymbol{b}\|_q.$$

Proof Let $\boldsymbol{a} = (a_1, \ldots, a_m)$ and $\boldsymbol{b} = (b_1, \ldots, b_m)$. Because all components of $\boldsymbol{a}, \boldsymbol{b}$ are nonnegative, we have

$$\forall i = 1, \ldots, m: \; a_i^q + b_i^q \leq \left(a_i + b_i\right)^q$$

and thus

$$\|\boldsymbol{a}\|_q^q + \|\boldsymbol{b}\|_q^q \leq \|\boldsymbol{a}+\boldsymbol{b}\|_q^q. \tag{39}$$

We conclude by ℓ_q-to-ℓ_1 conversion (see (20))

$$\begin{aligned}
\|\boldsymbol{a}\|_q + \|\boldsymbol{b}\|_q &= \left\|\left(\|\boldsymbol{a}\|_q, \|\boldsymbol{b}\|_q\right)\right\|_1 \overset{(20)}{\leq} 2^{1-\frac{1}{q}}\left\|\left(\|\boldsymbol{a}\|_q, \|\boldsymbol{b}\|_q\right)\right\|_q = 2^{1-\frac{1}{q}}\left(\|\boldsymbol{a}\|_q^q + \|\boldsymbol{b}\|_q^q\right)^{\frac{1}{q}} \\
&\overset{(39)}{\leq} 2^{1-\frac{1}{q}}\|\boldsymbol{a}+\boldsymbol{b}\|_q,
\end{aligned}$$

which completes the proof. ■

References

J. Aflalo, A. Ben-Tal, C. Bhattacharyya, J. S. Nath, and S. Raman. Variable sparsity kernel learning—algorithms and applications. *Journal of Machine Learning Research*, 12:565–592, Feb 2011.

F. R. Bach. Consistency of the group lasso and multiple kernel learning. *J. Mach. Learn. Res.*, 9: 1179–1225, 2008.

F. R. Bach, G. R. G. Lanckriet, and M. I. Jordan. Multiple kernel learning, conic duality, and the SMO algorithm. In *Proc. 21st ICML*. ACM, 2004.

P. Bartlett and S. Mendelson. Rademacher and gaussian complexities: Risk bounds and structural results. *Journal of Machine Learning Research*, 3:463–482, Nov. 2002.

P. L. Bartlett, O. Bousquet, and S. Mendelson. Local Rademacher complexities. *Annals of Statistics*, 33(4):1497–1537, 2005.

P. L. Bartlett, M. I. Jordan, and J. D. McAuliffe. Convexity, classification, and risk bounds. *Journal of the American Statistical Association*, 101(473):138–156, 2006. (Was Department of Statistics, U.C. Berkeley Technical Report number 638, 2003).

G. Blanchard, O. Bousquet, and P. Massart. Statistical performance of support vector machines. *Annals of Statistics*, 36(2):489–531, 2008.

R. R. Bouckaert, E. Frank, M. A. Hall, G. Holmes, B. Pfahringer, P. Reutemann, and I. H. Witten. WEKA—experiences with a java open-source project. *Journal of Machine Learning Research*, 11:2533–2541, 2010.

O. Bousquet and A. Elisseeff. Stability and generalization. *J. Mach. Learn. Res.*, 2:499–526, March 2002. ISSN 1532-4435.

O. Bousquet, S. Boucheron, and G. Lugosi. Introduction to statistical learning theory. In O. Bousquet, U. von Luxburg, and G. Rätsch, editors, *Advanced Lectures on Machine Learning*, volume 3176 of *Lecture Notes in Computer Science*, pages 169–207. Springer Berlin / Heidelberg, 2004.

C. Cortes. Invited talk: Can learning kernels help performance? In *Proceedings of the 26th Annual International Conference on Machine Learning*, ICML '09, pages 1:1–1:1, New York, NY, USA, 2009. ACM. ISBN 978-1-60558-516-1.

C. Cortes, A. Gretton, G. Lanckriet, M. Mohri, and A. Rostamizadeh. Proceedings of the NIPS Workshop on Kernel Learning: Automatic Selection of Optimal Kernels, 2008. URL `http://www.cs.nyu.edu/learning_kernels`.

C. Cortes, M. Mohri, and A. Rostamizadeh. L2 regularization for learning kernels. In *Proceedings of the International Conference on Uncertainty in Artificial Intelligence*, 2009.

C. Cortes, M. Mohri, and A. Rostamizadeh. Generalization bounds for learning kernels. In *Proceedings, 27th ICML*, 2010.

P. V. Gehler and S. Nowozin. Let the kernel figure it out: Principled learning of pre-processing for kernel classifiers. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 06 2009.

A. Gelman. Causality and statistical learning. *American Journal of Sociology*, 0, 2010.

R. Ibragimov and S. Sharakhmetov. The best constant in the rosenthal inequality for nonnegative random variables. *Statistics & Probability Letters*, 55(4):367 – 376, 2001. ISSN 0167-7152.

J.-P. Kahane. *Some Random Series of Functions*. Cambridge University Press, 2nd edition, 1985.

M. Kloft. *ℓ_p-Norm Multiple Kernel Learning*. PhD thesis, Berlin Institute of Technology, Oct 2011. URL `http://opus.kobv.de/tuberlin/volltexte/2011/3239/`.

M. Kloft, U. Brefeld, P. Laskov, and S. Sonnenburg. Non-sparse multiple kernel learning. In *Proc. of the NIPS Workshop on Kernel Learning: Automatic Selection of Kernels*, dec 2008.

M. Kloft, U. Brefeld, S. Sonnenburg, P. Laskov, K.-R. Müller, and A. Zien. Efficient and accurate lp-norm multiple kernel learning. In Y. Bengio, D. Schuurmans, J. Lafferty, C. K. I. Williams, and A. Culotta, editors, *Advances in Neural Information Processing Systems 22*, pages 997–1005. MIT Press, 2009.

M. Kloft, U. Brefeld, S. Sonnenburg, and A. Zien. ℓ_p-norm multiple kernel learning. *Journal of Machine Learning Research*, 12:953–997, Mar 2011.

V. Koltchinskii. Rademacher penalties and structural risk minimization. *IEEE Transactions on Information Theory*, 47(5):1902–1914, 2001.

V. Koltchinskii. Local Rademacher complexities and oracle inequalities in risk minimization. *Annals of Statistics*, 34(6):2593–2656, 2006.

V. Koltchinskii. Sparsity in penalized empirical risk minimization. *Annales de l'Institut Henri Poincaré, Probabilités et Statistiques*, 45(1):7–57, 2009.

V. Koltchinskii and M. Yuan. Sparsity in multiple kernel learning. *Annals of Statistics*, 38(6): 3660–3695, 2010.

S. Kwapién and W. A. Woyczyński. *Random Series and Stochastic Integrals: Single and Multiple*. Birkhäuser, Basel and Boston, M.A., 1992.

G. Lanckriet, N. Cristianini, L. E. Ghaoui, P. Bartlett, and M. I. Jordan. Learning the kernel matrix with semi-definite programming. *JMLR*, 5:27–72, 2004.

H. Leeb and B. M. Pötscher. Sparse estimators and the oracle property, or the return of Hodges' estimator. *Journal of Econometrics*, 142:201–211, 2008.

C. McDiarmid. On the method of bounded differences. In *Surveys in combinatorics, 1989 (Norwich, 1989)*, volume 141 of *London Math. Soc. Lecture Note Ser.*, pages 148–188. Cambridge Univ. Press, Cambridge, 1989.

S. Mendelson. On the performance of kernel classes. *J. Mach. Learn. Res.*, 4:759–771, December 2003.

C. A. Micchelli and M. Pontil. Learning the kernel function via regularization. *Journal of Machine Learning Research*, 6:1099–1125, 2005.

K.-R. Müller, S. Mika, G. Rätsch, K. Tsuda, and B. Schölkopf. An introduction to kernel-based learning algorithms. *IEEE Neural Networks*, 12(2):181–201, May 2001.

G. Raskutti, M. J. Wainwright, and B. Yu. Minimax-optimal rates for sparse additive models over kernel classes via convex programming. *CoRR*, abs/1008.3654, 2010.

H. Rosenthal. On the subspaces of L_p $(p > 2)$ spanned by sequences of independent random variables. *Israel J. Math.*, 8:273–303, 1970.

B. Schölkopf, A. Smola, and K.-R. Müller. Nonlinear component analysis as a kernel eigenvalue problem. *Neural Computation*, 10:1299–1319, 1998.

J. R. Searle. Minds, brains, and programs. *Behavioral and Brain Sciences*, 3(03):417–424, 1980. doi: 10.1017/S0140525X00005756. URL `http://dx.doi.org/10.1017/S0140525X00005756`.

S. Sonnenburg, G. Rätsch, C. Schäfer, and B. Schölkopf. Large scale multiple kernel learning. *Journal of Machine Learning Research*, 7:1531–1565, July 2006.

N. Srebro and S. Ben-David. Learning bounds for support vector machines with learned kernels. In G. Lugosi and H.-U. Simon, editors, *COLT*, volume 4005 of *Lecture Notes in Computer Science*, pages 169–183. Springer, 2006.

J. M. Steele. *The Cauchy-Schwarz Master Class: An Introduction to the Art of Mathematical Inequalities*. Cambridge University Press, New York, NY, USA, 2004. ISBN 052154677X.

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

M. Stone. Cross-validatory choice and assessment of statistical predictors (with discussion). *Journal of the Royal Statistical Society*, B36:111–147, 1974.

T. Suzuki. Unifying framework for fast learning rate of non-sparse multiple kernel learning. In J. Shawe-Taylor, R. Zemel, P. L. Bartlett, F. Pereira, and K. Weinberger, editors, *Advances in Neural Information Processing Systems 24*. 2011. To appear.

M. Talagrand. Concentration of measure and isoperimetric inequalities in product spaces. *Publications Mathmatiques de L'IHS*, 81:73–205, 1995.

A. Tsybakov. Optimal rates of aggregation. In B. Schölkopf and M. Warmuth, editors, *Computational Learning Theory and Kernel Machines (COLT-2003)*, volume 2777 of *Lecture Notes in Artificial Intelligence*, pages 303–313. Springer, 2003.

S. van de Geer. *Empirical Processes in M-estimation*. Cambridge University Press, 2000.

R. C. Williamson, A. J. Smola, and B. Schölkopf. Generalization performance of regularization networks and support vector machines via entropy numbers of compact operators. *IEEE Transactions on Information Theory*, 47(6):2516–2532, 2001.

H. Xu, S. Mannor, and C. Caramanis. Sparse algorithms are not stable: A no-free-lunch theorem. In *Proceedings of the 46th Annual Allerton Conference on Communication, Control, and Computing*, pages 1299 –1303, 2008.

Y. Ying and C. Campbell. Generalization bounds for learning the kernel problem. In *COLT*, 2009.